23rd International Conference in Central Europe
on
Computer Graphics, Visualization and Computer Vision

in co-operation with

EUROGRAPHICS Association

WSCG 2015

Full Papers Proceedings

Edited by

Marina Gavrilova, University of Calgary, Canada
Vaclav Skala, University of West Bohemia, Czech Republic

23rd International Conference in Central Europe
on
Computer Graphics, Visualization and Computer Vision

in co-operation with

EUROGRAPHICS Association

WSCG 2015

Full Papers Proceedings

Edited by

Marina Gavrilova, University of Calgary, Canada
Vaclav Skala, University of West Bohemia, Czech Republic

Vaclav Skala – Union Agency

WSCG 2015 – Full Papers Proceedings

Editor: Vaclav Skala
c/o University of West Bohemia, Univerzitni 8
CZ 306 14 Plzen
Czech Republic
skala@kiv.zcu.cz http://www.VaclavSkala.eu

Managing Editor: Vaclav Skala

Published and printed by:
Vaclav Skala – Union Agency
Na Mazinách 9
CZ 322 00 Plzen
Czech Republic http://www.UnionAgency.eu

Hardcopy: *ISBN 978-80-86943-65-7*

WSCG 2015

International Program Committee

Andrew, Glassner (United States)
Baranoski, Gladimir (Canada)
Benes, Bedrich (United States)
Benger, Werner (Austria)
Bengtsson, Ewert (Sweden)
Bourke, Paul (Australia)
Dachsbacher, Carsten (Germany)
Daniel, Marc (France)
Daniels, Karen (United States)
Debelov, Victor (Russia)
Feito, Francisco (Spain)
Ferguson, Stuart (United Kingdom)
Gavrilova, Marina (Canada)
Guthe, Michael (Germany)
Jung, Soon Ki (Korea)
Kalra, Prem K. (India)
Klosowski, James (United States)
Kraus, Martin (Denmark)
Linsen, Lars (Germany)
Lu, Aidong (United States)
Mark, Finch (United States)
Molla, Ramon (Spain)
Muller, Heinrich (Germany)
Murtagh, Fionn (United Kingdom)
Oyarzun Laura, Cristina (Germany)
Pan, Rongjiang (China)
Paquette, Eric (Canada)
Patow, Gustavo (Spain)
Pedrini, Helio (Brazil)
Platis, Nikos (Greece)
Renaud, Christophe (France)
Richardson, John (United States)
Rojas-Sola, Jose Ignacio (Spain)
Ruyam, Acar (Turkey)
Segura, Rafael (Spain)
Semwal, Sudhanshu (United States)
Schultz, Thomas (Germany)
Schulz, Hans-Jorg (Germany)
Sousa, A.Augusto (Portugal)
Stroud, Ian (Switzerland)
Szecsi, Laszlo (Hungary)
Teschner, Matthias (Germany)
Tevfik, Akgun (Turkey)
Tokuta, Alade (United States)
Ugur, Gudukbay (Turkey)
Wu, Shin-Ting (Brazil)
Wuensche, Burkhard,C. (New Zealand)
Wuethrich, Charles (Germany)
Zemcik, Pavel (Czech Republic)
Zwettler, Gerald (Austria)

Board of Reviewers

Agathos, Alexander (Greece)
Aires, Kelson (Brazil)
Aliaga-Badal, Carlos (Spain)
Apolinario Junior, Antonio Lopes (Brazil)
Assarsson, Ulf (Sweden)
Ayala, Dolors (Spain)
Bae, Juhee (United States)
Birra, Fernando (Portugal)
Bourke, Paul (Australia)
Brandao, Andre (Brazil)
Bucak, Serhat (United States)
Cakmak, Hueseyin Kemal (Germany)
Carozza, Ludovico (United Kingdom)
Cline, David (United States)
Didandeh, Arman (Canada)
Djado, Khalid (Canada)
dos Santos, Jefersson Alex (Brazil)
Drechsler, Klaus (Germany)
Durikovic, Roman (Slovakia)
Eisemann, Martin (Germany)
El Shafey, Laurent (Switzerland)
Emile, Bruno (France)
Fabio, Pellacini (Italy)
Facon, Jacques (Brazil)
Frejlichowski, Dariusz (Poland)
Fuenfzig, Christoph (Germany)
Galo, Mauricio (Brazil)
Gao, Zhi (Singapore)
Garcia Hernandez, Ruben Jesus (Germany)
Garcia-Alonso, Alejandro (Spain)
Gobron, Stephane (Switzerland)
Gois, Joao Paulo (Brazil)
Gomez-Nieto, Erick (Brazil)
Griffin , Amy (Australia)
Grottel, Sebastian (Germany)
Hast, Anders (Sweden)
Hernandez, Benjamin (United States)
Hinkenjann, Andre (Germany)
Hitomi, Yasunobu (Japan)
Hlawatsch, Marcel (Germany)
Horain, Patrick (France)
Hu, Xianlin (United States)
Hua, Binh-Son (Singapore)
Chajdas, Matthaeus (Germany)
Chen, Ding (Japan)
Chen, Weiya (France)
Iwasaki, Kei (Japan)
Jarabo, Adrian (Spain)
Jeschke, Stefan (Austria)
Jones, Mark (United Kingdom)
Jones, Ben (United States)
Jung, Soon Ki (Korea)
Kahler, Olaf (United Kingdom)
Kasprzak, Wlodzimierz (Poland)
Kerkeni, asma (Tunisia)
Klosowski, James (United States)
Kolcun, Alexej (Czech Republic)
Kraus, Martin (Denmark)
Kriglstein, Simone (Austria)
Kumar, Subodh (India)
Kurillo, Gregorij (United States)
Kurt, Murat (Turkey)
Lange, Benoit (France)
Last, Mubbasir (United States)
Lee, Jong Kwan (United States)
Lee, YT (Singapore)
Leite, Neucimar (Brazil)
Leon, Jean-Claude (France)
Lessig, Christian (Germany)
Li, Bo (United States)
Lin, Yuewei (United States)
Linsen, Lars (Germany)
Little, James (Canada)
Livesu, Marco (Italy)
Loscos, Celine (France)
Lu, Aidong (United States)
Maciel, Anderson (Brazil)
Mantiuk, Radoslaw (Poland)
Marques, Ricardo (France)
Masia, Belen (Spain)
Meiguins, Bianchi (Brazil)

Meng, Weiliang (China)
Menotti, David (Brazil)
Mestre, Daniel,R. (France)
Meyer, Alexandre (France)
Michael, Despina (Cyprus)
Michels, Dominik (United States)
Monti, Marina (Italy)
Montrucchio, Bartolomeo (Italy)
Movania, Muhammad Mobeen (Pakistan)
Mukai, Tomohiko (Japan)
Mura, Claudio (Switzerland)
Nagai, Yukie (Japan)
Nah, Jae-Ho (Korea)
Nanni, Loris (Italy)
Nogueira, Keiller (Brazil)
Nurzynska, Karolina (Poland)
Nyul, Laszlo (Hungary)
Oliveira, Joao Fradinho (Portugal)
Oztimur Karadag, Ozge (Turkey)
Paiva, Jose Gustavo (Brazil)
Parsons, Paul (Canada)
Patane, Giuseppe (Italy)
Paul, Padma Polash (Canada)
Peethambaran, Jiju (India)
Penedo, Manuel (Spain)
Pina, Jose Luis (Spain)
Pobegailo, Alexander (Belarus)
Puig, Anna (Spain)
Ramos, Sebastian (Germany)
Rasool, Shahzad (Singapore)
Reddy, Pradyumna (India)
Rehfeld, Stephan (Germany)
Rind, Alexander (Austria)
Rupprecht, Christian (Germany)
Sadlo, Filip (Germany)
Saito, Shunsuke (United States)
Santagati, Cettina (Italy)
Saraiji, MHD Yamen (Japan)
Saru, Dhir (India)
Seipel, Stefan (Sweden)
Shesh, Amit (United States)
Shi, Xin (China)
Shimshoni, Ilan (Israel)
Schaefer, Gerald (United Kingdom)
Schmidt, Johanna (Austria)
Schultz, Thomas (Germany)
Schwarz, Michael (Switzerland)
Silva, Romuere (Brazil)
Silva, Samuel (Portugal)
Singh, Rajiv (India)
Solis, Ana Luisa (Mexico)
Soriano, Aurea (Brazil)
Souza e Silva, Lucas (Brazil)
Spiclin, Ziga (Slovenia)
Svoboda, Tomas (Czech Republic)
Tavares, Joao Manuel (Portugal)
Teixeira, Raoni (Brazil)
Theussl, Thomas (Saudi Arabia)
Tomas Sanahuja, Josep Maria (Mexico)
Torrens, Francisco (Spain)
Tytkowski, Krzysztof (Poland)
Umlauf, Georg (Germany)
Vasseur, Pascal (France)
Vazquez, David (Spain)
Veras, Rodrigo (Brazil)
Walczak, Krzysztof (Poland)
Wanat, Robert (United Kingdom)
Wang, Lili (China)
Wang, Ruizhe (United States)
Wang, Lisheng (China)
Wenger, Rephael (United States)
Wijewickrema, Sudanthi (Australia)
Wu, YuTing (Taiwan)
Wu, Jieting (United States)
Wuensche, Burkhard,C. (New Zealand)
Xiong, Ying (United States)
Xu, Tianchen (Hong Kong SAR)
Xu, Chang (China)
Yang, Shuang (China)
Yasmin, Shamima (United States)
Yoshizawa, Shin (Japan)
Yu, Hongfeng (United States)
Zheng, Jianping (United States)
Zhong, Li (China)

Data Mining and Data Analytics in Biometric Security

Marina Gavrilova
Department of Computer Science
University of Calgary
Calgary
Canada

ABSTRACT

The security research domain has recently witnessed tremendous growth with respect to all aspects of information access and sharing. There has been notable progress in developing successful approaches to tackle the problem of user authentication. Among those approaches, biometric-based authentication has firmly established itself as one of the most reliable, efficient, and versatile tools for providing discretionary access control to a secure resource or system. While state-of-the art methods for biometric authentication are becoming increasingly more powerful and better understood, the same, unfortunately, cannot be said about security of users populating on-line communities or cyberworld.

Ensuring safe and secure communication and interaction among users and their on-line identities presents unique challenges to academia as well as industry, government, and the public. Despite the fact that those challenges are regularly making headlines in the news, in government reports and in the IT security domain, there is a lack of effort to address this urgent problem. The limited efforts that do exist are currently restricted to network security, password protection, encryption, database security and policy-making efforts. However, one of the most crucial components for ensuring biometric and on-line security: the relationship between communication among users and user authentication, has been largely overlooked. This crucial issue requires a systematic study and a targeted effort to develop effective machine intelligence security solutions for cyberworlds.

SHORT BIOGRAPHY

Marina L. Gavrilova received her M.Sc. in Applied Mathematics from Moscow Lomonosov State University in Moscow, Russia, in 1993, and her Ph.D. in Computer Science from the University of Calgary in 1998. She is currently an Associate Professor in the Department of Computer Science, University of Calgary. Dr. Gavrilova's research interests lie in the areas of biometric security, cognitive sciences, pattern recognition, social networking, and cyberworlds. Prof. Gavrilova is the founder and co-director of the Biometric Technologies Laboratory, with over 120 journal and conference papers, edited special issues, books and book chapters, including the World Scientific Bestseller (2007), Image Pattern Recognition: Synthesis and Analysis in Biometrics and Multimodal Biometrics and Intelligent Image Processing for Security System. Together with Dr. Kenneth Tan, Prof. Gavrilova founded the ICCSA Series of International Events in 2002. She was co-Chair of the International Workshop on Biometric Technologies (BT 2004) and General Chair of International Conference on Cyberworlds (CW2011), and currently serves as Founding Editor-in-Chief of Transactions on Computational Science journal, Springer. Prof. Gavrilova has given invited keynotes and panel lectures at such prestigious international events at INDIN 2003, 3A '06, ICBAKE 2008, ICCSA 2010, ICCI*CC 2011 and 2013, CyberWorlds 2012, GRAPHICON 2012 and appeared as panelist at the 14th Security and Privacy Conference. She has given invited talks at DIMACS Rutgers Univeristy, USA; Bell Labs, USA; Microsoft Research, Redmond, USA; Samsung Research, South Korea; Purdue University, USA, and at other universities worldwide. Her research was profiled in newspaper and TV interviews, most recently featured in Exhibit at National Museum of Civilization, Quebec (2012), on Discovery Channel Canada (2013) and in Business Magazine, Calgary, Alberta (2014).

WSCG 2015

Full Papers Proceedings

Contents

Identification of UV-Fluorescence Components Associated with and Detection of Surface Damage in Green Pepper (*Capsicum annum* L)

Danial Fatchurrahman	Makoto Kuramoto	Naoshi Kondo	Yuichi Ogawa	Tetsuhito Suzuki
Brawijaya University Kyoto University Bio-Sensing Engineering Graduate School of Agriculture Japan, Kyoto (606-8502)	Ehime University Integrated Center for Sciences Division of Synthesis and Analysis Japan, Ehime (790-8577)	Kyoto University Bio-Sensing Engineering Graduate School of Agriculture Japan, Kyoto (606-8502)	Kyoto University Bio-Sensing Engineering Graduate School of Agriculture Japan, Kyoto (606-8502)	Kyoto University Bio-Sensing Engineering Graduate School of Agriculture Japan, Kyoto (606-8502)
danial.fatchurrahman.38z@st.kyoto-u .ac.jp	kuramoto.makoto.mx@ehime-u.ac.jp	kondonao@kais.kyoto-u.ac.jp	ogawayu@kais.kyoto-u.ac.jp	ts@kais.kyoto-u.ac.jp

ABSTRACT

Fluorescence imaging has been used to detect fruit surface damage, but has not yet been applied to vegetables, such as green pepper. In this report, we extract and identify fluorescent components from the exocarp (skin) of green pepper. The fluorescence excitation and emission wavelengths of these extracted compounds were determined using a fluorescence spectrophotometer and identified using nuclear magnetic resonance spectroscopy and mass spectrometry. Red and blue fluorescent components with excitation and emission wavelengths 667 – 685 nm and 400 - 438 nm respectively, were found. In subsequent research, the red fluorescent compounds were targeted, as these compounds have a higher fluorescence intensity, around 97 a.u. Pheophytin *a* is one of these red fluorescent compounds, appearing in the mass spectrum at 871 m/z. Furthermore, when a fluorescence imaging system was set up, with halogen illumination, it was shown that this system could successfully detect surface damage in green pepper.

Keywords

Green pepper, fluorescence, excitation, wavelength, damage, machine vision

1. INTRODUCTION

Inspection for surface damage in green pepper (*Capsicum annum* L), such as scars, abrasion, cuts, bruises, and puncture marks, is an important component of grading and quality control. Not only does this damage lower the quality of the green pepper, but it also reduces shelf life, leading to economic losses. For the fresh market, peppers are initially sorted according to color and the presence of damage; acceptable peppers are then separated out for marketing. While grading bell peppers for color using machine vision can achieve accuracies of up to 96%, the detection of damage is more difficult: achieving accuracies of only around 63% (Shearer and Payne 1990). Finding alternative methods to detect surface damage in green peppers is a worldwide challenge for researchers.

Recently, fluorescence imaging techniques have been developed for characterizing plant tissues, such as leaves, fruit, and vegetables. These techniques have the advantage of being both rapid and non-destructive. Examples of such fluorescence imaging systems include: peel defect detection in citrus (Momin et al. 2013); orange fruit-grading using a machine vision system with a pair of white and UV LED lighting devices and a color CCD camera (Kurita et al., 2009); and rot detect in citrus (Kondo et al., 2009).

Furthermore, chlorophyll fluorescence technology has been used to detect various types of damage in agricultural products. Early postharvest studies used chlorophyll fluorescence techniques to follow the development of chilling injury in banana (*Musa* Group AAA, Subgroup Cavendish); in mango *Mangifera indica* L.) (Smillie et al. 1987); and in cucumber and bell pepper (Tijskens et al. 1994; van Kooten et al. 1994). Moreover, scald development in 'Delicious' apples at harvest was evaluated by DeEll et a1. (1996) using chlorophyll fluorescence. Such an approach for the detection of surface damage in green pepper, though, has yet to be reported.

 ISBN 978-80-86943-65-7

There is, however, reason to think that such chlorophyll fluorescent markers are present in green pepper. It has been reported that the primary pigments in green pepper include chlorophylls *a* and *b* (Deli and Molnar, 2002). Two types of chlorophyll exist in higher plants; chlorophyll *a* and the minor components chlorophyll *b* approximately in a ratio of 3:1 (Lichtentaler and Babani 2004). As isolated pigments in an organic solution, both chlorophyll types exhibit typical fluorescence emission spectra with a high maximum in the red region and a shoulder in the far-red, with an emission range near 690 nm in the red region and near 730-740 nm in the far-red region (Da-wen sun, 2009).

Thus, the objectives of this paper are to identify in-vitro fluorescence components in the exocarp of green pepper. Then to focus on identifying the strong red fluorescence compounds in the exocarp of green pepper. By so doing, we will be able to identify appropriate excitation wavelengths for these components in order to construct an efficient light excitation and detection system. Once the light excitation properties are determined, an imaging system will be set in order to validate the detection of surface damage in green pepper based on fluorescence emission characteristics.

2. MATERIAL AND METHOD

2.1. Sample preparation, extraction and spectra measurements

Green peppers (*Capsicum annum* L; variety Miogi) were collected from a farmer in Kochi Prefecture, Japan to be used in this experiment. Prior to experimentation, the green peppers were stored at 25 ^{0}C for a day to acclimate to room (laboratory) temperature. One kilogram of green pepper exocarp was taken and mixed with 1000 mL methanol and 2000 mL chloroform, crushed finely using a centrifugal mill, and then soaked for a day to extract the fluorescence components. The chloroform and methanol layer of the extract was filtered (Ø125 mm) and then concentrated using a rotary vacuum evaporator (Evaporator Type N-1, EYELA). One hundred ninety four milligram of extract sample from resulted extract was then purified through SiO_2 column chromatography with solvent (40 mL Hexane:40 mL CHCL3 ; 60 mL Hexane:20 mL CHCL3 ; 80 mL CHCL3 ; 47.5 mL CHCL3:2.5 mL MeOH ; 45 mL CHCL3:5 mL MeOH ; 40 mL CHCL3:10 mL MeOH ; 100 mL MeOH) respectively. Approximately 25 test tube fractions were acquired and then checked under UV-A (black light blue lamp) illumination. Two fluorescence patterns were observed, a blue fluorescence was observed in fraction tubes 15 and 16; and a red fluorescence was observed in fraction tubes 20 through to 23. The active blue and red fluorescence fractions were then concentrated using a rotary evaporator to remove all the solvent. The dried active fractions were then dissolved in chloroform and placed in a measuring cell (quartz fluorometer cell, 4 clear windows, Teflon stopper, with pathlengths 10 mm) for UV and fluorescence spectra measurements. A fluoro-spectrophotometer (F-4500, Hitachi, Ltd., Tokyo, Japan) was used to measure excitation and fluorescence spectra. A UV-VIS-NIR spectrometer (U-4000, Hitachi, Ltd., Tokyo, Japan) was used to measure the UV absorbance spectra. Nuclear Magnetic Resonance (NMR) analysis with deuterated chloroform solvent was performed. The ^{1}H NMR spectra were recorded at 500 MHz frequency and a temperature of 23 ^{0}C, using a Bruker AVANCE III NMR instrument. Mass Spectrometry (MS) was also performed, 1 mg of dried sample was dissolved in 1mL of organic methanol as solvent, 10 µL of solution was mixed with matrix (glycerol). The mixed solution was set on the target MS and measured using (JEOL JMS-700 Mass Spectrometer).

2.2. Fluorescence image acquisition

Fluorescence images were taken from intact green pepper to compare with spectrum information, which were obtained during the identification procedure. The system consisted of a CCD camera (FC1450, Takex, Japan), tunable filter (VariSpec, Cambridge Research & Instrumentation, Inc., USA). The spectrum was measured over the 400-720 nm range with a resolution of 10 nm. The camera with a 8 mm focus lens (iris set at 1.2 opening) was adjusted as follows; gain: 255; offset: 255; and shutter speed: 66.76 ms. Four halogen lamps (12V50W-AKW, Philips, Japan), equipped with wide band C-PL (W) 62 mm filters, were used. The images were obtained at a light intensity of 15.540 lx, at a 5 nm interval over the 400-720 nm range.

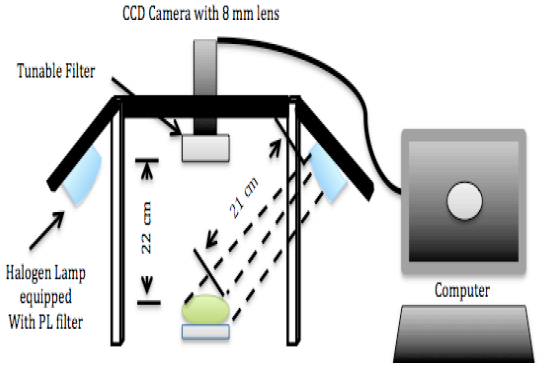


Figure 1a. Image acquisition system (schematic layout picture)

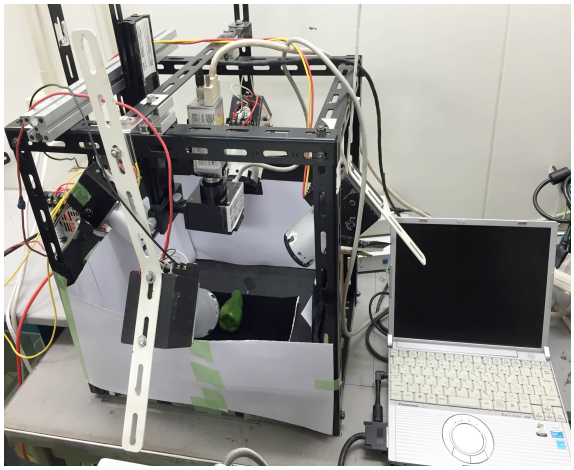


Figure 1b. Real system picture

To detect fluorescence associated with damage, the surface of the green pepper was artificially damaged by knife. The target object was manually placed in the view area of the camera, an image of the target captured and displayed on a computer monitor. The resulting image was processed using programming software MATLAB 2013b (windows platform). The set-up of the image acquisition system is shown in **Figure 1 a to b.**

3. Result and Discussion

3.1. Spectra of extracts

Fluorescence spectra at specific excitation wavelengths are shown in **Figure 2** and **3**. There were two broad peaks in fluorescence, a blue and a red one. The blue fluorescence had an emission range from 400-550 nm, with a maximum peak at 438 nm when excited at 250 nm. The red fluorescence had an emission range from 610-740 nm, with a maximum peak at 685 nm when excited at 667 nm. Blue-green fluorescence in plants mostly originates from the covalently bound blue-green fluorophore ferulic acid present in all plant cell walls; as has been shown by detailed chemical analysis of hydrolyzed cell walls (Harris and Hartley 1976; Lichtenthaler and Schweiger 1998). Blue-green fluorescence is characterized by maximum peaks between 420-430 nm, and a much lower shoulder in the green wavelength region near 520 nm. Our results show a similar pattern. However, it was not confirmed that the blue fluorescence compound of green pepper actually originates from ferulic acid. Further chemical analysis will be necessary to ascertain this.

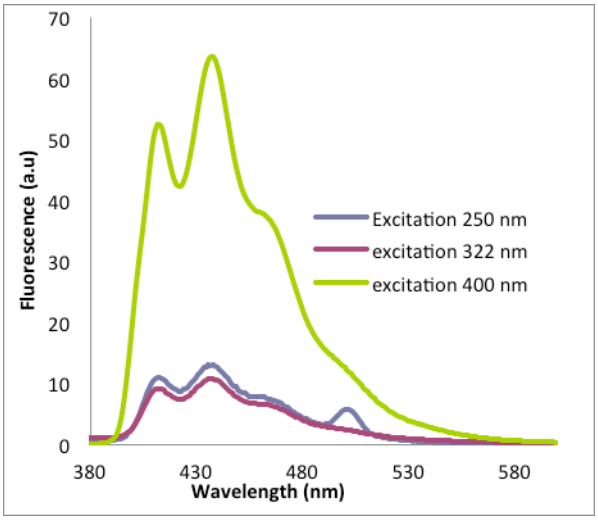


Figure 2. Fluorescence Spectra of Blue Fluorescent

Since the intensity of the red fluorescence was significantly higher than that of blue fluorescence in green pepper: 64 a.u for blue fluorescent and 97 a.u for red fluorescent; we focused on identifying the red fluorescence compound.

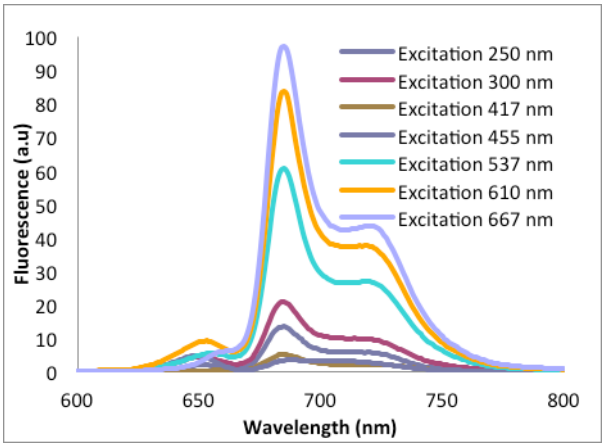


Figure 3. Fluorescence Spectra of Red Fluorescent

3.2. Identification of targeted red fluorescence components

Analysis of fluorescence components was performed using a Nuclear Magnetic Resonance (^{1}HNMR), and Mass Spectroscopy (MS) in order to identify their chemical structures. Results of a NMR and MS spectrum is shown in **Figure 4**. From the NMR spectrum we identified a ^{1}H chemical shift (shown in Figure 4a) of Pheophytin *a* red fluorescence, which is in agreement with what has been reported by Smith et al. (1984). The chemical structure of this is shown in **Figure 5**.

 ISBN 978-80-86943-65-7

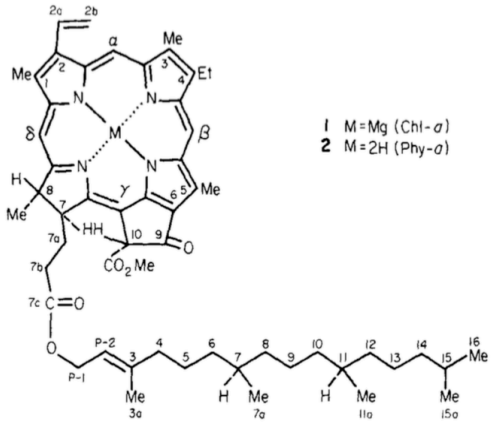


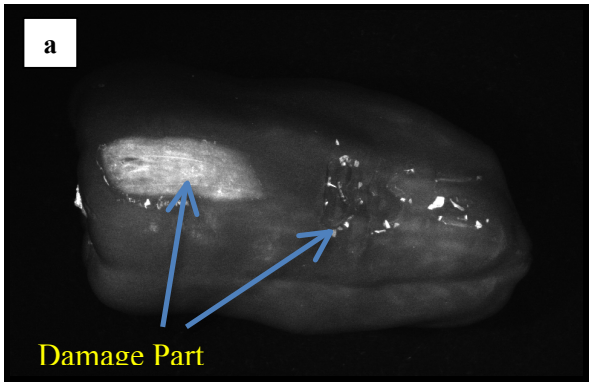
Figure 5. Pheophytin *a*

Further analysis of Pheophytin *a* as the red fluorescence component in green pepper was done by Mass Spectroscopy.

The initial Mass Spectrum of Pheophytin *a* was conducted by Jackson (1979). A removal of Mg-atom from chlorophyll fraction was confirmed by a peak appearing at *m/z* 871, which is in agreement with Lim (2009). This leads to the identification of the molecular ion as Pheophytin *a*. The mass spectrum shows that the two most abundant fragments–ions were observed: at *m/z* 533 corresponding to $(M\text{-}CH_3COOC_{20}H_{39})^+ = (M\text{-}338)^+$; and 593 corresponding to $(M\text{-}C_{20}H_{38})^+ = (M\text{-}278)^+$, a result that is in agreement with Sanja et al (2012).

3.3. Fluorescence Image

A prototype image acquisition system was set-up using a tunable filter set at 685 nm, in accordance with the identified optimal wavelength for red fluorescent emission, to detect damage in green pepper. The resulting image captured by this system is shown in **Figure 6.** By using image segmentation the damaged part can be clearly distinguished.



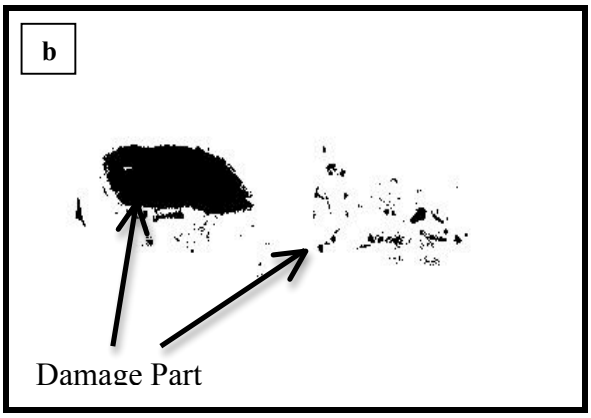


Figure 6. Image resulted from the system, 685 nm wavelength image (a). Processed image threshold minimum 4 and threshold maximum 114 (b).

Thus, it can be seen that a 685 nm wavelength image can successfully identify fluorescence of a different intensity from the damaged part of the green pepper. An explanation for this phenomenon may include reflected light from the surface of the undamaged part of the green pepper, or it could also originate from the red fluorescence of the chlorophyll in green pepper. Further study is necessary to investigate this phenomena.

4. CONCLUSION

We have demonstrated that fluorescence components can be extracted from green pepper, and that these are associated with blue and red fluorescence. As the red fluorescence has a higher emission intensity than that of blue fluorescence, the red fluorescent component was targeted and determined to be Pheophytin *a*, which has fluorescence emission peak at 685 nm (intensity of 97 a.u.) when excited at 667 nm. Furthermore, a prototype fluorescence imaging system based on the above characteristics demonstrated that surface damage on a green pepper can be successfully identified. Thus, damage on a green pepper can be successfully distinguished by fluorescence imaging at 685 nm; a result consistent with red fluorescence emission from the damaged part of the green pepper.

5. ACKNOWLEDGEMENT

Our thanks to Professor Garry Piller, Kyoto University, for his editing and proof reading of the paper. Secondly thanks to Indonesian General of Higher Education for the scholarship given to the author for pursuing Master in Kyoto University.

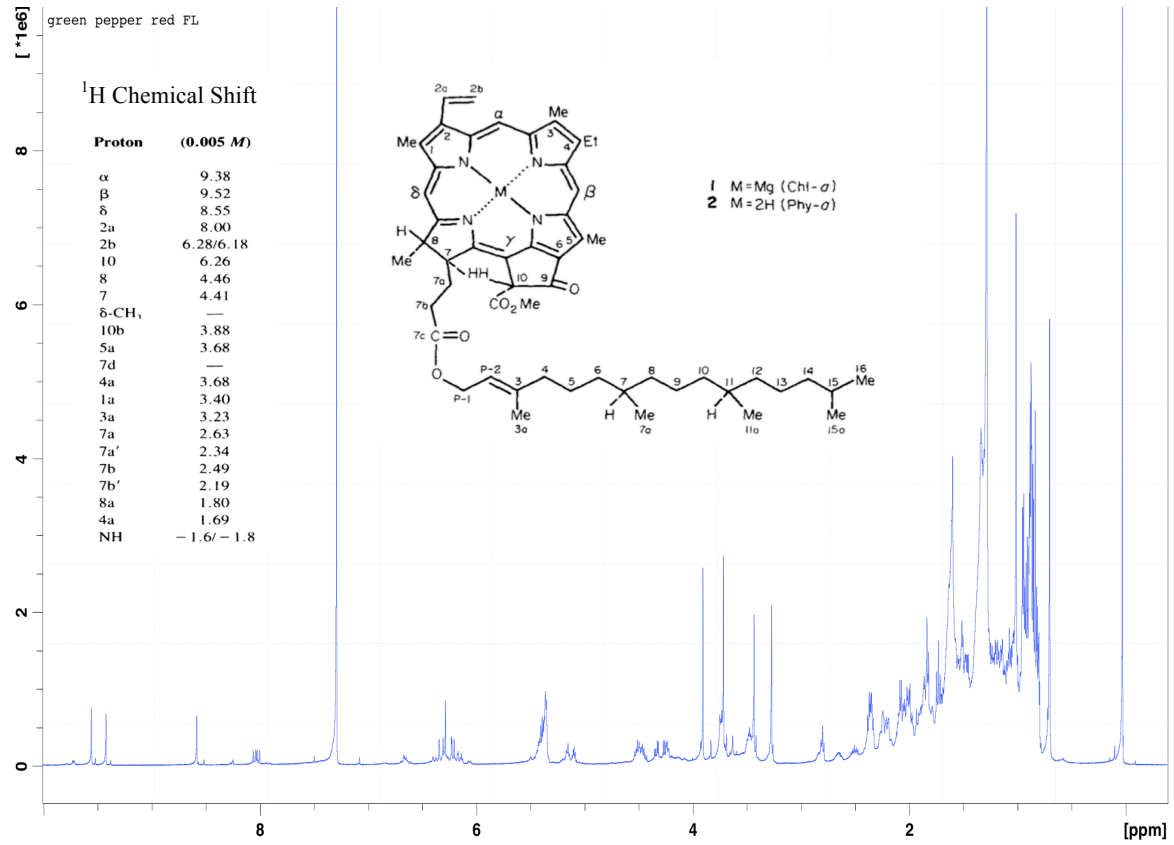


Figure 4 a. Nuclear Magnetic Spectrum (^{1}H NMR) of red Fluorescence

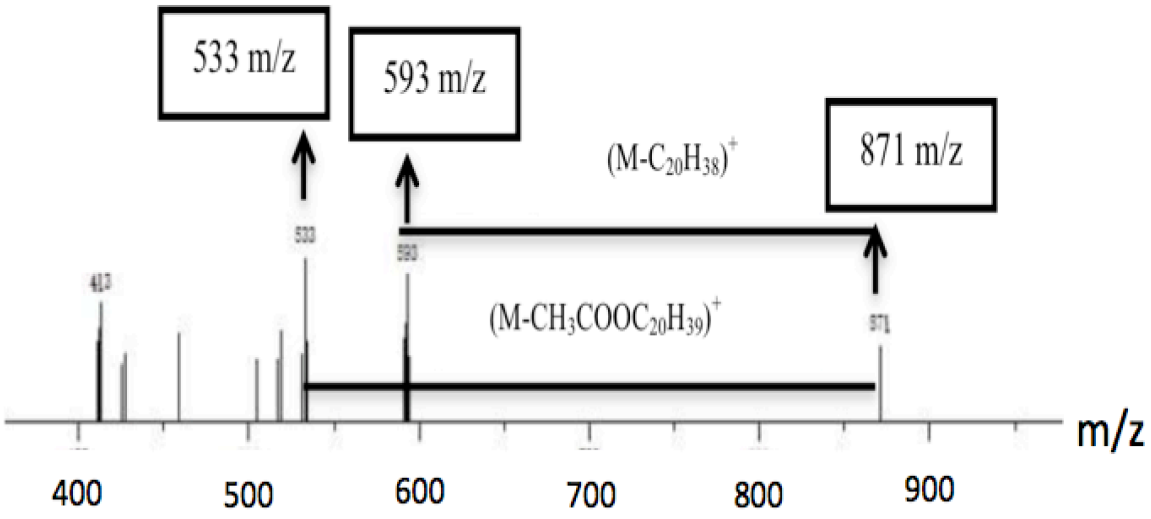


Figure 4b. Mass Spectroscopy spectrum of Red Fluorescence

 ISBN 978-80-86943-65-7

6. REFERENCES

A. H. Jackson. Applications of mass spectrometry in studies of porphyrins and related tetrapyrroles, Philosophical Transactions of the Royal Society. A: Mathematical, Physical and Engineering Sciences, 293 21-37. 1979

C.K. Lim. High Performance Liquid Chromatography and Mass Spectrometry of Phorphyrins, Chlorophylls and Bilins, World Scientific Publishing Co., SGP, 2009, p. 177. 2009

Da-Wen Sun. Optical Monitoring of Fresh and Processed Agricultural Crops. CRC Press is an imprint of the Taylor & Francis Group. 2009.

Deli, J. and Molnar, P. Paprika carotenoids: analysis, isolation, structure elucidation. In: Current Organic Chemistry, Volume 6(13). Bentham Science Publishers, Oak Park, Illinois, pp. 1197–1219. 2002

DeEll, J. R., R. K. Prange, and D. P. Murr. Chlorophyll fluorescence of Delicious apples at harvest as a potential predictor of superficial scald development during storage. Postharvest BioI. TechnoI. 9:1-6. 1996.

Harris, P.J. and R.D. Hartley. Detection of bound ferulic acid in cell walls of the Gramineae by ultraviolet fluorescence microscopy. Nature 259:508–510. 1976.

Kondo, N., M. Kuramoto, H. Shimizu, Y. Ogawa, M. Kurita, T. Nishizu, V. K. Chong and K. Yamamoto. Identification of fluorescent substance in mandarin orange skin for machine vision system to detect rotten citrus fruits. Engineering in Agriculture, Environment and Food 2(2):54-59. 2009.

Kurita, M., N. Kondo, H. Shimizu, P. Ling, P. D. Falzea, T. Shiigi, K. Ninomiya, T. Nishizu and K. Yamamoto. A double image acquisition system with visible and UV LEDs for citrus fruit. Journal of Robotics and Mechatronics 21(4): 533-540. 2009.

Lichtenthaler, H.K. and F. Babani. Light adaption and senescence of the photosynthetic apparatus: Changes in pigment composition, chlorophyll fluorescence parameters and photosynthetic activity during light adaptation and senescence of leaves. In Chlorophyll Fluorescence: A Signature of Photosynthesis, eds. G.C. Papageorgiou and Govindjee, pp. 713–736. Springer, Dordrecht, ISBN: 1-4020-3217-X. 2004.

Lichtenthaler, H.K. and J. Schweiger. Cell wall bound ferulic acid, the major substance of the blue-green fluorescence emission of plants. Journal of Plant Physiology 152:272–282. 1998

Momin Md Abdul, Naoshi kondo, Kazunori ninomiya, Yuichi ogawa. Patterns of Fluorescence Associated with Citrus Peel Defects. Engineering in Agriculture, Environment and Food. EAEF 6(4) : 165-171. 2013.

Sanja M. Milencovic, Jelena B. Zvezdanovic, Tatjana D. Andelkovi. The identification of chlorophyll and its derivatives in the pigment mixtures: HPLC-Chromatography, visible and Mass Spectroscopy studies. 2012.

Shearer, S. A. and Payne, F.A. Color and defect sorting of bell pepers using machine vision. Transactions of the American Society of Agricultural and Biological Enginers 33, 2045-2050. 1990.

Smith, K. M.,Goff, D. A., and Abraham, R. J. The nmr spectra of porphyrins 27. Proton nmr spectra of Chlorophyll *a* and pheophytin *a, Org. Magn. Reson.,* 22, 779. 1984.

Tijskens, L. M. M., E. C. Otma, and O. van Kooten. Photosystem II quantum yield as a measure of radical scavengers in chilling injury in cucumber fruits and bell peppers. A static, dynamic and statistical model. Planta 194:478-486. 1994.

van Kooten, *D.,* L. M. M. Tijskens, and E. C. Otma. Photosystem II quantum yield as a measure of radical scavengers in chilling injury in fruits of tropical origin. A model of approach. Proc. Int. Conference on Cooling and Quality of Fresh Vegetables. p.30-41. 1994.

Cuts in Organs with Internal Structures

D. G. Balreira[1], A. Maciel[1], L.T. Cavazzola[2], M. Walter[1]

[1]Institute of Informatics - Federal University of Rio Grande do Sul (UFRGS) - Brazil

[2]Institute of Health Sciences - Federal University of Rio Grande do Sul (UFRGS) - Brazil

dgbalreira, amaciel, marcelo.walter@inf.ufrgs.br, cavazzola@gmail.com

ABSTRACT

Current simulations in virtual surgery use three-dimensional representations of organs without any internal structure. For some applications, however, there is a need to represent also the organs internal anatomical structures, such as blood vessels. We present, in this paper, a technique that allows arbitrarily oriented cuts through objects, particularly anatomical structures, reconstructing the mesh surface in the cutting zone. In the process, all internal structures participate in the final rendering of the generated surface. As a case study, we selected a human liver model with vessels and present the internal visualization of the liver in real time for arbitrary cutting planes. Our work has applications, for instance, in improving current state-of-the-art surgery simulators for training of students and medical doctors. Simulations present many advantages over other training since they reduce time and cost spent by professionals, offering less risk to the patients. Besides, studies show that the amount of realism seen in the simulators is positively correlated to the engaging of students in learning.

Keywords

Computer graphics, visualization system, solid textures, cuts in objects

1 INTRODUCTION

For many applications in Computer Graphics, geometric representations using only the surface of objects (boundary representation) is enough. However, there are some applications that require visualization and possibly interaction with the interior of the objects. Surgical simulators are an example. Surgical operations such as hepatectomy [Clavien et al., 2007] for instance, require cutting the liver in distinct regions depending on the condition of each patient. Furthermore, this possibility is missing from current research on virtual simulators [Delingette and Ayache, 2005] [Echegaray et al., 2014] [Endo et al., 2014] which have focused their work on other aspects of the simulation.

Current solid texturing techniques generate seamless textures inside objects. However, they cannot deal with cases when there are other objects inside the surface geometry, such as blood vessels inside the liver. This paper proposes a new technique that allows arbitrary cuts in objects, particularly anatomical structures. The technique allows not only the reconstruction of the surface's texture in the cutting zone, but also takes into account the object's internal structures. Our approach considers the geometric object as a whole, composed by surface and inside, and reconstructs the mesh in the cutting zone. Finally, it renders an internal texture on the cutting surface. In Fig. 1 we show an example of a result from our approach. Next generation surgical simulators, for instance, can benefit from this approach in

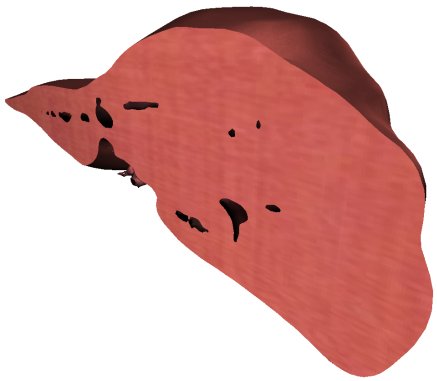


Figure 1: Result from our technique. A liver with a cut showing the blood-vessels as empty shapes. Information about this result as cut 1 in Table 2.

 ISBN 978-80-86943-65-7

order to improve realism, and, therefore, skills development.

2 RELATED WORK

In this section, we discuss selected related work in the field of solid texturing.

The idea of solid texturing or 3D texturing was introduced in 1985 [Perlin, 1985] [Peachey, 1985] and has evolved much since. Solid textures are defined as a process in which a texture generating function is evaluated directly on $\mathbb{R}^3$. This function defines a volume, and the object looks as if it was *carved* from this volume. A survey published by Pietroni and colleagues [Pietroni et al., 2010] presents a detailed review of several techniques for solid texturing and texture synthesis. From the original papers, other researchers [Ghazanfarpour and Dischler, 1995] [Ghazanfarpour and Dischler, 1996] extended the idea using spectral analysis and texture synthesis applied to 2D orthogonal views [Dischler et al., 1998]. Another method adapted a texture synthesis process called optimization-based with histogram matching, globally minimizing its energy function [Kopf et al., 2007]. In the following year, a procedure introduced a new technique to synthesize solid textures restraining them to a voxel subset, forcing spatial determinism [Dong et al., 2008]. Recently, a new approach tried to deal particularly with regular and semi-regular patterns [Du et al., 2013].

Cutler et al. [2002] presented a scripting language in order to define internal layers for objects. Two years later, [Owada et al., 2004] proposed a new method that consists in specifying the interior of an object by using a browsing and modeling interface, controlled by the user. Pietroni et al. [2007] used few images representing cross sections of an object in order to render any point inside it. Takayama's research [Takayama et al., 2008] extended the concept of lapped textures [Praun et al., 2000] to solid textures, covering the whole object's volume instead of only its surface. In the same year, a new system called volume painter [Owada et al., 2008] projected volumes from sketches defined by the user. In 2010, a new concept called diffusion surfaces [Takayama et al., 2010] was able to render structures that have a smooth color variation in its internal structures, like fruits and vegetables.

From the above review, it can be understood that, in the current state-of-the-art on solid texturing, no technique deals with objects with internal structures. Our technique considers internal structures and allows visually and geometrically consistent arbitrary cuts on the objects, extending, in this way, the state-of-the-art in solid texturing.

3 OUR TECHNIQUE

This section presents our technique for dealing with arbitrary cuts in objects with internal structures. Overall, our algorithm receives two 3D meshes and a set of three images as input. The two meshes represent respectively an object's surface and the content in its interior. The three images are orthogonal samples of the 3D internal texture of the object.

Then, for any given cutting plane, our technique returns a consistent geometry – object's surface plus interior – and a texture to be mapped on the region defined by the cut. We generate the final solid texture by an interpolation function applied at each point on the cutting plane. If the plane intersects with the internal structure, the triangles in the overlap area are defined as holes and thus are not rendered. Figure 3 presents the pipeline of our technique with the two primary processes labeled: A-Remeshing the Objects, detailed in Section 3.1; and B-Texturing the Object, detailed in Section 3.2.

3.1 Remeshing the Objects

This section describes the sequence of steps for remeshing the input objects, surface and interior, according to the cutting plane. Our algorithm works for any triangular mesh, even concave and with disconnected parts. At the end of process A, each object is sectioned at the intersection of the model with the cutting plane, the surface model is retriangulated, and the internal parts are flagged as holes.

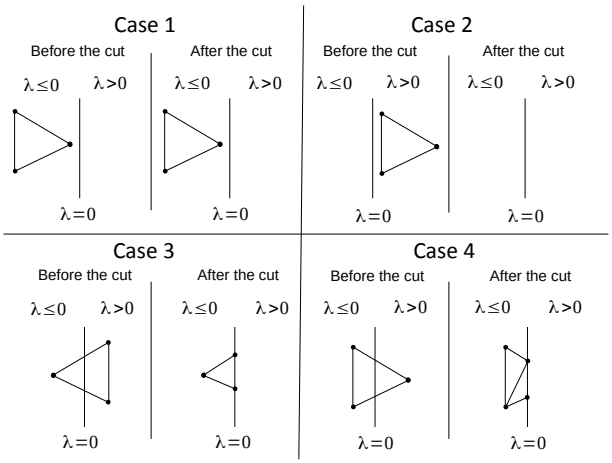


Figure 2: Possible cases of cuts applied to each triangle of the mesh in 2D view.

3.1.1 Cutting the Models

The first step consists of sectioning the object's triangles according to the cutting plane. Depending on the spatial location of the vertices with respect to the plane, defined by the λ value, different solutions are applied. We calculate λ by applying the coordinates of each triangle vertex on the equation of the cutting plane, producing the possible cases seen in Figure 2.

For all cases, the three vertices of each triangle are tested. In case 1, if λ of all vertices satisfy $\lambda \leq 0$, they

 ISBN 978-80-86943-65-7

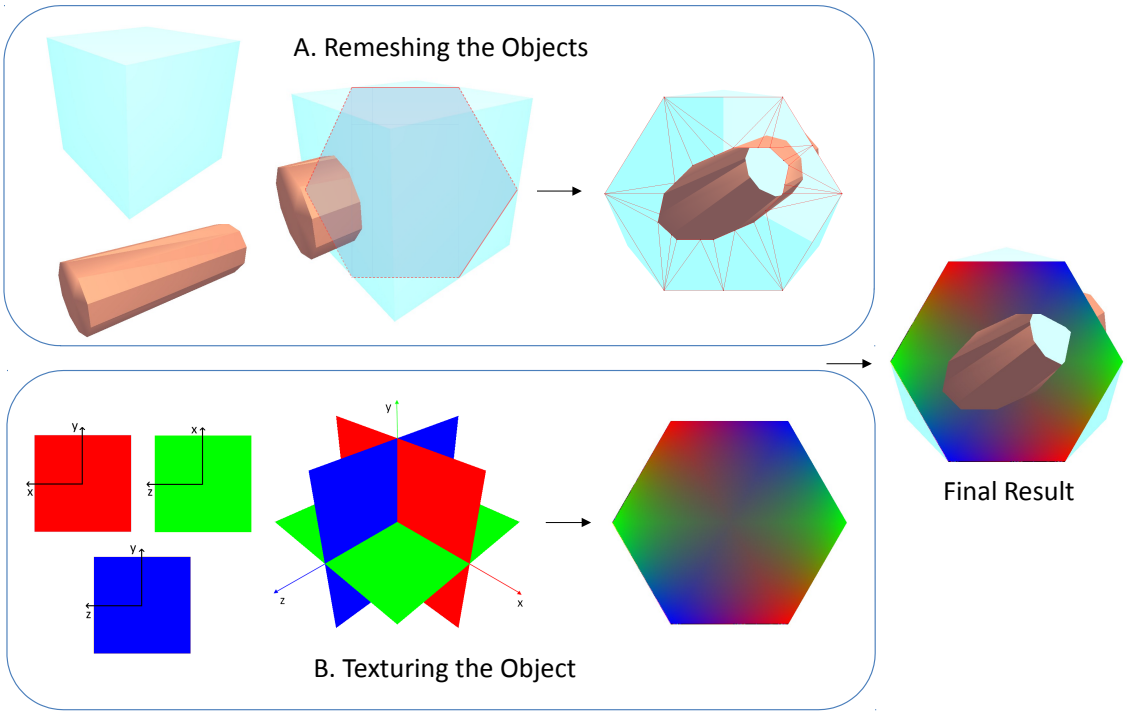


Figure 3: Overview of our technique. Given two 3D meshes, a set of three images and a cutting plane as input, our technique calculates the resulting geometry (A) and a texture to be mapped on the cutting zone (B).

remain the same because they are in the side of the object that remains. In Case 2, if λ of all vertices satisfies $\lambda > 0$, the triangle is removed from the object, since it is on the side which will be eliminated. In Case 3, if only one of the three λ values satisfies $\lambda \leq 0$, then the remaining two vertices are repositioned to the intersection point of the plane with the triangle. In the last case, if two λ satisfy $\lambda \leq 0$, the remaining vertex is moved to one of the intersection points and another triangle is created connecting the two intersection points with the generated triangle. At the end of this step, the results are consistent new meshes taking into account the cutting plane, as shown in Figure 4.

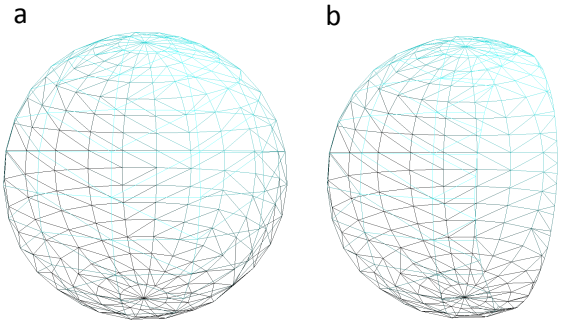


Figure 4: Example of a sphere model before (a) and after (b) the cut.

3.1.2 *Segmentation in Topologically Connected Sets*

The result of the previous step is the two original objects modified by the cut. Now we need to group sets of vertices into topologically connected sets, called *segments*. This step classifies in separate groups all the connected vertices on the cutting plane and allows our solution to work with geometries where a cut will split the original object into two disjoint parts. The key idea is to make a depth search along the vertices which were cut in the previous step, searching for the neighboring ones under the same conditions until the first vertex is reached again, forming a topologically connected set and their respective order. Figure 5 shows an example where two connected sets are detected.

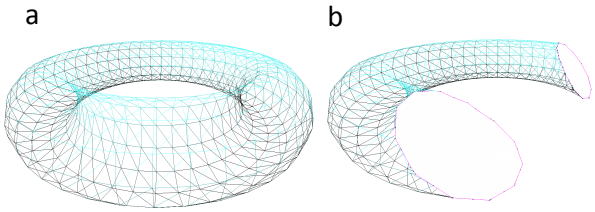


Figure 5: Example of a torus mesh before (a) and after (b) being cut by a plane. In (b) two distinct connected sets can be observed.

3.1.3 *Transforming the vertices to 2D*

All the vertices in the cut area already segmented in connected sets are defined in world coordinates. In

order to apply two-dimensional algorithms in the next steps, these vertices are transformed to a new basis determined by u, v and n, with n being the cutting plane's normal vector.

3.1.4 Computing the Holes for each Segment

The triangulation algorithm, presented in the next step, needs information about which parts are of the surface object and which parts are of the interior of the object, to triangulate only the parts which are part of the object's surface. We need therefore to identify these parts. We call h a 2D point located inside a two-dimensional segment which will not be triangulated (a hole). We compute h by calculating the average of the first two ordered intersection points found between a line connecting one of the diagonals from the hole's bounding box and the intersected edges (Figure 6).

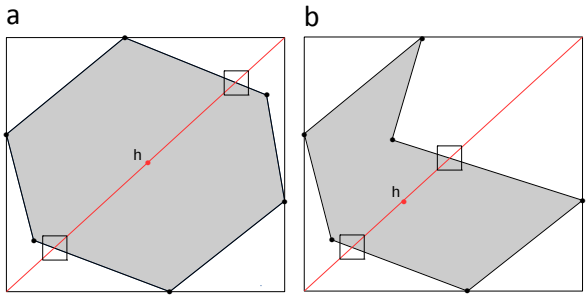


Figure 6: Examples of holes. h marks the point inside the hole. It is computed as the average of the two intersection points with one of the diagonals.

After, we apply a point-in-polygon algorithm to check if h is inside the hole. Otherwise, we repeat the process with the remaining diagonal, as seen in Figure 7.

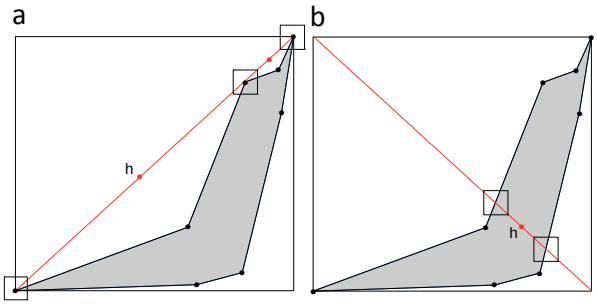


Figure 7: A case when the average of the intersection points is outside the hole (a). In this case we use the other diagonal to compute h (b).

3.1.5 Triangulation

Once we have all the segments and holes identified, we can compute a triangulation of the segments; holes are not triangulated. We use the *Triangle* library presented by Shewchuk [1996] to triangulate the vertices that define a segment. This library correctly creates a set of triangles leaving the holes without triangles. Figure 8 shows an example for two disconnected segments, one of them containing a hole, marked in gray in the figure.

3.1.6 Removal of External Triangles

In this step, we remove all the triangles that are outside the original segments by calculating the barycenter of each one followed by a point-in-polygon technique, as shown in Figure 8. Since we decided to compute the triangulation algorithm only once due to its cost instead of for each segment separately, this simple procedure correctly eliminates all triangles which are not part of the mesh.

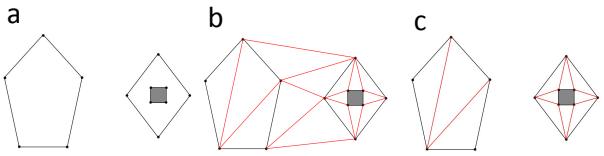


Figure 8: A set of vertices grouped in two segments with a hole in the second one (hatched) (a). Respective triangulation of the set in red lines before (b) and after (c) the removal of external triangles.

3.1.7 Mapping back to 3D

Since the next step, texturing, requires the points in world coordinates, we transform the computed triangles back to world coordinates. This is the last step in remeshing the original objects according to the cutting plane, resulting in a new cut object, triangulated at the defined plane.

3.2 Texturing the Object

This section describes the steps comprising the process B of Figure 3, *Texturing the Object*. At the end of this process, we will obtain a rendering for the cutting plane taking into account the holes. We use a simple solid texturing technique based on the interpolation of images. Although this solution can be used on its own as a texturing technique, we will explain it as a continuation of the previous section. Given the triangles on the plane already computed and three images defining the object's internal texture, we apply a function that interpolates the images, returning a color for each point according to its spatial location.

3.2.1 Color Sampling

Let I be an image formed by a matrix $I_w \cdot I_h$ of pixels, each pixel accessed by $I_{i,j}$ with i as rows and j as columns. The three input images denoted by I_k, $k \in \{1,2,3\}$ are set in the orthogonal planes $z = 0$, $y = 0$ and $x = 0$, respectively. Also, they are centered at the origin of the surface object and limited according to the object's bounding box defined by $max = (max_x, max_y, max_z)$ and $min = (min_x, min_y, min_z)$.

 ISBN 978-80-86943-65-7

Our color sampling technique can compute a color for any point inside this domain although we are only interested in the cutting plane's triangles. Figure 9 shows a graphical representation of the images on the three-dimensional space.

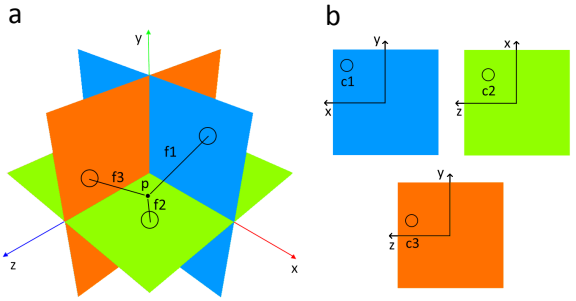


Figure 9: Color sampling for a point p from three images arranged in the 3D space (a) and their respective color contributions c_k (b).

Let c_k, $k \in \{1,2,3\}$ be the color related to a given pixel for each of the input images I_k and p a point defined inside the object's bounding box. The positions I_{k_i} and I_{k_j} of each pixel corresponding to each of the images associated with the point p can be calculated as:

$$I_{1_{i,j}} = \left\{ \left\lfloor \frac{(max_y - p_y)}{(max_y - min_y)} \cdot I_{1_h} \right\rceil, \left\lfloor \frac{(max_x - p_x)}{(max_x - min_x)} \cdot I_{1_w} \right\rceil \right\}$$

$$I_{2_{i,j}} = \left\{ \left\lfloor \frac{(max_x - p_x)}{(max_x - min_x)} \cdot I_{2_h} \right\rceil, \left\lfloor \frac{(max_z - p_z)}{(max_z - min_z)} \cdot I_{2_w} \right\rceil \right\}$$

$$I_{3_{i,j}} = \left\{ \left\lfloor \frac{(max_y - p_y)}{(max_y - min_y)} \cdot I_{3_h} \right\rceil, \left\lfloor \frac{(max_z - p_z)}{(max_z - min_z)} \cdot I_{3_w} \right\rceil \right\}$$

Therefore, each one of the colors associated with the point p can be computed as $c_k = I_{k_{i,j}}$.

3.2.2 Calculating the Weight Factors

In this step, we interpolate the three colors computed in the previous step in order to generate a unique color for the point p. This interpolation is based on the distance between p and the images, where each c_k has an associated weight factor when computing the final color.

Let f be a weight factor corresponding to a contribution percentage of an image I on a point p. We calculate the auxiliary factors f_q as:

$$f_q = 1 - \left| 1 - \frac{(max_q - p_q)}{\frac{max_q - min_q}{2}} \right|,$$

where q indicates the axes z, y and x, respectively.

Given the factors f_k related to the images I_k, as in Figure 9, then:

$$f_k = \frac{f_q}{f_x + f_y + f_z}$$

3.2.3 Rendering by Texture Mapping

This step has two main goals: the first shows how to obtain the final color of the point p and the second explains how to render the triangles on the cut plane using texture mapping.

We generate the final color c for a given point p as:

$$c = \sum_{k=1,2,3} c_k f_k \tag{1}$$

In order to obtain the final texture for the triangles that define the cutting plane, we use a two-dimensional regular grid on the plane to sample the texture and create a texture map. We set this grid with n lines and m columns on the two-dimensional bounding box defined according to the triangles to be rendered in the plane space.

We compute the colors of these points with equation (1), assigning these colors directly onto a new image I. Further, we define this image as a texture and normalized (u,v) coordinates for the vertices on the plane. Finally, we map the created texture on the triangles and render the image. This process is illustrated in Figure 10.

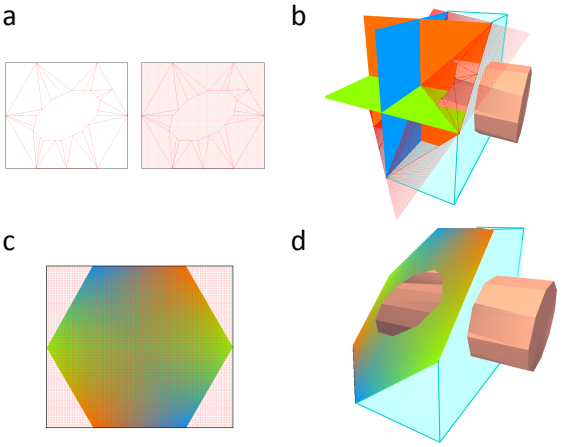


Figure 10: Rendering and texture mapping process. Given a triangulation and the points on the grid in the plane space (a), the points are represented in the world space (b). Then, the colors are calculated, producing a new texture image (c) that is mapped on the triangles to be rendered (d).

4 RESULTS AND DISCUSSION

We used OpenGL for the graphics API and the tests ran on an Intel Core i7-4770 CPU 3.40 GHz with 16GB

RAM, Windows 8 64 bits and NVIDIA GeForce GTX 770 for graphics. We choose a liver model with blood vessels as a case study to test our solution. We constructed the models from CT images of the SLIVER07 project [Heimann et al., 2009]. We used the Training data - part 1 pack containing about 300 images with resolution 512x512 from the abdominal region already with binary masks to segment the liver. Then we used MeVisLab to obtain the geometric models (Figures 11-a and 11-b) as detailed in Table 1.

Model	Vertices	Triangles
Liver	3350	6673
Vessels	11216	22662

Table 1: Details of the input models.

We used the textures for the liver from Xue-mei et al. [2009] where they addressed texture synthesis for surgery simulators and the texture for the vessels from ElHelw et al. [2004]. We used the same image for the three textures to represent the inner liver. Ideally, three different images should be used. These results can be seen in Figure 11-c. The textured models separately and superimposed are illustrated in Figures 11-d, 11-e and 11-f, respectively.

Next we show several results from the inputs, as a set of cuts generated by our system. Table 2 presents information related to these cuts, including the number of intersection surface points (IP), number of triangles on the plane (TP), plane texture grid resolution (GR) and execution time (ET) in milliseconds.

The cuts are organized into different groups to illustrate our technique for various goals. The first cut is shown with full texture (Figure 1) and in wireframe (Figure 12) to illustrate a plane containing several intersections with vessels.

Cuts 2 to 7 (Figure 13) present six cuts maintaining the same cutting direction. As the grid resolution remains constant, its related execution time increases according to the number of triangles intersecting with the vessels.

Cuts 8 to 13, shown in Figure 14, use the same cutting plane but with increasing resolution of the grid, zoomed in on the right of each cut. In this sequence, we also see an increase in the rendering time, due to the increasing resolutions of the grid. Also, we can barely see differences in the final rendering with resolutions above 256.

For a video illustrating our technique in action, please visit `https://vimeo.com/128415963`. This video was captured directly from the screen.

5 CONCLUSIONS

We presented a technique for cuts in objects with internal structures with possible application on surgery simulators. In general, research related to surgical simulators does not deal with internal structures of organs, lacking information that could increase visual realism. Although we presented our technique focusing on organs, it can also deal with other types of three-dimensional models that have an internal structure. For the sake of generalization, we tested our approach using a watermelon model. Results are presented in Figure 15. The seeds are geometrically modeled as internal objects. The watermelon texture at the cutting plane looks plausible, and the seeds are correctly cut.

After this paper, a number of new avenues can be explored. First, other image combinations can be tested besides the trilinear approach we presented here. Then, validation studies should be planned to, using photographs of real objects, assess the outcome produced by the different texturing interpolations. For the cutting, we aim to extend our approach to support arbitrary cutting surfaces, such as those defined by implicit equations or 3D meshes, instead of a single planar surface. Further, as those improvements will require higher computation power, we intend to explore parallel implementations on the GPU before integrating our solution into an operational surgery simulator.

6 ACKNOWLEDGMENTS

We gratefully acknowledge the partial financial support from FAPERGS through grants 12/1944-2 and 2283-2551/14-8, and CNPq through grants 305071/2012-2 and 478730/2012-8. We appreciate all the annonymous reviewer's contributions, helping us improving our work.

REFERENCES

P.-A. Clavien, H. Petrowsky, M. L. DeOliveira, and R. Graf. Strategies for safer liver surgery and partial liver transplantation. *New England Journal of Medicine*, 356(15):1545–1559, 2007.

B. Cutler, J. Dorsey, L. McMillan, M. Müller, and R. Jagnow. A procedural approach to authoring solid models. *ACM Trans. Graph.*, 21(3), July 2002.

H. Delingette and N. Ayache. Hepatic surgery simulation. *Communications of the ACM*, 48(2):31–36, 2005.

J.-M. Dischler, D. Ghazanfarpour, and R. Freydier. Anisotropic solid texture synthesis using orthogonal 2d views. 17(3):87–95, 1998.

Y. Dong, S. Lefebvre, X. Tong, and G. Drettakis. Lazy solid texture synthesis. In *Proceedings of the Nineteenth Eurographics Conference on Rendering*, EGSR '08, pages 1165–1174, 2008.

S.-P. Du, S.-M. Hu, and R. R. Martin. Semiregular solid texturing from 2d image exemplars. *IEEE Transactions on Visualization and Computer Graphics*, 19 (3):460–469, 2013. ISSN 1077-2626. doi: http://doi.ieeecomputersociety.org/10.1109/TVCG.2012.129.

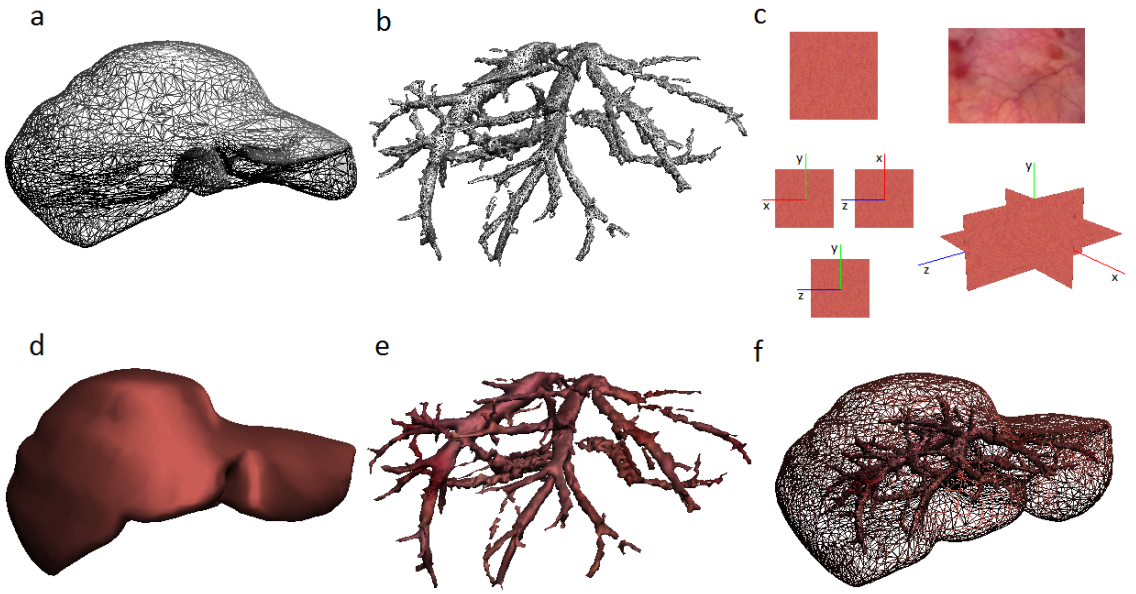


Figure 11: Overview of the models and textures for the case study. Liver (a) and vessels models (b). Textures for the liver and for the vessels and internal images (c). Textured liver (d) and vessels (e), and both superimposed (f).

Cut	1	2	3	4	5	6	7	8	9	10	11	12	13
IP	667	182	586	569	618	476	269	479	479	479	479	479	479
PT	657	178	600	597	647	498	273	492	492	492	492	492	492
GR	256	256	256	256	256	256	256	32	64	128	256	512	1024
ET	136.21	41.32	120.8	118.02	143.25	93.86	55.58	37.73	40.42	51.06	92.49	255.65	905.47

Table 2: Information about the number of intersection points (IP), number of triangles on plane (TP), texture grid resolution (GR) and execution time (ET) in milliseconds for each cut.

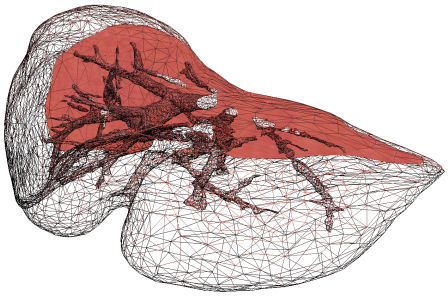


Figure 12: Cut illustrating a plane with several intersections with the vessels in wireframe. Same result presented in Fig. 1

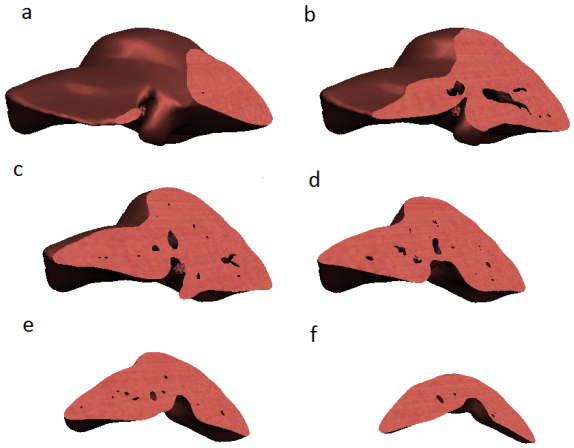


Figure 13: Cuts 2 (a), 3 (b), 4 (c), 5 (d), 6 (e) and 7 (f) presenting six different cuts maintaining the same direction and grid resolution.

G. Echegaray, I. Herrera, I. Aguinaga, C. Buchart, and D. Borro. A brain surgery simulator. *Computer Graphics and Applications, IEEE*, 34(3):12–18, 2014.

M. A. ElHelw, B. P. Lo, A. Darzi, and G.-Z. Yang. Real-time photo-realistic rendering for surgical simulations with graphics hardware. In *Medical Imaging and Augmented Reality*, pages 346–352. Springer, 2004.

K. Endo, N. Sata, Y. Ishiguro, A. Miki, H. Sasanuma, Y. Sakuma, A. Shimizu, M. Hyodo, A. Lefor, and Y. Yasuda. A patient-specific surgical simulator using preoperative imaging data: an interactive simulator using a three-dimensional tactile mouse. *Journal of Computational Surgery*, 3(1):1–8, 2014.

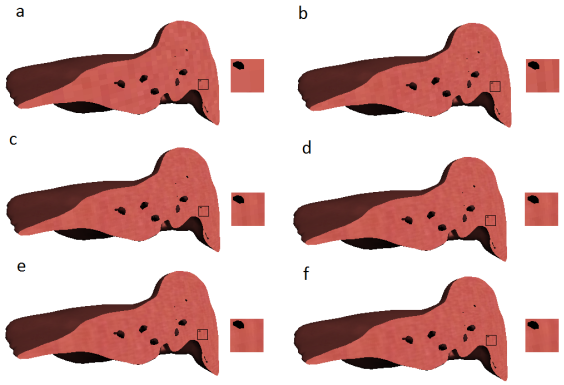


Figure 14: Cuts 8 (a), 9 (b), 10 (c), 11 (d), 12 (e) and 13 (f) showing the same cutting plane for different grid resolutions.

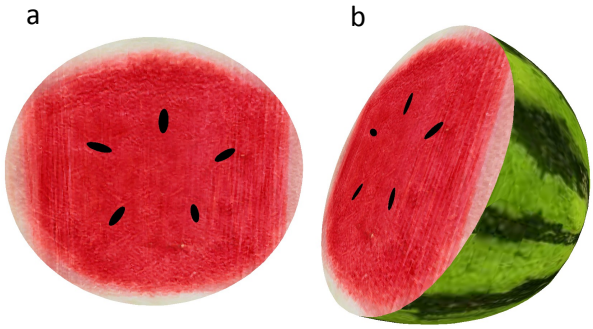


Figure 15: Watermelon cut using our technique.

D. Ghazanfarpour and J.-M. Dischler. Spectral analysis for automatic 3-d texture generation. *Computers & Graphics*, 19(3):413–422, 1995.

D. Ghazanfarpour and J.-M. M. Dischler. Generation of 3D texture using multiple 2D models analysis. 15 (3):C311–C323, C478–C479, Sept. 1996.

T. Heimann, B. Van Ginneken, M. A. Styner, Y. Arzhaeva, V. Aurich, C. Bauer, A. Beck, C. Becker, R. Beichel, G. Bekes, et al. Comparison and evaluation of methods for liver segmentation from ct datasets. *Medical Imaging, IEEE Transactions on*, 28(8):1251–1265, 2009.

J. Kopf, C.-W. Fu, D. Cohen-Or, O. Deussen, D. Lischinski, and T.-T. Wong. Solid texture synthesis from 2d exemplars. *ACM Transactions on Graphics (TOG)*, 26(3):2, 2007.

S. Owada, F. Nielsen, M. Okabe, and T. Igarashi. Volumetric illustration: designing 3d models with internal textures. *ACM Transactions on Graphics (TOG)*, 23 (3):322–328, 2004.

S. Owada, T. Harada, P. Holzer, and T. Igarashi. Volume painter: Geometry-guided volume modeling by sketching on the cross-section. In *Proceedings of the Fifth Eurographics conference on Sketch-Based Interfaces and Modeling*, pages 9–16. Eurographics Association, 2008.

D. R. Peachey. Solid texturing of complex surfaces. *SIGGRAPH Comput. Graph.*, 19(3), July 1985.

K. Perlin. An image synthesizer. *SIGGRAPH Comput. Graph.*, 19(3):287–296, 1985.

N. Pietroni, M. A. Otaduy, B. Bickel, F. Ganovelli, and M. Gross. Texturing internal surfaces from a few cross sections. 26(3):637–644, 2007.

N. Pietroni, P. Cignoni, M. Otaduy, and R. Scopigno. Solid-texture synthesis: a survey. *Computer Graphics and Applications, IEEE*, 30(4):74–89, 2010.

E. Praun, A. Finkelstein, and H. Hoppe. Lapped textures. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 465–470. ACM Press/Addison-Wesley Publishing Co., 2000.

J. R. Shewchuk. Triangle: Engineering a 2d quality mesh generator and delaunay triangulator. In *Applied computational geometry towards geometric engineering*, pages 203–222. Springer, 1996.

K. Takayama, M. Okabe, T. Ijiri, and T. Igarashi. Lapped solid textures: filling a model with anisotropic textures. 27(3):53, 2008.

K. Takayama, O. Sorkine, A. Nealen, and T. Igarashi. Volumetric modeling with diffusion surfaces. *ACM Transactions on Graphics (TOG)*, 29(6):180, 2010.

L. Xue-mei, H. Ai-min, and Z. Qin-ping. Organ texture synthesis for virtual reality-based surgical simulators. In *Computer Science and Engineering, 2009. WCSE'09. Second International Workshop on*, volume 1, pages 238–241. IEEE, 2009.

Real-Time 3-D Wavelet Lifting

David Barina Pavel Zemcik

Faculty of Information Technology
Brno University of Technology
Bozetechova 1/2, Brno
Czech Republic
{ibarina,zemcik}@fit.vutbr.cz

ABSTRACT

This work presents a fast streaming unit for computing a 3-D discrete wavelet transform. The unit can continuously consume source data and instantly produce resulting coefficients. Considering this approach, every input as well as output sample is visited only once. The streaming unit can be further improved by exploiting suitable SIMD instruction set. Depending on the platform, the proposed method reaches speedup $11\times$ and $8\times$ compared to the naive implementation. The measurements presented in the paper confirm the linear theoretical complexity of the transform. Our method requires a constant amount of time to transform a sample independently of the data size.

Keywords

wavelet transform, volumetric data, real-time processing

1 INTRODUCTION

Discrete wavelet transform (DWT) is mathematical tool that uses discretely sampled wavelets to decompose input data into several scales. Regarding multi-dimensional signals, every such scale is also split into several directionally selective subbands. The transform is used as the basis of many sophisticated algorithms and applications. A forward direction of the transform is composed of several consecutive levels of signal decompositions. These can be briefly understood as convolutions with two complementary FIR filters – low-pass and high-pass one. An inverse direction of the transform has a symmetric nature to the forward one.

Considering the number of arithmetic operations, a lifting scheme [12] is often the most efficient way for computing the discrete wavelet transform. The entire calculation consists of several lifting steps. These steps alternately update odd and even intermediate results.

This paper focuses on the Cohen-Daubechies-Feauveau (CDF) 9/7 wavelet [11], which is often used for image compression (e.g., JPEG 2000 standard). The resulting coefficients using this wavelet can be computed by the convolution with two FIR filters, one with 7 and the other with 9 real-valued coefficients. Daubechies *et al.* demonstrated that the transform employing this wavelet can be computed in four successive lifting steps.

The 1-D discrete wavelet transform can be straightforwardly extended to transform 3-D signals. A separated decomposition along each axis is commonly used. The applications of 3-D DWT include video coding, medical data compression, video watermarking, medical image enhancement, or segmentation of medical volumes.

This paper presents SIMD-accelerated single-loop algorithm for 3-D discrete wavelet transform computation. Considering this proposed algorithm, every source memory element is visited only once while the resulting output coefficients are instantly produced.

This paper discusses implementation of 3-D DWT implemented over single-precision floating-point format (referred to as binary32 in IEEE 754 standard). As indicated above, the CDF 9/7 wavelet was employed. Only a single level of the transform is considered wherein subband coefficients (LLL, LLH, LHL, LHH, HLL, HLH, HHL, HHH) are kept interlaced in an output memory area. All the methods presented in this paper are evaluated using mainstream PCs with Intel x86 CPUs. The SIMD-accelerated methods was coded using the Streaming SIMD Extensions (SSE) instruction set. Intel Core2 Quad Q9000 running at 2.0 GHz and AMD Opteron 2380 running at 2.5 GHz were used in the tests below. Intel CPU has 32 KiB of level 1 data cache and 3 MiB of level 2 shared cache (two cores share one cache unit). In contrast, AMD CPU has 64 KiB of level 1 data, 512 KiB of level 2 cache per core and 6 MiB of level 3 shared cache (all four cores share one unit). All the algorithms below were implemented

 ISBN 978-80-86943-65-7

in the C language, using the SSE compiler intrinsics.[1] In all cases, a 64-bit code compiled using GCC 4.8.1 with `-O3` flag was used.

The rest of the paper is organized as follows. Section 2 discusses the state of the art. Especially, the lifting scheme and the single-loop approach is described here. Section 3 proposes a novel SIMD-accelerated single-loop algorithm for computing the discrete wavelet transform in 3-D. Finally, Section 4 summarizes the paper and outlines the future work.

2 RELATED WORK

On conventional architectures, the lifting scheme [20, 12] is the most efficient scheme for computing the discrete wavelet transform. Any discrete wavelet transform can be factorized into a sequence of N pairs of lifting steps. These steps are denoted as the predict P_n and update U_n convolution operators where each P_n corresponds to an FIR filter $p_i^{(n)}$ and each U_n to a filter $u_i^{(n)}$. The lifting factorization is not generally unique. This non-uniqueness can be exploited to maintain the symmetry of lifting steps in case of symmetric DWT filters.

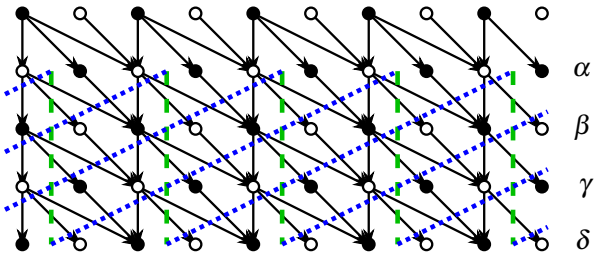


Figure 1: Lifting scheme of CDF 9/7 wavelet. Areas between dashed green lines represent the vertical and dotted blue lines the diagonal vectorization.

In [12], Daubechies *et al.* demonstrated an example of CDF 9/7 transform factorization. This lifting scheme consists of four lifting steps (two pairs). After these steps, an additional scaling of the coefficients is performed. The individual lifting steps use 2-tap symmetric filters. When evaluating this lifting scheme, some intermediate results can be appropriately shared between neighbouring coefficients. The original calculation is presented as an in-place algorithm, which means the transform might be calculated in a place of the input signal. However, this formulation is not advantageous especially due to a) a complicated border treatment (e.g., symmetric extension), b) a tendency to evicting the intermediate results from the CPU cache.

The entire calculation of CDF 9/7 DWT (without scaling) is depicted in Figure 1. In this figure, the $\alpha, \beta, \gamma, \delta$ are real constants specific to CDF 9/7 transform. Formally, the forward transform in the referred figure can be expressed by the dual polyphase matrix

$$\tilde{P}(z) = \begin{bmatrix} 1 & \alpha\left(1+z^{-1}\right) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \beta\left(1+z\right) & 1 \end{bmatrix} \begin{bmatrix} 1 & \gamma\left(1+z^{-1}\right) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \delta\left(1+z\right) & 1 \end{bmatrix} \begin{bmatrix} \zeta & 0 \\ 0 & 1/\zeta \end{bmatrix}. \quad (1)$$

The ζ is called the scaling constant.

A naive approach of the lifting scheme evaluation can directly follow the lifting steps. Using this strategy, an entire input signal is updated several times using the predict P_n and update U_n convolution operators. This method will hereinafter be referred to as a horizontal vectorization. Chrysafis *et al.* [10] addressed the problem of online (pipelined, line-based) implementations of the lifting scheme. However, their approach was very general and it is not focused on a real implementation. In contrast, Kutil [16] presented an useful implementation of CDF 9/7 lifting employing SSE instruction set. He splits the lifting scheme into vertical areas (see Figure 1). Thus, his method is below referred to as a vertical vectorization. Due to data dependencies between adjacent vertical areas, the vertical vectorization cannot be directly parallelized. For this reason, we have presented a diagonal vectorization (Figure 1) of the scheme in [1]. This vectorization allows the use of SIMD instructions directly without buffering of the coefficients into blocks.

In the image processing, a fast implementation of 2-D DWT is a widely examined area. At this place, we will briefly revise important works. Chatterjee *et al.* [7] proposed two techniques intended for an optimization of the 2-D transform. The first one interleaves the operations of several 1-D transforms on multiple columns. The second one modifies the memory layout so that each sub-band is situated there contiguously. Chaver *et al.* [8] pipelined computation in a vertical direction on subsequent rows. This is similar to pipelined strategies mentioned above in case of 1-D transform. Other similar approaches can be found in [8] and [9], where the authors vectorized a transform using an approach similar to the one described in [16]. Considering the CPU cache, the authors of [18] as well as [19] proposed several techniques for reducing cache misses.

However, the most important work was done by Kutil in [16]. The author fused vertical and horizontal 1-D filterings into a single loop (from here the single-loop approach). This was partly done in the pipelined 2-D approaches above, although not performed on the whole image at once. Let us revise the Kutil's work in more detail. The original single-loop approach is based on the vertical vectorization. One step of this vectorization requires two values to perform a single iteration. Consequently, the approach needs to perform two filterings on two consecutive rows at once. For each row, two

[1] `http://www.fit.vutbr.cz/research/prod/?id=211`

coefficients are produced, which makes four values in total. The vertical vectorization algorithm passes four values from one iteration into the other for each row. Finally, the Kutil's approach needs to pass four rows between merged iterations. Moreover, using the prime stride between subsequent rows of the image was suggested in his paper.

In [2], we have extracted the minimal core of the original single-loop approach leading to a novel single-loop core approach. This core can be built over the vertical as well as the diagonal vectorization. Several such cores can be further fused together in order to exploit a suitable SIMD instruction set. The core approach completely eliminated the problems of a complicated border treatment used by the original approach. This fact is caused by the separation of data accesses from the intrinsic transform. The influence of CPU caches on the 2-D transform was also studies in this work. In the end, we have proposed a parallelization of the SIMD-vectorized core approach that led to additional speedups.

Furthermore, many papers exist in the field of an efficient 3-D DWT implementation. Let us to mention the most significant works. In [4], Bernabe *et al.* presented two methods reducing the 3-D transform execution time. However, in both of these methods, they employed the convolution scheme that does not take advantage from benefits of the lifting scheme. In their first method, they split the original 3-D volume into several independent sub-volumes. Thus, they have performed several independent transforms[2] which is different and much easier task comparing to what we are dealing with in this paper. On the other hand, such method benefits from a small working set of the transformed data. The independent transforms can be further applied in an overlapped and non-overlapped manner. The second method is just a modification of the first one where the independent transforms are applied on cuboid subvolumes instead of cubes. The method should better exploit the memory locality occurred due to their particular memory layout. The authors also exploited a fine-grained parallelism by vectorizing loops using the SSE instructions. Unfortunately, their methods are far away from the single-loop and also the single-loop core approaches. Finally, the authors reported $5\times$ speedup compared to the non-tuned implementation. In [5], Bernabe *et al.* published a subsequent work. They exploited advantages of a parallel processing using multiple threads. The work is closely focused on hyper-threading (HT) technology. However, the principles of the methods employed remained the same as in previous paper.

[2] introducing a block effect

In [17], Lopez *et al.* introduced a fast frame-based 3-D DWT video encoder with low memory usage. The authors used the convolution scheme. In their approach, the video frames are continuously consumed by the 2-D DWT algorithm. Then, this transformed frame is stored in a buffer. Unfortunately, this buffer must be able to hold as many frames as the number of taps for the FIR filter in the temporal direction. Although their encoder reduced memory as well as computational requirements compared to the original 3D-SPIHT algorithm, it is still far away from the true 3-D pipelined transform.

In another two papers [15, 3], the authors applied separatelly 2-D spatial and 1-D temporal transform. Both of the works deal with a video compression. As in the previous case, their approaches still need several input frames to be accumulated in a buffer in order to filter the frames along the third dimension.

The implementation of 3-D DWT was also studied on modern programmable graphics cards (GPUs). For instance, the authors of [14] and [6] used the convolution scheme keeping transforms along three dimensions separated.

As it can be seen, the problem of the efficient 3-D discrete wavelet transform implementation on conventional PCs was studied to some extent. However, we see several gaps which can allow for additional speedups.

3 PROPOSED METHOD

In this section, several 3-D transform techniques are proposed. At the beginning, we discuss a problem of appropriate choice of row and slice strides (steps between consecutive rows or slices). Then, we gradually extend the single-loop core approach to three dimensions.

3.1 Naive Approaches

Several works, e.g. [16], studied a performance degradation of 2-D transform in the vertical filtering caused by poor choice of the row stride. This degradation is especially significant when the stride is a power of two. The same problem occurs in 3-D. Fortunately, the problem is eliminated by using the single-loop approach in which there is no separated horizontal and vertical filtering.

Based on our previous works [1, 2], we have implemented the 3-D transforms in two naive ways. Firstly, a volume was transformed using the horizontal vectorization separately along each dimension. This is the most naive method employing the lifting scheme. Secondly, we have implemented this transform using the vertical vectorization. This method suffers from fewer CPU cache related problems than the one employing the horizontal vectorization. However, both of them exhibit huge amount of cache misses as soon as a size of

Figure 2: Address structure in relation to the CPU cache. The sizes of the individual parts depend on the particular architecture.

the data exceeds a size of the CPU cache. In both implementations, the transform in the first dimension is computed coupled with copying the data into a destination area. This is followed by a real in-place computation for the remaining two dimensions.

The cache of the considered CPUs consists of many 64-bytes buckets called cache lines. They are divided into several sets according to a associativity of the cache (e.g., four sets for 4-way associativity). Considering such CPU cache, a virtual address is split into three parts. Low six bits specify offset in a cache line. Few upper bits specify the associativity set of the cache. The rest of bits represent a tag stored for each individual cache line. Such address structure is outlined in Figure 2. The details can be found in [13].

We have measured the performance for both of the described naive methods in combination with unchanged as well as modified strides. Note that we are talking about the row stride as well as the slice stride. The unchanged stride means that rows and slices are placed without gaps densely one by one. In relation to the second choice, we have modified the strides in the following way. Low six bits of the addresses correspond to offsets in cache lines and thus do not directly influence the selection of the cache set. These are set to zero. The other bits are changed to the next highest prime number (the "set" and "tag" in Figure 2). Although a choice of the next highest odd number is sufficient since any odd number is coprime with any power of two. The performance comparison is shown in Figure 5.

As it can be seen, the performance of unchanged stride is unstable. It exhibits poor properties especially on the power-of-two strides. However, the data has smaller memory footprint. So, if the stride length does not hit the inappropriate value, the entire transform runs faster than in the case with modified stride. In the rest of this paper, we will continue using the modified stride (i.e. prime stride).

As an intermediate step towards the full 3-D single-loop transform, we have experimented with highly optimized 2-D transforms followed by a separated filtering in the third dimension. The implementation of the optimized 2-D transform is described in our previous paper [2]. Essentially, this transform is computed "out of place" in the single loop by a single-threaded SIMD-optimized 4×4 core. Afterwards, the transform in the third dimension is computed "in place" using the vertical vectorization. For 4×4 core approach, the auxiliary buffer of a slice edge size times four has to be allocated. The buffer is used to transfer information between steps of the vertical vectorization along y axis direction. The comparison with the previous two methods is shown in Figure 6. We have plot only the implementations using the modified (stable) strides. The transform performed in the slices outperforms the two naive implementations practically for all sizes on the x axis.

3.2 True 3-D Approaches

Based on our work in [2], we have created two baseline implementations transforming the entire volume in the single loop. These implementations employing 2^3 cores – first built over the vertical and the second over the diagonal vectorization. Both of them process the data out of a place. The implementation with the diagonal core is inherently accelerated by SIMD instructions. The vertical implementation does not allow SIMD-optimizations at this stage.

In more detail, both 2^3 cores are composed from three parts. The first part loads the input data from a source memory area. The inner part performs a fragment of the transform computation. Finally, the last part stores the resulting coefficients into a destination area. The first and last parts treat data borders using the symmetric extensions.

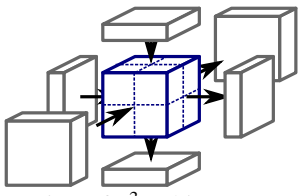


Figure 3: Illustration of 2^3 cubic core. Portions of auxiliary buffers to update are shown on each side.

Access to three auxiliary buffers is required during the inner part of the computation. See Figure 3. This most important part consists of three blocks performing calculations corresponding to the three dimensions. Each block updates the necessary intermediate results in the corresponding auxiliary buffer. This is followed by scaling of the output coefficients.

In the most generic variant, each 2-D auxiliary buffer has the same size like the corresponding volume side. The depth of each auxiliary buffer is 4 coefficients for the vertical vectorization or 12 coefficients for the diagonal one. See Figure 4. This memory consumption can be reduced using a appropriate processing order. When using the horizontal[3] order, it is not necessary to allocate the full side size buffers. It is sufficient to allocate only the following sizes. One full side size buffer for the first dimension. One edge size buffer for the second dimension. Finally, one point size buffer for the third

[3] raster scan

 ISBN 978-80-86943-65-7

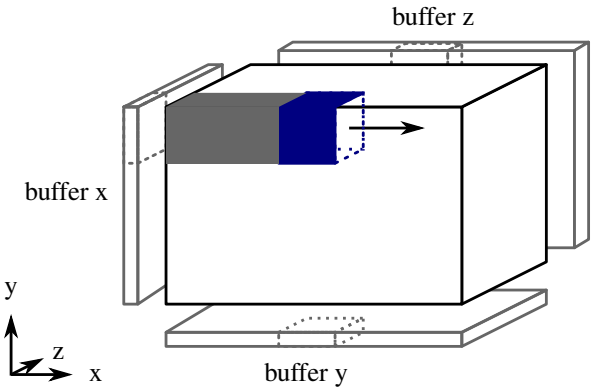


Figure 4: Complete processing of the volumetric data by the 3-D single-loop core. Auxiliary buffer for each dimension is shown on the sides of the volume.

dimension. For instance, considering the vertical vectorization and 2^3 core, it is sufficient to allocate buffers of total size $4 \cdot (s_x \cdot s_y + s_x \cdot 2 + 2 \cdot 2)$ elements, where s_x, s_y are sizes of the volume in first two dimensions.

Now, we take a closer look on the conjoined scaling. With the scaling constant ζ, the scaling of one pair of coefficients can be written as element-wise multiplication by a scaling vector

$$Z_1(x) = \begin{bmatrix} \zeta^{-1} & \zeta \end{bmatrix}, \tag{2}$$

where the coordinate $x \in \{0,1\}$.

In 2-D case, the scaling of one 2×2 block of coefficients can be written as element-wise multiplication by a scaling matrix

$$Z_2(x,y) = Z_1(x) \cdot Z_1(y), \tag{3}$$

where the coordinates $x, y \in \{0,1\}$, giving

$$Z_2(x,y) = \begin{bmatrix} \zeta^{-2} & 1 \\ 1 & \zeta^2 \end{bmatrix}. \tag{4}$$

Finally, in 3-D case, the scaling of one 2^3 box of coefficients can be written as element-wise multiplication by a scaling cube

$$Z_3(x,y,z) = Z_1(x) \cdot Z_1(y) \cdot Z_1(z), \tag{5}$$

where $x, y, z \in \{0,1\}$. For better understanding, the slices through the z axis look like

$$\begin{aligned} Z_3(x,y,0) &= \begin{bmatrix} \zeta^{-2} & 1 \\ 1 & \zeta^2 \end{bmatrix} \cdot \zeta^{-1} = \begin{bmatrix} \zeta^{-3} & \zeta^{-1} \\ \zeta^{-1} & \zeta \end{bmatrix} \\ Z_3(x,y,1) &= \begin{bmatrix} \zeta^{-2} & 1 \\ 1 & \zeta^2 \end{bmatrix} \cdot \zeta = \begin{bmatrix} \zeta^{-1} & \zeta \\ \zeta & \zeta^3 \end{bmatrix} \end{aligned}. \tag{6}$$

We have compared the performance of the two 3-D core approaches to the previous methods. The result can be seen in Figure 7. Apparently, the 3-D approaches approximately above 1 megavoxel outperform all the previous methods.

method	Intel Core2		AMD Opteron	
	time	speedup	time	speedup
naive horiz.	159.8	1.0	105.7	1.0
naive vert.	100.1	1.6	73.5	1.4
vert. slice 4^2	53.8	2.9	41.0	2.5
vertical 2^3	23.3	6.8	21.7	4.7
diagonal 2^3	22.8	6.9	21.1	4.9
vertical 4^3	**13.5**	**11.7**	**12.9**	**8.0**

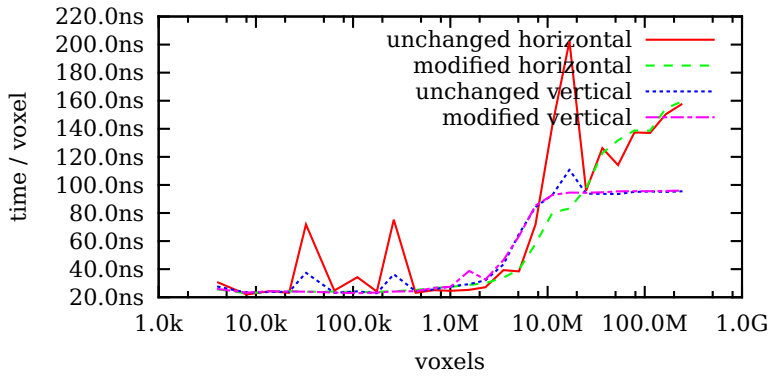
Table 1: Performance evaluation for large data. Best results are in bold.

Within the context of the vertical vectorization, utilization of SIMD instruction set is straightforward. In first two dimensions, 2×2 small 2^3 cores are merged together in analogous manner as in [2]. In the third dimension, there is no reason to increase the size of the core as SIMD can be used directly. In all three dimensions, the basic building block of the transform is 2×4 core. In this 2×4 core, four steps of the vertical vectorization are computed in parallel using SIMD instructions. The scaling of coefficients is performed together as the last step of the calculation. As a result, $4 \times 4 \times 2$ SIMD-vectorized core was formed.

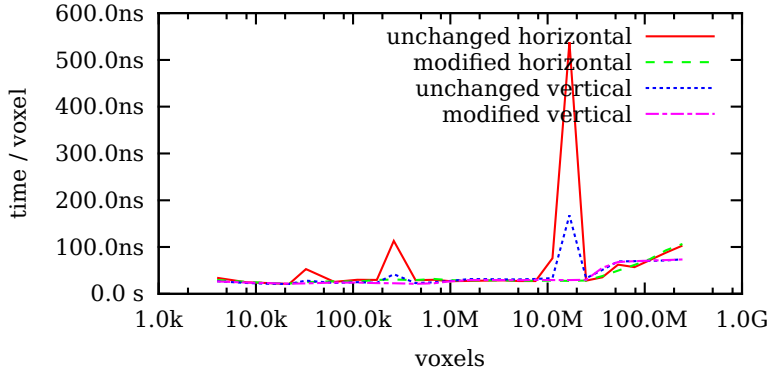
Although there is no reasonable justification, it can be tempting to build a compact 4^3 core as an analogy to the well-performing 4^2 counterpart. Such core is internally composed of two dependent $4 \times 4 \times 2$ sub-cores. However, such a connection may be slightly advantageous due to prefetching of intermediate results. Moreover, the implementation is more regular.

The final comparison is shown in Figure 8. The SIMD-optimized 3-D cores exhibit the best results. The 4^3 core slightly outperforms the baseline $4 \times 4 \times 2$ one. Above initial transients, all the single-loop approaches confirm the linear asymptotic complexity of the discrete wavelet transform.

Based on [2], the single-loop core method can be further accelerated utilizing the multi-threading. Table 1 summarizes performances and speedups for a volume of 238 megavoxels. The testing platforms are described in Section 1. The speedups are given against the separable method using the horizontal vectorization and the prime stride. The achieved processing time 13 nanoseconds per sample is roughly equivalent to process 37 frames per second with Full HD resolution (1920×1080 per frame). For $4 \times 4 \times 2$ core and any infinite video sequence, only two frames (the currently coded and the immediately preceding) have to be held in memory. The summarizing comparison of all significant approaches is shown in Figure 9.

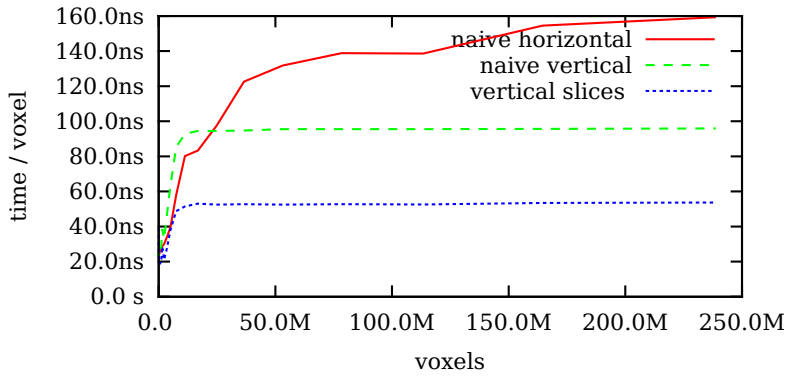


(a) Intel Core2

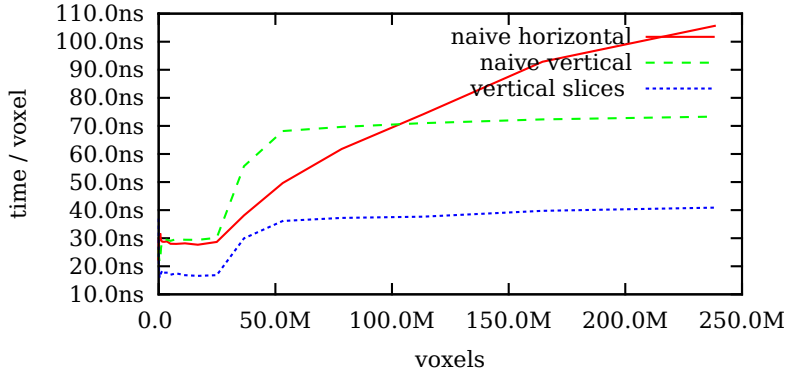


(b) AMD Opteron

Figure 5: Performance comparison of naive approaches with unchanged and prime strides.

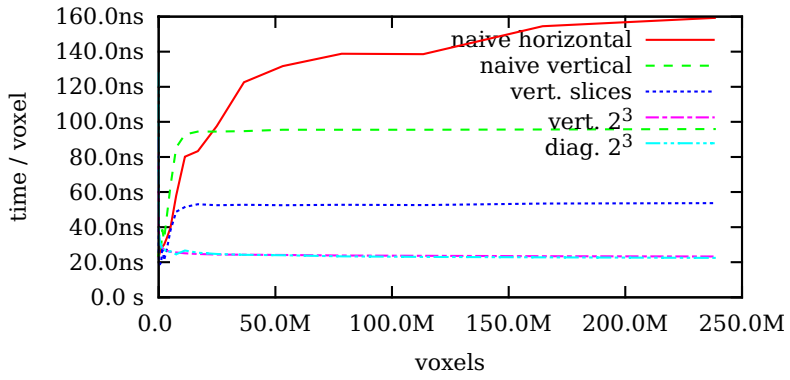


(a) Intel Core2

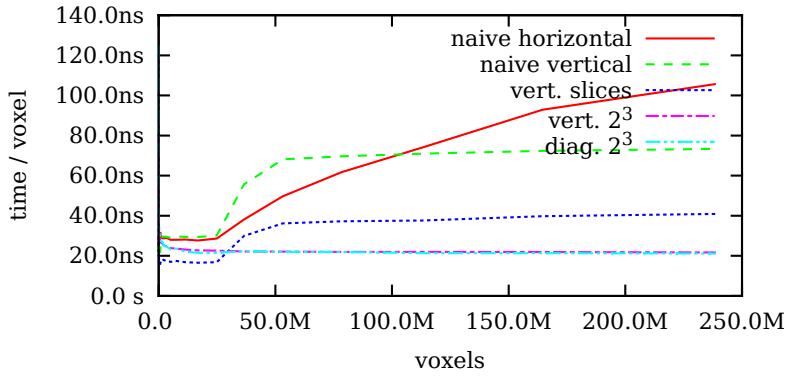


(b) AMD Opteron

Figure 6: Performance comparison of slicing 4^2 and naive approaches under the prime stride.

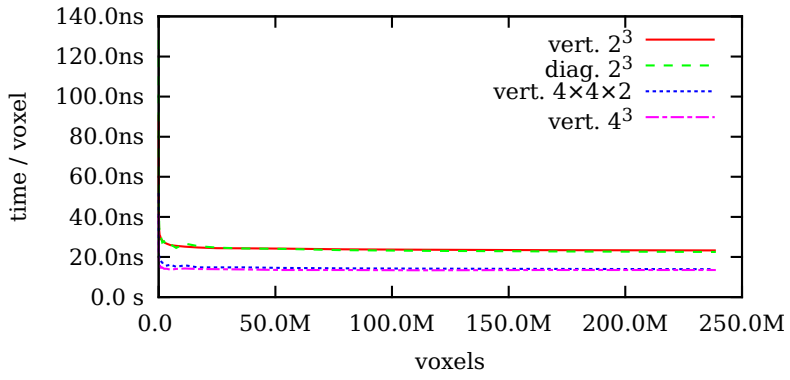


(a) Intel Core2

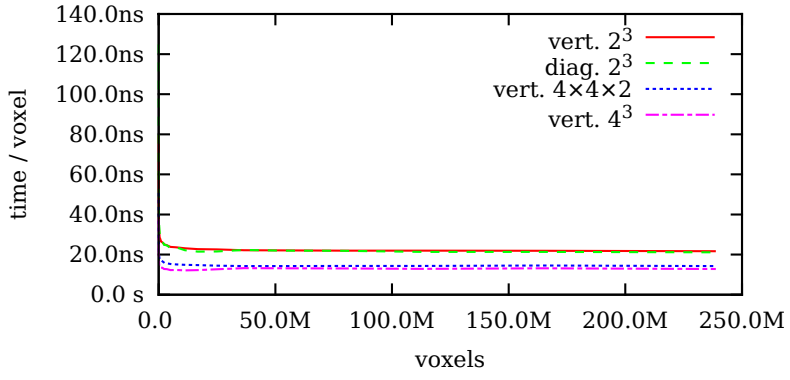


(b) AMD Opteron

Figure 7: Performance comparison of two baseline 3-D approaches with previous ones.



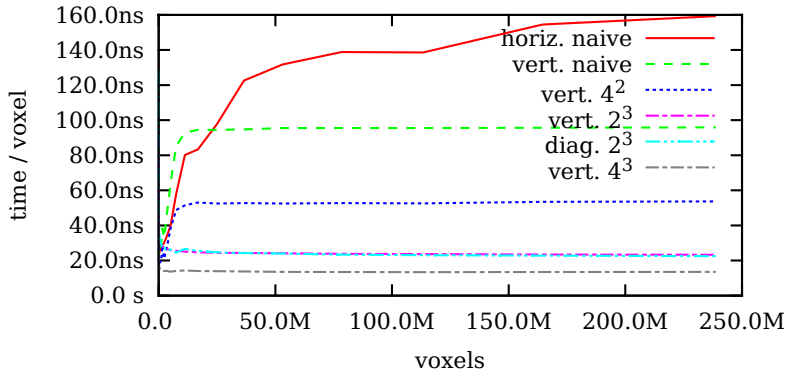
(a) Intel Core2



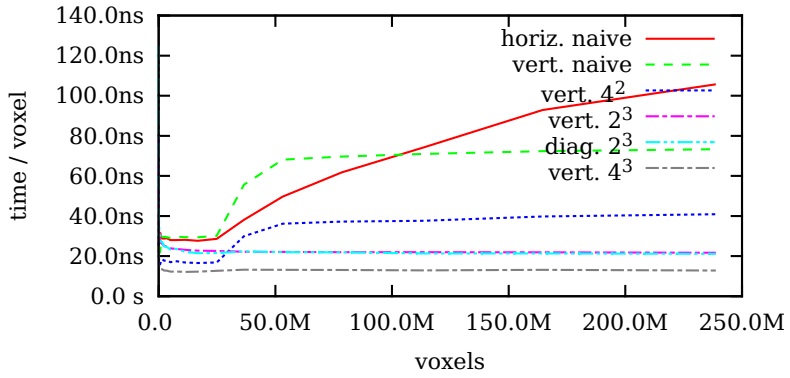
(b) AMD Opteron

Figure 8: Performance comparison of all 3-D approaches.

 ISBN 978-80-86943-65-7



(a) Intel Core2



(b) AMD Opteron

Figure 9: Summarizing performance comparison of all approaches under the prime stride.

4 CONCLUSIONS

We have presented a true single-loop approach designed to transform 3-D data. The proposed method can be understood as a streaming unit which visits every input as well as output data element only once. The core can further be improved by exploiting suitable SIMD instruction set. The best of proposed methods reaches speedup 11× on Intel and 8× on AMD compared to the naive implementation. The measurements confirm the linear theoretical complexity of the transform in 3-D. In absolute numbers, the best method can transform a data sample in 13 nanoseconds what is roughly equivalent to transform 37 frames per second in Full HD.

As a future work, our methods can be parallelized by multithreading. For real applications, it can be necessary to perform several levels of such decomposition.

Acknowledgements

This work has been supported by the IT4Innovations Centre of Excellence (no. CZ.1.05/1.1.00/02.0070).

REFERENCES

[1] D. Barina and P. Zemcik. Minimum memory vectorisation of wavelet lifting. In *Advanced Concepts for Intelligent Vision Systems*, volume 8192 of *Lecture Notes in Computer Science*, pages 91–101. Springer, 2013. ISBN 978-3-319-02894-1. doi: 10.1007/978-3-319-02895-8_9.

[2] D. Barina and P. Zemcik. Vectorization and parallelization of 2-D wavelet lifting. *Journal of Real-Time Image Processing*, 2015. ISSN 1861-8200. doi: 10.1007/s11554-015-0486-6.

[3] E. Belyaev, K. Egiazarian, and M. Gabbouj. Low complexity bit-plane entropy coding for 3-D DWT-based video compression. In *Proceedings of SPIE*, volume 8304, 2012. doi: 10.1117/12.912017.

[4] G. Bernabe, J. Garcia, and J. Gonzalez. Reducing 3D fast wavelet transform execution time using blocking and the streaming SIMD extensions. *Journal of VLSI signal processing systems for signal, image and video technology*, 41(2): 209–223, 2005. ISSN 0922-5773. doi: 10.1007/s11265-005-6651-6.

[5] G. Bernabe, R. Fernandez, J. M. Garcia, M. E. Acacio, and J. Gonzalez. An efficient implementation of a 3D wavelet transform based encoder

on hyper-threading technology. *Parallel Computing*, 33(1):54–72, 2007. ISSN 0167-8191. doi: 10.1016/j.parco.2006.11.011.

[6] G. Bernabe, G. Guerrero, and J. Fernandez. CUDA and OpenCL implementations of 3D fast wavelet transform. In *IEEE Third Latin American Symposium on Circuits and Systems (LASCAS)*, pages 1–4, Feb. 2012. doi: 10.1109/LASCAS.2012.6180318.

[7] S. Chatterjee and C. D. Brooks. Cache-efficient wavelet lifting in JPEG 2000. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, volume 1, pages 797–800, 2002. doi: 10.1109/ICME.2002.1035902.

[8] D. Chaver, C. Tenllado, L. Pinuel, M. Prieto, and F. Tirado. Wavelet transform for large scale image processing on modern microprocessors. In *High Performance Computing for Computational Science – VECPAR 2002*, volume 2565 of *Lecture Notes in Computer Science*, pages 549–562. Springer, 2003. ISBN 978-3-540-00852-1. doi: 10.1007/3-540-36569-9_37.

[9] D. Chaver, C. Tenllado, L. Pinuel, M. Prieto, and F. Tirado. Vectorization of the 2D wavelet lifting transform using SIMD extensions. In *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS)*, page 8, 2003. doi: 10.1109/IPDPS.2003.1213416.

[10] C. Chrysafis and A. Ortega. Minimum memory implementations of the lifting scheme. In *Proceedings of SPIE, Wavelet Applications in Signal and Image Processing VIII*, volume 4119 of *SPIE*, pages 313–324, 2000. doi: 10.1117/12.408615.

[11] A. Cohen, I. Daubechies, and J.-C. Feauveau. Biorthogonal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 45(5):485–560, 1992. ISSN 1097-0312. doi: 10.1002/cpa.3160450502.

[12] I. Daubechies and W. Sweldens. Factoring wavelet transforms into lifting steps. *Journal of Fourier Analysis and Applications*, 4(3):247–269, 1998. ISSN 1069-5869. doi: 10.1007/BF02476026.

[13] U. Drepper. What every programmer should know about memory, 2007. URL `http://www.akkadia.org/drepper/cpumemory.pdf`.

[14] J. Franco, G. Bernabe, J. Fernandez, and M. Ujaldon. Parallel 3D fast wavelet transform on manycore GPUs and multicore CPUs. *Procedia Computer Science*, 1(1):1101–1110, 2010. ISSN 1877-0509. doi: 10.1016/j.procs.2010.04.122. ICCS 2010.

[15] V. Galiano, O. Lopez-Granado, M. Malumbres, and H. Migallon. Multicore-based 3D-DWT video encoder. *EURASIP Journal on Advances in Signal Processing*, 2013(1):84, 2013. doi: 10.1186/1687-6180-2013-84.

[16] R. Kutil. A single-loop approach to SIMD parallelization of 2-D wavelet lifting. In *Proceedings of the 14th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, pages 413–420, 2006. ISBN 0-7695-2513-X. doi: 10.1109/PDP.2006.14.

[17] O. Lopez, M. Martinez-Rach, P. Pinol, M. Malumbres, and J. Oliver. A fast 3D-DWT video encoder with reduced memory usage suitable for IPTV. In *2010 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1337–1341, July 2010. doi: 10.1109/ICME.2010.5583570.

[18] P. Meerwald, R. Norcen, and A. Uhl. Cache issues with JPEG2000 wavelet lifting. In *Visual Communications and Image Processing (VCIP)*, volume 4671 of *SPIE*, pages 626–634, 2002.

[19] A. Shahbahrami, B. Juurlink, and S. Vassiliadis. Improving the memory behavior of vertical filtering in the discrete wavelet transform. In *Proceedings of the 3rd conference on Computing frontiers (CF)*, pages 253–260. ACM, 2006. ISBN 1-59593-302-6. doi: 10.1145/1128022.1128056.

[20] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis*, 3 (2):186–200, 1996. ISSN 1063-5203.

Improvement of a Topological-Physical Model to manage different physical simulations

K. Golec, M. Coquet, F. Zara, G. Damiand

Université de Lyon, CNRS
Université Lyon 1,
LIRIS, UMR5205,
F-69622, Villeurbanne, France

firstname.name@liris.cnrs.fr

ABSTRACT

We present an improvement of a unified topological-physical model which permits topological modifications during physical simulation of soft tissues. Our improvement makes the model more generic, efficient and simpler to update. The main principle of our improvement is to associate information to elements of the model, depending on the underlying physical model. Our modification of the architecture enables to easily integrate different physical models. Moreover, topological operations and physical simulations can be factorized between the different physical models. Our solution is more efficient as it leads to simpler modification algorithms after topological alterations with less changes to apply. In this paper, we present our new solution and illustrate its new properties thanks to several experiments performed on two well-known physical models: Mass-Spring System and Tensor-Mass model. The results present a comparison of our solution with the previous one for the cutting topological operation. Moreover, as our model permits to easily compare several physical models, we performed some simulations to reproduce experiments made on real tissues.

Keywords

Physical Simulation; Mass-Spring System; Tensor-Mass Model; Combinatorial Maps.

1 INTRODUCTION

In medical applications, the actual challenge is to propose physical models which provide interactive simulations with topological modifications (such as cutting). The goal is to simulate surgical operations for training. In computer graphics, the field of soft bodies simulation covers many methods [NMK+06]. The two main approaches, namely Finite Element Method (FEM) and Mass-Spring System (MSS), focus either on accuracy or on the performance of the system with visually satisfactory results. MSS is faster and easier to implement than FEM, but provides less accurate results. However, some papers focus to improve the MSS model by integrating some mechanical properties (such as Young's modulus and Poisson's ratio) in the computation of the springs properties [BSSH04, LSH07, BBJ+09].

Furthermore, as an alternative to the classic FEM, the Tensor-Mass (TM) approach has been introduced by Delingette, Cotin and Picinbono [DCA99] and extended by Schwartz [SDR+05]. This approach is based on another way to solve the mechanical equations of the objects, with a direct computation of the forces applied on each node considering the evaluation of the strain energy density. This approach allows to compute elements' information regardless of the neighborhood of this element.

In this context, this paper proposes an extension of the work of Flechon [FZDJ13, FZDJ14] who proposed a unified topological-physical model (called LCC+MSS) adapted for a mass-spring simulation that handles cutting or piercing simulated objects. Only one model is used to define the topology, the geometry and the physical properties of the simulated object, enabling topological modifications during its simulation by updating on the fly. In this paper, our contributions are:

- The improvement of the efficiency of the topological-physical model for cutting by changing the architecture of the model.
- The improvement of the topological-physical model to manage several physical models. We illustrate this issue by integrating the mass-spring system and tensor-mass model which both naturally allow dynamical topological modifications thanks to a local formulation of the strain.

- The use of the topological-physical model to simulate realistic and accurate experiments on soft tissues.

The paper is organized as follows. Section 2 presents the two physical models used. Section 3 presents the unified topological-physical model proposed by Flechon. Section 4 presents our extension of this unified model with discussion in Section 4.3. Section 5 presents some experiments showing that the performances of the LCC+MSS model have been improved by the new architecture of the topological-physical model. Moreover, Section 6 shows that our model can be used to conduce real experiments made on soft tissues and to compare physical models.

2 PHYSICAL MODELS

Mass-Spring System. In computer graphics, mass-spring systems have largely been used to simulate the behavior of deformable objects [TPF91, BFA02]. This physical model is based on the discretization of the object into a set of masses (also called particles) interconnected by springs. Then at each time step, the following main steps are performed for each particle:

(1) Computation of the forces applied on the particle due to springs and external forces. Remember that the force involved by a spring connecting particles i and j, with stiffness constant k_{ij} and initial length l_{ij}, is defined at time t by $F_{ij}(t) = F^e_{ij}(t) + F^v_{ij}(t)$.

- $F^e_{ij}(t)$ is the elasticity force of this spring with

$$\begin{cases} F^e_{ij}(t) &= k_{ij}\left(\|P_j(t) - P_i(t)\| - l_{ij}\right) u_{ij}(t), \\ F^e_{ji}(t) &= -F^e_{ij}(t), \end{cases}$$

  where $P_i(t), P_j(t)$ are the positions of i, j and $u_{ij}(t) = (P_j(t) - P_i(t))/\|P_j(t) - P_i(t)\|$.

- $F^v_{ij}(t)$ is the viscosity force of this spring with

$$F^v_{ij}(t) = \gamma_{ij}\left(V_j(t) - V_i(t)\right) \cdot u_{ij}(t)\, u_{ij}(t),$$

  where $V_i(t), V_j(t)$ are the velocities of i, j and $\gamma_{ij} = 2\sqrt{k_{ij}(m_i + m_j)/2}$ the spring's viscosity coefficient where m_i, m_j are the masses of i, j.

(2) Computation of the acceleration according to Newton's second law with $\frac{d^2}{dt^2}P_i(t) = F_i(t)/m_i$ where $F_i(t)$ represents the forces applied on particle i.

(3) Computation of the velocity and position according to the acceleration using a numerical integration scheme (for instance the semi-implicit Euler one).

Tensor-Mass model. The TM approach is based on the discretization of the object into several elements as for the FEM, but then the equations are solved locally, making this approach more suitable for topological modifications of the object. To account for various mechanical behaviours, several formulations have been presented: linear Hooke's model [CDA00], non-linear geometrical model based on Saint Venant-Kirchhoff's elasticity model [PDA00], anisotropic material [Pic03] and non-linear visco-elastic deformations [SDR$^+$05] with some pre-computations to accelerate the process.

The simulation's loop of the MT approach involves the following main steps for each element E of the object:

(1) Computation of the displacement of a point X inside E with

$$U_E(X) \simeq \sum_{j=0}^{n-1} \Lambda_j(X)\, U_j,$$

where n is the number of 3D nodes P_j defining the element, U_j the displacement of each node P_j from its initial position, and $\Lambda_j(X)$ some interpolation functions defined according to the type of element used for the discretization.

(2) Computation of the deformation energy according to the displacement of the element's nodes. In our work, we consider a non-linear elasticity behavior using the Green-Saint Venant strain-tensor

$$\varepsilon(X) = \frac{1}{2}\left(\nabla U^T(X) + \nabla U(X) + \nabla U^T(X)\, \nabla U(X)\right).$$

The associated strain energy density is defined by

$$W(X) = \frac{\lambda}{2}\left(\text{tr}\ \varepsilon(X)\right)^2 + \mu\ \text{tr}\ \varepsilon(X)^2,$$

where λ and μ are the Lamé coefficients characterizing material stiffness.

(3) Computation of the elasticity force on any node $P_j \in E$ for $j \in [0, n-1]$ with

$$F_E(P_j) = -\frac{\partial W_E(P_j)}{\partial U_j},$$

where $W_E(P_j)$ is the energy density of deformation of the considered element evaluated at node P_j.

(4) Computation of the acceleration according to Newton's second law (in the same way that for MSS).

(5) Computation of the deformation and displacement of the object using a numerical integration scheme (in the same way that for MSS).

In this paper, we used the approach proposed by Faure [FZJM12]. The formulation of the forces applied on each node of the elements was generated thanks to symbolic computation. Thus, the steps (1), (2) and (3) of the simulation loop were replaced by the direct formulation of the force applied on the nodes.

3 LCC+MSS MODEL

3D Linear Cell Complex. The unified topological-physical model (called LCC+MSS) proposed by Flechon [FZDJ13, FZDJ14] used as underlying data structure the 3D Linear Cell Complex (LCC) [Dam12] from the CGAL Open Source geometric algorithms library [The12]. With this topological structure, an orientable object is represented using 3D combinatorial maps (called 3-maps) [Lie94, DL14] as a subdivision of 0-cells (vertices), 1-cells (edges), 2-cells (faces) and 3-cells (volumes). The cells are described by darts, a generalization of half-edges [Män87] to higher dimension, plus pointers between these darts called β_i with $i \in \{0,1,2,3\}$. The 3D linear cell complex data structure is then obtained by associating to 0-cells the coordinates in $\mathbb{R}^3$ of corresponding points.

Fig. 1 represents two adjacent cubes described by a 3-map. Darts are displayed as arrows. $\beta_1(d)$ is the next dart following dart d in the same face and the same volume. $\beta_0(d)$ is the previous dart in the same face and volume. $\beta_2(d)$ gives the other dart from d belonging to the same edge, the same volume but not the same face. $\beta_3(d)$ gives the other dart from d belonging to the same edge, the same face but not the same volume.

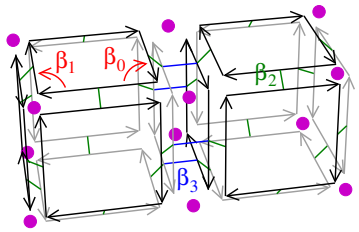


Figure 1: Two adjacent cubes described by a 3-map. Each cube has 6×4 darts. Purple disks represent vertices associated with points to obtain a 3D LCC.

This representation using darts and pointers describes the cells of an object as well as all the incidence and adjacency relationships. If two i-cells share a common $(i-1)$-cell, they are adjacent (for example two volumes sharing a common face). If one cell belongs to the boundary of another cell, these two cells are incident (for example a vertex belonging to the boundary of a face). This information is very useful to allow topological modifications during the simulation.

Lastly, any information can be associated with cells thanks to the *attribute notion* (*e.g.* a length to edges or a color to faces). We denote by *i-attribute(d)* the attribute associated with the i-cell containing dart d.

3D LCC for MSS. Flechon added information to a 3D LCC to construct the LCC+MSS model which is suitable for a MSS based on the Baudet formulation for hexahedral mesh [BBJ+09]. This formulation is available for basic hexahedra including four inner springs.

In the LCC+MSS model, a data structure `Particle` is associated to each 0-cell thanks to a 0-attribute and a data structure `Spring` is associated to each 1-cell thanks to a 1-attribute. Four additional springs inside each hexahedron are also described by the `Spring` data structure. But contrary to particles and springs previously described, these inner springs are not associated with any cells of the 3-map, but directly associated to the object.

- The `Particle` structure stores information related to a particle: its mass, velocity, acceleration and the sum of the forces applied to it. Moreover, it stores the list of the inner springs attached to it.
- The `Spring` structure stores the physical properties of a spring: its initial length, its stiffness and its two extremities (thanks to two pointers to the two `Particle` connected by the considered spring).

Fig. 2 illustrates an example of a LCC+MSS model for two adjacent cubes.

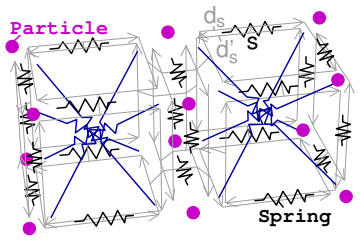


Figure 2: Two adjacent cubes described by the LCC-MSS model which is a 3D LCC associated with particles (purple disks), springs (in black) and inner springs (in blue).

Interest of using topological model for physical simulation. The classical data-structure used in MSS simulations is a graph where vertices correspond to particles and edges correspond to springs. This simple model allows to directly and easily compute the forces resulting from the springs and to accumulate these forces on the particles. However, the treatment becomes more difficult for topological modifications. Indeed, it is not possible with such a graph, to directly retrieve the elements of the object, nor all the neighborhood relations between the elements, the springs and the particles when these relations are necessary to implement topological modifications.

For example, to cut an object between two adjacent hexahedra (*i.e.* two hexahedra sharing a face), particles of the face could need to be duplicated (as we will see later with Fig. 4). To detect if a particle needs to be duplicated or not, we need to search for existence of a path of adjacent elements joining the two cut hexahedra. This is not possible directly in the graph. Thus, a data-structure has to be added to represent elements by

storing indices of its particles and indices of its adjacent elements. Then, we also need to add to each particle the indices of its incident elements. These additional data-structure and links need to be initialized and updated coherently after each operation. Moreover, other data-structures and links will be required to implement another topological modification.

The main interest of using a topological model is that we are sure that all the topological relations are described and can be retrieved directly. Existing topological constraints ensure the topological validity of the described objects and can be used to test the consistency of implemented operations. Moreover, we can use many existing and proved topological operations that can serve as basic tool to implement high level operation in our physical simulation.

4 OUR NEW SOLUTION

In this paper, we propose an improvement of the LCC+MSS model. At this time, we consider only hexahedral elements due to the physical models used, but the structure is suitable for any topology. Our model follows two new principles: (1) to store the darts of each element in 3-cells of the 3-map; (2) to associate the physical information directly in the corresponding cells of the 3-map.

4.1 Storing of the darts in their element

For the first principle, we add in each element an array called `dart` containing all the darts of the element grouped by faces *i.e.* an array `dart[6][4]` is associated to each hexahedron. The initialization of these arrays is performed at the construction of the 3-map using the convention given in Fig. 3. With this structure, all points of a given hexahedron can be retrieved thanks to the stored darts. For example, point X_0 is obtained from `dart[0][3]` and point X_1 is obtained from `dart[4][2]`. Moreover, each point of an hexahedron can be retrieved by three different darts. For example X_0 can also be retrieved thanks to `dart[1][0]` and `dart[4][1]`.

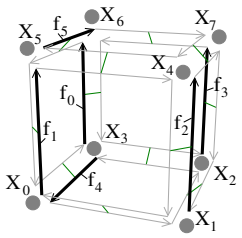


Figure 3: Convention used to store darts of a hexahedron composed of points X_i with $i \in \{0,\dots,7\}$. $f_j =$ `dart[j][0]` with $j \in \{0,\dots,5\}$ correspond to the first darts of each face (the bolded arrows).

One major interest of our proposal is to provide a direct access to all the incidence relations between cells, allowing a direct access in $O(1)$ to information required by the different physical models. This direct access is done either by using the relations given by the 3-map or by the relations given by the `dart` array.

4.2 Storing of physical information in the corresponding cells

The second principle is based on the fact that the physical information associated to the *i*-cells (with $i \in \{0,1,2,3\}$) of the 3-map is depending on the physical model chosen. Consequently, the direct association of the physical information to the corresponding cells allows to consider easily different physical models. In this subsection, we present the use of our proposal for a mass-spring system and a tensor-mass model, but any physical model can be envisaged. We terminate this subsection by showing the generic computation of the forces accumulation thanks to this second principle. Naturally, the computation of the acceleration of the object and its integration in order to obtain its velocity and its position is the same for all physical objects who are based on the newtonian dynamics.

Mass-Spring System. For a mass-spring system based on the Baudet formulation [BBJ+09], particles are associated to 0-cells thanks to 0-attributes, springs are associated to 1-cells thanks to 1-attributes, and inner springs are associated to 3-cells thanks to 3-attributes. Indeed, we remember that in the considered MSS, each hexahedron includes 4 inner springs s_i with $i \in \{0,1,2,3\}$. By convention, we fix spring s_0 (*resp.* s_1, s_2, s_3) between particles (P_0,P_7) (*resp.* (P_0,P_7), (P_1,P_6), (P_2,P_5) and (P_3,P_4)) where particles P_i correspond to point X_i with $i \in \{0,\dots,7\}$ presented in Fig. 3.

To compute the forces involved by springs (supported both by 1-cells and 3-cells), each spring needs to know its two extremity particles which are the parameters of the `addForce()` function used for this purpose. This can be retrieved thanks to the 3-map and to the `dart` array, without storing explicitly the link between springs and particles.

- For each spring s associated to a 1-cell, one dart d_s of the edge is directly known thanks to the 3-map (see Fig. 2). The first particle extremity of the spring is given by 0-attribute(d_s). The second particle extremity of the spring is given by 0-attribute(d'_s) with $d'_s =$`other_extremity(`d_s`)` the dart associated with the second extremity of the edge.
- For any inner spring s_i associated to a 3-cell, its two particles extremity are directly retrieved by our convention and the `dart` array. For example for s_0, its first extremity is 0-attribute(`dart[1][0]`) and its second extremity is 0-attribute(`dart[3][1]`).

 ISBN 978-80-86943-65-7

Tensor-Mass model. For the tensor-mass model, nodes are associated to 0-cells thanks to 0-attributes, and the formulation of the forces applied on each node of an element are associated to 3-cells thanks to 3-attributes. In this paper, we only consider hexahedral mesh with a non-linear behavior using the Green-Saint Venant strain-tensor, but it could easily be extended to any topology or physical behavior thanks to the formulation of the forces proposed by Faure [FZJM12].

To compute the forces applied on each element inside the `addForce()` function, each element starts by computing the displacement of their nodes using their current and initial position. Thanks to the `dart` array, each node of a 3-cell is retrieved in constant time. Then, the force applied on each node of an element is directly computed using the formulation of forces generated by symbolic computation.

Generic forces accumulation. Thanks to this new way to associate physical information to a LCC, the forces applied on the object are easily computed by performing a loop on all the relevant i-cells, that is the i-cells affected by the force computation (*e.g.* the 1-cells and 3-cells for MSS; the 3-cells for MT) and calling the appropriate `addForce()` function on the associated physical information. In practice, it consists to iterate through all the enabled i-attributes (*i.e.* the attributes which are associated to cells of the 3-map).

4.3 Comparison with previous solution

As recalled in Section 3, the initial definition of the LCC+MSS topological-physical model [FZDJ13] associates physical entities to cells (similarly than in our new solution), except for inner springs which are not associated with elements. Moreover, each spring stores its two extremity particles to be able to accumulate the force of this spring to the particles.

Improvement for cutting. With this previous solution, extremity particles of springs may have to be updated after a cutting. For example in Fig. 4, the cut of the two top hexahedra leads to the split of particles v_1 and v_2 into four particles v_1, v_1', v_2, v_2'. But after this cutting, springs of the top-right hexahedra are still associated with original particles v_1 and v_2 instead of particles v_1' and v_2' (see Fig. 4(a)). To solve this problem after the cutting, a post-processing is applied to update the incorrect extremities of springs (see Fig. 4(b)).

The solution proposed [FZDJ13] is to update all the springs having as extremity the particles involved in the cutting. This requires to store in each particle the list of the springs touching it. Then, when a particle is duplicated due to a cutting, an iteration is performed through this list to update the extremities of the springs to point to the new particles. Moreover, the list of springs is split in two parts: the first part for springs incident to v_1 and the second one for springs incident to v_1' (the new particle). This treatment requires a complex and time consuming processing.

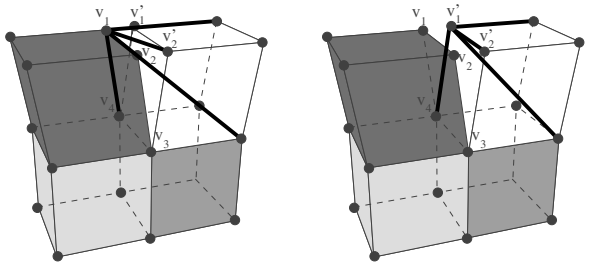


Figure 4: In the LCC+MSS model (pictures from [FZDJ13]), the cutting involves the updating of the spring extremities. (a) The bold segments represent the inner springs wrongly attached to particle v_1 after the cutting. (b) The same springs updated after a post-processing step.

With our new solution, as the darts of an element are constant whatever the topological modification applied on its neighbor elements, no change is required for cutting. For example, the darts of the two hexahedra split in Fig. 4 are still valid after the cutting. Moreover, as springs and inner springs extremities are only defined by using the darts of the hexahedra, they are still valid after the cutting without any updating. Thus our algorithms with topological modifications are simpler than Flechon and our method is faster (as shown in our experiments in Section 5) than Flechon.

Improvement for any physical models. Another advantage of our proposal concerns the ease to consider several physical models as presented in previous subsection. Indeed, as physical information are associated with cells thanks to the 3-map, the same algorithm is used to accumulate forces regardless of the physical model used (in our actual implementation, for both mass-spring system and tensor mass model).

Moreover, the 3-map and the `dart` array are up to date after cutting without any need of post-processing. Thus, only physical properties concerning by the cutting (like the mass) have to be updated. This simplifies the integration of new physical models in our system.

5 RESULTS

In this section, we present results obtained in a Intel®i5 4690 CPU, 4 cores @3.50 GHz with 16 Go RAM. For the following tests, we consider a gravity put to $g = -9.8$ m.s^{-2} and beams with a density $\rho = 1000$ Kg.m^{-3}, a Young's modulus $E = 400$ KPa and a Poisson's ratio $\nu = 0.3$.

Time for cutting. We start by a comparison in time for cutting between our approach and that of

Flechon [FZDJ13]. In Fig. 5, the curves represent the time in secondes involved to cut a beam compared to the number of faces disconnected. We consider beams discretized in $200 \times 200 \times 2$, $400 \times 400 \times 2$, $600 \times 600 \times 2$ and $800 \times 800 \times 2$ elements of size $1 \times 1 \times 1$ m with a cutting according to the Z axis.

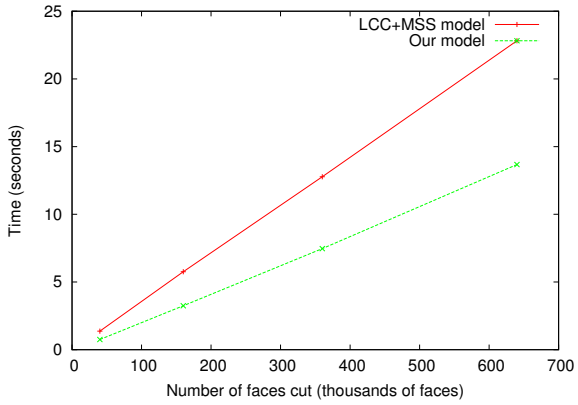


Figure 5: Comparison for the time involved for cutting.

In Table 1, more complex meshes are used for the comparison in time (in ms) for cutting: a frog (2,354 elements), an elephant (125,754 elements), a mushroom (27,000 elements) and a giraffe (85,405 elements).

Mesh	# faces cut	Flechon	Our model
Frog	112	4.0	2.0
Elephant	592	14.5	7.6
Mushroom	987	32.2	17.7
Giraffe	5210	176.3	91.6

Table 1: Comparison of time (in ms) for cutting.

We observe that our method is nearby twice as fast than the LCC+MSS model as less information has to be updated in our approach to take into account a topological modification of the object.

Precision. Fig. 6 presents a comparison for the traction test between the results obtained using the MSS and the MT approach compared to the analytic solution with beams discretized in $5 \times 5 \times 20$ and $10 \times 10 \times 40$ elements. We consider the analytic solution given by

$$y = \frac{\rho g l h}{24\,E\,I}(4Lx^3 - 6L^2x^2 - x^4)$$

with $L = 1$ m the length, $h = 0.25$ m the height, $l = 0.25$ m the thickness of the beams and $I = lh^3/12$ the inertia moment. Fig. 7 presents a visual comparison of the equilibrium state obtained with the $10 \times 10 \times 40$ MSS beam and the $10 \times 10 \times 40$ MT beam.

As expected, we observe that the MT simulation is closest to the analytic solution than MSS simulation. This experiment shows the interest of having several physical models in a same software to simply comparisons between different simulations.

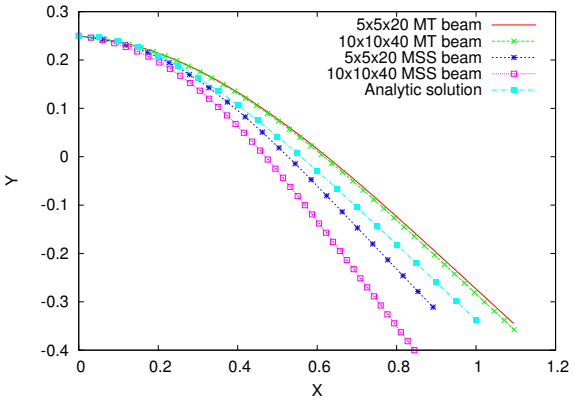


Figure 6: Comparison between the MSS, the MT and the analytic solution for a traction test.

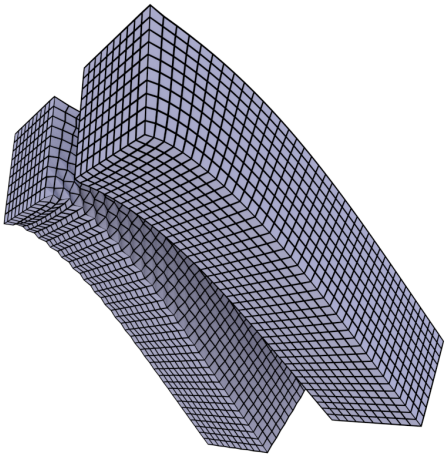


Figure 7: Comparison between the MSS (left) and the MT (right) for a traction test performed on beams discretized in $10 \times 10 \times 40$ elements.

Time of execution. Fig. 8 presents a comparison in time for the traction test performed on beams with several discretization. The aim was to vary the number of elements of size $0.025 \times 0.025 \times 0.025$ m. We present sequential and parallel time with a parallelization of the simulation's loop performed using the Intel®TBB (Threading Building Blocks) C++ template library for task parallelism. Four threads were involved for the parallel time. We obtained an average speedup of 1.45 for the MSS and 2.73 for the MT.

The MT simulation is slower than the MSS simulation due to the more complex physical equations. However its parallelization has a better speedup than the MSS one. This is due to the fact that MT uses only once type of attribute for the physical information (associated with 3-cells), while MSS uses two different types of attributes (1-cells for springs and 3-cells for inner springs) requiring two different loops.

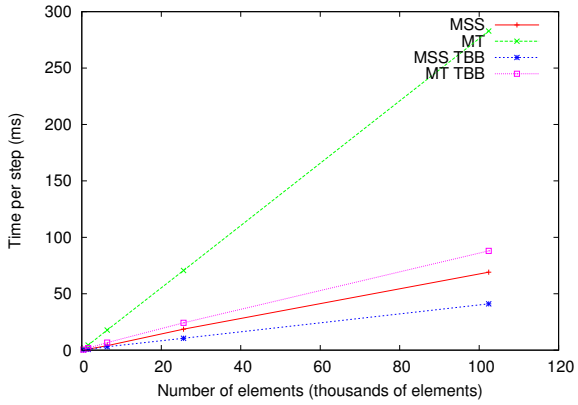


Figure 8: Comparison in time for the MSS and MT simulation performed in sequential or in parallel with TBB.

Illustrations of cutting. Fig. 9 and 10 show cutting made during a MT or a MSS simulation of meshes subject to the gravity. These pictures show that our model allows topological modifications of the objects during their physical simulation.

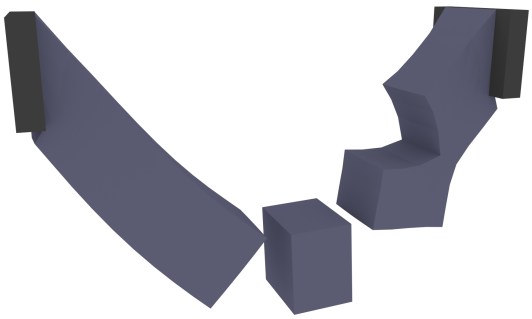


Figure 9: Cutting made during a MT simulation of a beam (subject to gravity) fixed to its two extremities.

Figure 10: Cutting made during a MSS simulation. Some particles were fixed and the gravity was applied. The density of the letters was put to $\rho = 200$ Kg.m^{-3}.

Fig. 11 shows several cuts applied on a giraffe mesh (with no simulation) that contains 85,405 elements. The different parts were separated by hand to obtain a proper visualization.

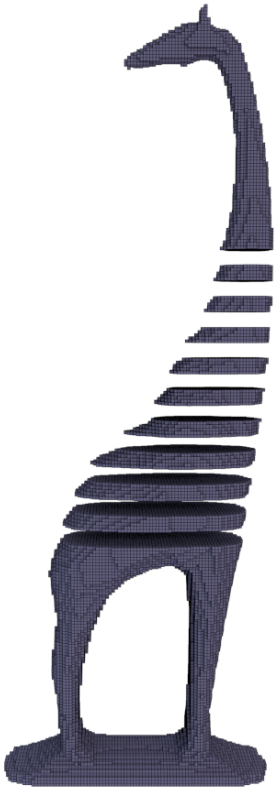


Figure 11: Ten successive cuts on a giraffe mesh.

6 REPRODUCE REAL EXPERIMENTS

In this section, we present the use of our model to simulate experiments performed on real tissues. Firstly, we explain the original experiments made on porcine tissues [NNP12]. Secondly we present our simulation and the preliminary results obtained with the MSS and the MT model. Note that the comparison between two physical models in this simulation is possible thanks to our improvement of the physical-topological model.

Experiment. The original experiments were performed on a porcine liver disks of $2-3.7$ mm thickness and diameter of approximately 15 mm. Fig. 12 presents a schematic picture of the experiment. The disks of porcine tissues were glued between the plates of a rotational rheometer. Then, the top plate was rotating at a frequency varied from 0.1 Hz to 2 Hz, while the bottom one was constrained. This experiment is equivalent to a local deformation of the porcine disks.

Figure 12: Schematic representation of the experiments made on porcine tissues [NNP12].

 ISBN 978-80-86943-65-7

The authors [NNP12] of the original experiments derived a non-linear model from the results obtained with the experiments. However, since we only consider the linear case, the following equations are modified accordingly.

The model is represented by the relaxation modulus $G(t)$ and the stress-strain relationship $\sigma[\varepsilon(t)]$. We have:

$$G(t) = \frac{K}{\Gamma(1-n)} t^{-n}$$

with K the consistency, n the linear constitutive index and t the time [KM09, NVP10]. The Gamma function is defined by

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt.$$

Considering the experiments made on liver tissues, we used as values $K = 861$ Pa.s^n and $n = 0.1138$ [NNP12]. The stress-strain relationship is defined for the linear case by

$$\sigma[\varepsilon(t)] = \int_0^t G(t-t')\dot{\varepsilon}(t')dt',$$

where σ represents the stress, $\varepsilon(t)$ the strain and $\dot{\varepsilon}$ a constant strain rate. We considered the same three strain rates as the authors [NNP12] with $\dot{\varepsilon} \in \{0.0151\ s^{-1}, 0.133\ s^{-1}, 0.67\ s^{-1}\}$.

Simulation. We used our model to reproduce the experiments by simulation. We assumed that locally the tissue is represented by a rectangular block of infinitely long plates. The bottom of the tissue did not move and the top of the tissue only moved on the abscissa X. For this, we applied a direct displacement to the top particles and studied the response of the tissue. The simulated tissue consists of 3,751 particles with $10 \times 10 \times 30$ volumes (see Fig. 13). The unit of a cube is 1 mm.

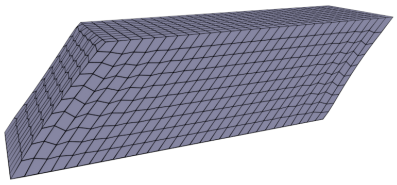


Figure 13: MSS composed of $10 \times 10 \times 30$ hexahedra to reproduce the experiments made on tissues.

As in the LCC+MSS model of Flechon, the formulation of Baudet [BBJ+09] is used to compute the stiffness constant according to Poisson's ratio and Young's modulus to integrate physical parameters into the MSS.

Results. We considered porcine liver which is a non-compressible tissue with $E = 1000$ Pa, $\nu = 0.499$ and $\rho = 1000$ Kg/m^3. The gravity force is set to zero as it is negligible in the real experiment. As we performed a shear experiment in the linear case, the strain is small enough to be negligible. For this reason, we consider the stress-time relationship to analyse our simulations.

Fig. 14 shows the stress according the time obtained for the MT model and the MSS model. For the simulations, the stress is computed by dividing the force applied on the object by the area of the top plate of the tested tissue (in our case 300 mm^2).

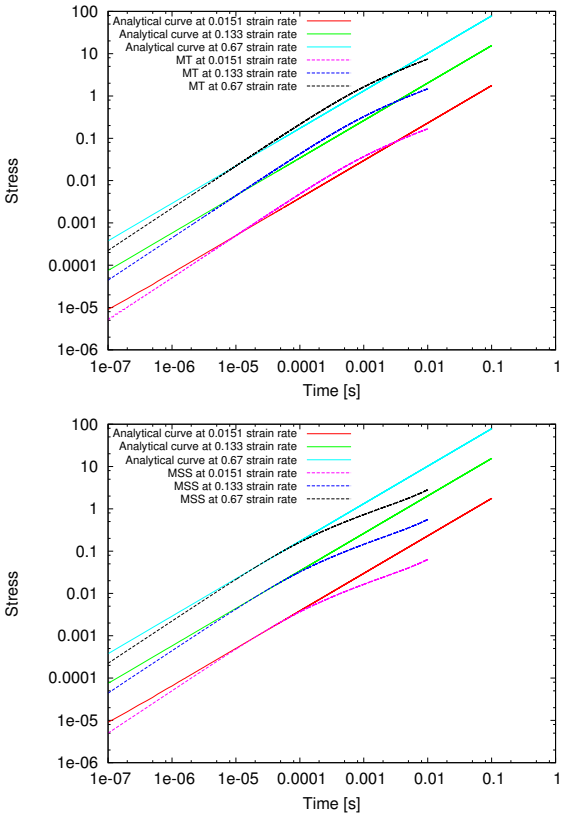


Figure 14: Evolution of stress in time for respectively the MT model (top) and the MSS model (bottom) for three different strain rates.

As expected, we observe that MT and MSS have a linear behavior for small deformations.

In the future work, we will improve the MSS solutions by implementing volume preservation and applying some correction forces. We also want to implement a new damping solution to ensure the correctness of the model for linear case.

7 CONCLUSION AND PERSPECTIVE

In this paper, we presented an improvement of the LCC+MSS model proposed by Flechon [FZDJ13]. Our improvement is based on two new principles: (1) to store the darts of each element in the 3-cells of the 3-map; (2) to associate the physical information directly in the corresponding cells of the 3-map.

Thanks to these two new principles, our model allows simpler algorithms since no more modifications are required for topological modification operations. Secondly, our model gives better results since less operations are required. Lastly, our model is more generic since it is now easier to add a new physical model while keeping all the existing operations. We have illustrated these improvements in our experiments with better results obtained for cutting than the previous model. Moreover, two different physical models (namely the mass-spring system and the tensor-mass model) have been presented who are both based on the same unified topological model. With these two physical models, we have illustrated the generic computation of the accumulation of the forces thanks to our two new principles.

In future work, we are considering adding others physical models to easily compare the results of simulations. The most interesting would be to implement the Finite Element Method and give thought to the Position Based Dynamics approach, like the one of Müller [MHHR07]. According to the MSS, we plan to include the volume preservation constraint in order to improve its precision. We plan to study the work of Aubert [AB97] which created a space deformation model (called DOGME), to deal with the volume preservation, and the more recent article of Duan [DHC+14] who treats about preserving volume by position corrections. Furthermore, to improve the precision of our simulations, it is also important to take a deeper look into constraining the particles to distribute correctly the acting forces. Lastly, we want to extend our simulation of soft tissues to the case of non-linear mechanical behavior to consider bigger deformations. All these future work will benefit of our new solution of the LCC+MSS model allowing to define simpler and more efficient algorithms to perform topological modifications during simulation for any kind of physical objects.

Acknowledgments

This work was supported by the LABEX PRIMES (ANR-11-LABX-0063) of Université de Lyon, within the program "Investissements d'Avenir" (ANR-11-IDEX-0007) operated by the French National Research Agency (ANR). The authors would like to thank the following people for their support: E. Flechon for providing the original version of TopoSim (software implementing the LCC+MSS model), X. Faure and F. Jaillet for providing the code to compute the forces generated in the mass-tensor model.

8 REFERENCES

[AB97] F. Aubert and D. Bechmann. Volume-preserving Space Deformation. *Computers & Graphics*, 21(5):625 – 639, 1997.

[BBJ+09] V. Baudet, M. Beuve, F. Jaillet, B. Shariat, and F. Zara. Integrating Tensile Parameters in Hexahedral Mass-Spring System for Simulation. In *WSCG'2009*, 2009.

[BFA02] R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph.*, 21(3):594–603, July 2002.

[BSSH04] G. Bianchi, B. Solenthaler, G. Székely, and M. Harders. Simultaneous Topology and Stiffness Identification for Mass-Spring Models based on FEM Reference Deformations. In *MICCAI 2004*, volume 2, pages 293–301. Springer, November 2004.

[CDA00] S. Cotin, H. Delingette, and N. Ayache. A Hybrid Elastic Model allowing Real-Time Cutting, Deformations and Force-Feedback for Surgery Training and Simulation. *Visual Computer*, 16(8):437–452, 2000.

[Dam12] G. Damiand. Linear cell complex. In *CGAL User and Reference Manual*. CGAL Editorial Board, 4.0 edition, 2012.

[DCA99] H. Delingette, S. Cotin, and N. Ayache. A Hybrid Elastic Model Allowing Real-Time Cutting Deformations and Force Feedback for Surgery Training and Simulation. In *Computer Animation'99*, pages 70–81, États-Unis, 1999.

[DHC+14] Y. Duan, W. Huang, H. Chang, W. Chen, J. Zhou, S. K. Teo, Y. Su, C. Chui, and S. Chang. Volume Preserved Mass-spring Model with Novel Constraints for Soft Tissue Deformation. *IEEE journal of biomedical and health informatics*, 2194(c):1–12, November 2014.

[DL14] G. Damiand and P. Lienhardt. *Combinatorial Maps: Efficient Data Structures for Computer Graphics and Image Processing*. A K Peters/CRC Press, September 2014.

[FZDJ13] E. Flechon, F. Zara, G. Damiand, and F. Jaillet. A generic topological framework for physical simulation. In *WSCG'2013*, pages 104–113, June 2013.

[FZDJ14] E. Flechon, F. Zara, G. Damiand, and F. Jaillet. A unified topological-physical model for adaptive refinement . In *VRIPHYS'2014*, pages 39–48, September

2014.

[FZJM12] X. Faure, F. Zara, F. Jaillet, and J.-M. Moreau. An Implicit Tensor-Mass solver on the GPU for soft bodies simulation. In *VRIPHYS'2012*, pages 1–10, December 2012.

[KM09] J. F. Kelly and R. J. McGough. Fractal ladder models and power law wave equations. *The Journal of the Acoustical Society of America*, 126(4):2072–81, October 2009.

[Lie94] P. Lienhardt. N-dimensional generalized combinatorial maps and cellular quasi-manifolds. *Int. J. Comput. Geometry Appl.*, 4(3):275–324, 1994.

[LSH07] B. A. Lloyd, G. Székely, and M. Harders. Identification of spring parameters for deformable object simulation. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):1081–1093, 2007.

[Män87] M. Mäntylä. *An Introduction to Solid Modeling*. Computer Science Press, Inc., New York, NY, USA, 1987.

[MHHR07] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *J. Vis. Comun. Image Represent.*, 18(2):109–118, April 2007.

[NMK+06] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson. Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum*, 25(4):809–836, December 2006.

[NNP12] S. Nicolle, L. Noguer, and J.-F. Palierne. Shear mechanical properties of the spleen: experiment and analytical modelling. *Journal of the mechanical behavior of biomedical materials*, 9:130–6, May 2012.

[NVP10] S Nicolle, P Vezin, and J-F Palierne. A strain-hardening bi-power law for the nonlinear behaviour of biological soft tissues. *Journal of biomechanics*, 43(5):927–32, March 2010.

[PDA00] G. Picinbono, H. Delingette, and N. Ayache. Real-Time Large Displacement Elasticity for Surgery Simulation: Nonlinear Tensor-Mass Model. In *Proceedings of MICCAI'00*, pages 643–652, London, UK, 2000. Springer-Verlag.

[Pic03] G Picinbono. Non-linear anisotropic elasticity for real-time surgery simulation. *Graphical Models*, 65(5):305–321, September 2003.

[SDR+05] J.-M. Schwartz, M. Denninger, D. Rancourt, C. Moisan, and D. Laurendeau. Modelling liver tissue properties using a non-linear visco-elastic model for surgery simulation. *Medical image analysis*, 9:103–112, 2005.

[The12] The CGAL Project. *CGAL User and Reference Manual*. CGAL Editorial Board, 4.0 edition, 2012.

[TPF91] D. Terzopoulos, J. Platt, and K. Fleischer. Heating and melting deformable models. *The Journal of Visualization and Computer Animation*, 2(2):68–73, 1991.

An Adaptive Subdivision Scheme On Composite Subdivision Meshes

Anh-Cang PHAN
VinhLong College of Economics and Finance
VinhLong, Vietnam
pacang@vcef.edu.vn

Romain RAFFIN
Aix-Marseille University
CNRS, LSIS UMR 7296
13009, Marseille, France
Romain.Raffin@univ-amu.fr

Marc DANIEL
Aix-Marseille University
CNRS, LSIS UMR 7296
13009, Marseille, France
Marc.Daniel@univ-amu.fr

ABSTRACT

One of the commonly used techniques in hole filling and mesh joining is the construction of connecting meshes between meshes to generate a new mesh model consisting of the composite subdivision meshes. One problem in subdivision meshes is how to further subdivide this reconstructed mesh model or these composite subdivision meshes to enhance the quality of the surface as needed. In this paper, we propose a new local subdivision method of the composite subdivision meshes. Our method does not alter the surrounding mesh areas, and guarantees that the discrete continuity between these meshes is preserved without the occurrence of cracks or holes between them. We specifically address a local subdivision scheme on a connecting mesh (the mesh area covering hole or crack) suitable for refining only the connecting mesh or the selected mesh areas. Our method can produce a smooth mesh model with a natural shape, and allows approximation or interpolation of surfaces. We implement our method for various triangular meshes and present our experimental results.

Keywords

Subdivision surfaces, Adaptive scheme, Triangular mesh.

1 INTRODUCTION

3D object models with complex shapes are generated by a set of assembled patches or separate mesh areas which may be at different resolution levels, even with different subdivision schemes. Cracks, gaps or holes may appear along the boundary between these patches if we further subdivide them at different resolution levels, even with different subdivision schemes. On the other hand, some research related to mesh connection such as in [Phan12, Phan13a] often lead to the construction of a high quality connecting mesh CM (the mesh area covering hole or crack) between meshes M_1, M_2, and a continuous surface. After the connecting mesh CM is produced, we can further subdivide CM and/or M_1, M_2 of a model to enhance the quality of the surface as needed. Unfortunately, there are two problems posed for subdivision of CM and/or M_1, M_2. If we subdivide M_1, M_2 and CM separately by different subdivision schemes, i.e global subdivision schemes in Butterfly [Dyn90], Loop [Loop87], Kobbelt [Kobbelt96], cracks or holes can also reappear on the surface after subdivision. This prevents some further processing of the mesh and high quality rendering. Therefore, it is not ideal to apply a global scheme for this subdivision. But if we use adaptive subdivision methods, i.e methods available in [Amresh02, Liu04, Pakdel04, Pakdel07, Husain10, Husain11], the neighboring faces around the subdivided area are also subdivided to avoid cracks. Thus, it changes the connectivity and valence of vertices around the subdivided area, produces some extraordinary vertices and long faces that are unavoidable in most adaptive subdivision algorithms. This will not only alter the shape of the limit surfaces, but also reduce its smoothness. Consequently, the challenge is to subdivide the connecting mesh or composite subdivision meshes.

In order to deal with these problems, we have developed a local subdivision on a connecting mesh or composite subdivision meshes suitable for refining only the connecting mesh or the selected mesh areas so that it does not change the surrounding mesh areas, and ensures the surface continuity. Our contributions are as follows: 1) Provide a local subdivision scheme that manages whether or not a given face in a selected mesh needs to be subdivided at the next level of subdivision. 2) Propose an adaptive subdivision method of composite subdivision meshes defined with subdivision surfaces, each mesh being at a different subdivision level to generate a smooth discrete surface with a natural shape and visually fair connectivity.

The remaining of the paper is organized as follows: We briefly review previous and related work for subdivision surfaces in Section 2. Section 3 details the construction of our adaptive subdivision algorithm. We present

the continuity of surfaces generated by our method in Section 4. Then we show and compare implementation results of our algorithm in Section 5. Finally, we draw the conclusion in Section 6.

2 PREVIOUS WORK

As mentioned in Section 1, the subdivision schemes of the whole mesh will be inefficient because if a face is subdivided in a subdivision step but its neighboring faces are not, cracks will be created as shown in Fig. 1.

To overcome this problem, incremental adaptive subdivision is sometimes necessary in applications to refine only selected mesh areas of 3D models. Several adaptive subdivision methods [Amresh02, Liu04, Pakdel04, Pakdel07, Husain10, Husain11] had been proposed to determine the areas to be subdivided and handle cracks. These methods refine a subset of the faces of the control mesh and remove cracks so that the surface can be further used in applications. On the other hand, many methods dealing with cracks and holes are introduced in [Jiang, Cas05]. These methods adopt various criteria to compute the planar shape from a 3D surface patch. However, they do not mention the continuity and the progressive change in resolution between meshes after dealing with cracks and holes.

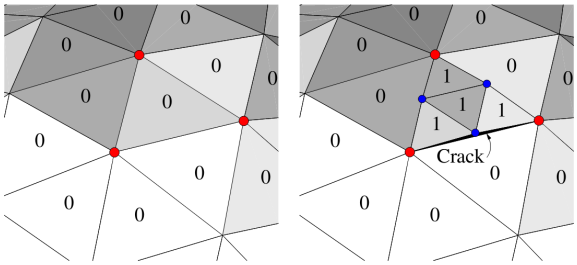


Figure 1: Crack on the mesh is caused by subdividing a face (taken from [Pakdel07]). The numbers represent the subdivision level.

Although these existing algorithms deal with cracks and holes, they provide partial or inefficient solutions for subdivision of a connecting mesh. These methods recommend refining the areas of interest with the same subdivision scheme. The area outside the adaptively subdivided area is also subdivided in the most common case to handle cracks, and therefore they create more faces. On the other hand, removing cracks with the insertion of edges makes a change in the connectivity of vertices. It also modifies the resulting surface of the subdivision process. Consequently, handling cracks is complex and expensive in computational time.

In fact, no existing adaptive subdivision method satisfies all desires for a smooth surface consisting of patches or meshes with different resolutions. In this paper, we introduce a new adaptive subdivision method on composite subdivision meshes or a connecting mesh between two meshes of different resolutions. This subdivision method is based on the *dead* property of vertices which are boundary vertices between triangular meshes to consider if the faces should be subdivided or not at the next subdivision level. It has been designed as an efficient solution for refining a selected mesh area without handling cracks and altering the original mesh areas around the adaptively subdivided area while maintaining the continuity between them. Consequently, it can avoid creating long faces both inside and outside the subdivided area. It permits to follow the junction between two initial triangular meshes along the subdivisions of the meshes. In addition, our subdivision algorithm can be easily applied to triangular subdivision schemes (i.e Butterfly or Loop schemes).

3 OVERVIEW OF THE ALGORITHM

Our method is inspired from the adaptive subdivision methods for Loop subdivision scheme [Amresh02, Liu04]. They are based on the angle of the normal vectors of adjacent faces of a face or a vertex considered as error estimation to decide whether the face needs to be subdivided or not at the next subdivision step. However, the user needs to determine a good threshold for these angles to set the "flatness" property of faces. Our adaptive subdivision method is based on the "*boundary vertices*" of the selected mesh area to define the property of each vertex to be *dead* or *alive* as introduced in Section 3.1. Based on the property of each vertex, our subdivision scheme only refines the connecting mesh or the selected mesh areas.

In order not to alter the mesh areas around the adaptively subdivided mesh area, faces of these areas closest to the original boundaries between meshes must not be subdivided and their boundaries are kept to avoid handling cracks. This problem is overcome by a dead vertex based refinement (DVR) scheme. It is designed to refine faces of the adaptively subdivided mesh such that the connectivity between meshes is preserved. In the following, we will first introduce a definition of *dead* vertices, edges and faces related to our algorithm.

3.1 Definition of dead vertices, edges and faces

Given two triangular meshes M_1 and M_2 joined by CM, an edge is usually shared by two triangular faces. If it is shared by only one, it corresponds to a boundary edge and its end vertices are called boundary vertices. Let us introduce some items used in our algorithm.

- **For vertices**: to keep the boundaries of meshes M_1, M_2 and CM, we will mark boundary vertices as *dead* vertices. In contrast, the remaining vertices are called *alive*. Thus, we can classify a vertex depending on its position.
- **For edges**: to keep boundary edges of meshes M_1, M_2 and CM, we must not subdivide these edges and mark them as *dead* edges. An edge is called *dead*, if its two vertices are *dead*. In all other cases, the edge is called *alive*.

 ISBN 978-80-86943-65-7

- **For faces**: A face is called *dead*, if its three vertices are *dead*. In all other cases, the face is called *alive*. *Dead* faces must not be subdivided.

  Let n be the degree of deadness of a triangular face which equals to its number of *dead* vertices ($0 \leq n \leq 3$). For instance, if $n = 3$ (all vertices of a face are classified as *dead*), the face is called a *dead* face. In case $n \leq 2$, the face is called an *alive* face. It will be suitably refined with DVR scheme where the *dead* edge connecting two *dead* vertices must not be subdivided to keep the boundaries of meshes. Details of the refinement are presented in the following section. Fig. 2 is an example of *dead* and *alive* vertices, edges and faces.

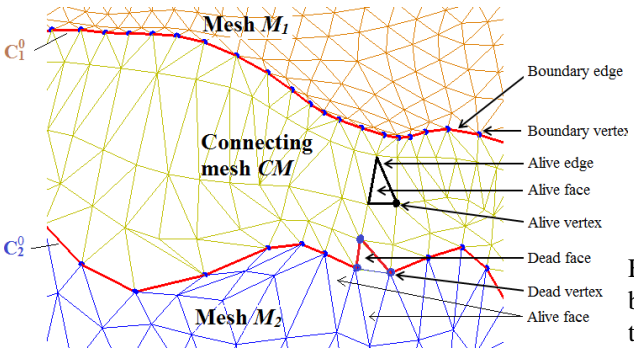


Figure 2: Definition of *dead* vertices, edges and faces.

3.2 Dead vertex based refinement scheme (DVR)

In order to refine a selected mesh area, such as the connecting mesh CM, we determine the property of faces of the model based on *dead* vertices. Our scheme called "Dead vertex based refinement scheme" is illustrated in Fig. 3. The rules of DVR scheme work as follows:

- **Classification of vertices**: For each vertex, we specify its property (*alive* or *dead*) based on definition in Section 3.1 and then mark it.

- **Adaptive subdivision rules**: For each face, we determine n the degree of deadness of a face based on the *dead* property of its vertices. Subdivision is then performed as shown in Fig. 3. There are four cases for a face $F_0 = \{v_1, v_2, v_3\}$ to create new subfaces based on the degree of deadness of F_0:

  - **Case n = 0**: if all vertices of F_0 are *alive*, F_0 is split into four new subfaces based on the regular subdivision method as shown in Fig. 3a. In this case, no new *dead* face is created while four new *alive* faces F_1, F_2, F_3, F_4 are created.
  - **Case n = 1**: if one vertex of F_0 is a *dead* vertex, F_0 is split into four new subfaces consisting of one newly created *dead* face and three newly created *alive* faces. This splitting indicates that *dead* faces spread gradually after each subdivision step. It implies that faces around the boundaries of meshes are gradually refined and thus it progressively changes the resolution between meshes. In Fig. 3b, one new *dead* face F_2 and three new *alive* faces F_1, F_3, F_4 are created.
  - **Case n = 2**: if two vertices of F_0 are *dead* vertices while one remaining vertex of F_0 is *alive*, F_0 is split into three new subfaces consisting of two newly created *dead* faces and one newly created *alive* face. Because we have two *dead* vertices, the *dead* edge connecting these two vertices must not be split while the remaining edges of F_0 are *alive* edges, and thus they are split. As a result, F_0 is split into three subfaces. In Fig. 3c, a triangular face is split into three smallest subfaces as possible. That is, three subfaces are created by comparing the length d_1 of edge v_2v_{31} with the length d_2 of edge v_3v_{12}. If $d_1 < d_2$, F_0 is split into one new *alive* face F_1 and two new *dead* faces F_2, F_3 to have an optimal split with more compact triangles. Conversely, F_0 is split into new subfaces F_1, (v_3, v_{12}, v_{31}), and (v_3, v_{12}, v_2).
  - **Case n = 3**: if all three vertices of F_0 are *dead* vertices, F_0 will not be split and thus no new subface is created (see Fig. 3d).

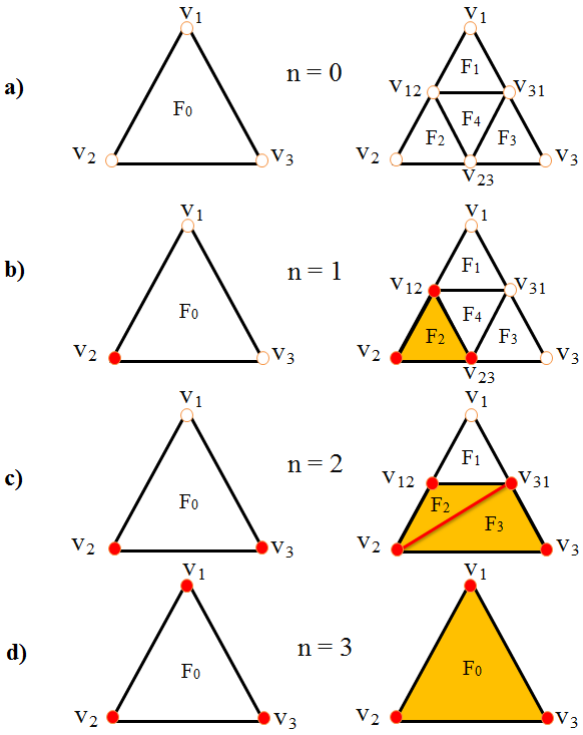


Figure 3: Our DVR scheme: adaptive subdivision rules based on the degree of deadness, where an *alive* vertex is represented as an empty circle; a *dead* vertex is represented as a red circle; an *alive* face is represented as an empty triangle; a *dead* face is represented as a yellow triangle.

 ISBN 978-80-86943-65-7

The local refinement scheme indicates that faces of CM (or the subdivided area) close to the boundaries are split less than faces far to these boundaries leading to a progressive change in resolution between meshes. It can be also applied adaptively to generate more details in needed areas.

3.3 Our proposed adaptive subdivision method

A process of boundary detection is performed before applying DVR scheme to have the suitable mesh refinements. The adaptive subdivision is performed in two phases (Fig. 4).

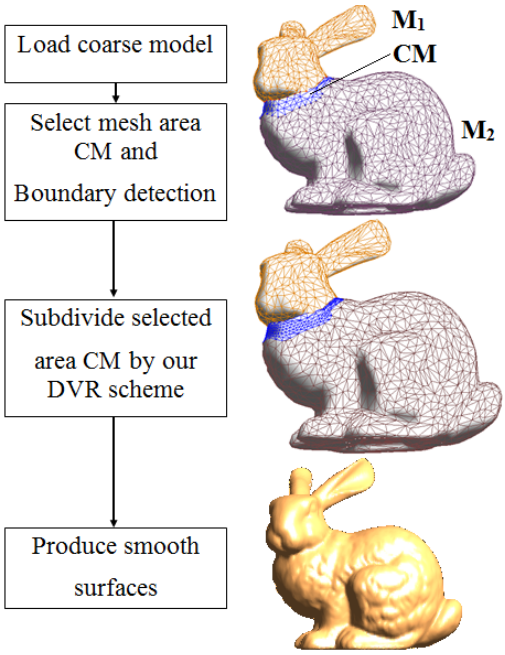


Figure 4: Framework for our subdivision method.

1. **Phase 1. Select mesh area CM and Boundary detection.**

   In this phase, we first read an input model at subdivision level j including meshes M_1 and M_2 joined by a connecting mesh CM (see Fig. 2). Then, we detect and mark boundary vertices of M_1, M_2 and CM.

   In this work, we join M_1 and M_2 by CM2D-TPW method [Phan13a]. CM is constructed by adding triangle strips to each boundaries of M_1 and M_2 until they are close enough to be linked based on the distance between them (smaller than a user specified threshold). This is done by using a tangent plane local approximation to create vertices of CM. Then, a Lifted B-spline wavelet transform is applied for multiresolution analysis of CM to obtain a progressive change in resolution between meshes after joining them together.

2. **Phase 2. Subdivide CM by our DVR scheme.**

   The adaptive subdivision process on CM at the next subdivision level $j+1$ is based on DVR scheme and described as follows:

   - ***Step 1. Classification of vertices and faces***. This step is to determine the *alive* or *dead* property of vertices and faces. Initially, we can consider the marked boundary vertices as a list of *dead* vertices. All the remaining vertices are set to *alive*. Based on definition of *dead* edges and faces in Section 3.1, we obtain the lists of *dead* edges and faces.
   - ***Step 2. Creation and classification of new vertices***.

     - New vertex creation: The vertex creation improves the smoothness of the mesh and the transition between M_1, M_2 and CM. No new vertices will be created for *dead* edges. For each *alive* edge, a new vertex is created by the odd vertex masks of the Linear, Butterfly or Loop subdivision schemes. The choice of these subdivision schemes to apply for our method does not depend on the subdivision schemes for initial meshes M_1 and M_2. Note that in case of the Linear subdivision scheme, each triangular face is divided into four new subfaces by adding new vertices in the middle of each edge. In case of the Loop subdivision scheme, if we use this scheme to create new vertices, *alive* vertices (even vertices) of CM will be repositioned as a linear combination of their neighbors by the even vertex mask.

     - New vertex classification: a newly created vertex of an *alive* edge is classified as a *dead* vertex if one vertex of the edge is a *dead* vertex. Otherwise, it is set to be an *alive* vertex as shown in Fig. 3. After that, we mark the property of the newly created vertices. This implementation is to gradually increase the number of *dead* vertices after each subdivision step to obtain a progressive change in resolution.
   - ***Step 3. Creation and classification of new faces***. For each *dead* face ($n = 3$), no new subfaces will be created. For each *alive* face ($0 \leq n \leq 2$), new subfaces will be created by adaptive subdivision rules of DVR scheme (see Fig. 3).
   - ***Step 4. Update a list of*** *dead* ***vertices***. We add the newly created vertices which are marked to be *dead* into a list of *dead* vertices consisting of existing *dead* vertices and new *dead* vertices. Finally, we repeat steps 2 through 4 up to a user defined number of iterations.

4 THE CONTINUITY OF SURFACES GENERATED BY OUR METHOD

Because we concentrate on subdivision of triangular faces and the new vertices created by our method are based on triangular subdivision schemes, the convergence of subdivision surfaces built with our method depends on these schemes. For example, if we apply our method combined with the Loop scheme to create new vertices, the convergence analysis of our method

 ISBN 978-80-86943-65-7

resides in the convergence of this scheme presented in [Loop87].

In this work, we implement our subdivision method on Linear, Butterfly, and Loop schemes, but it can be applied to other triangular schemes with at least C^1 continuity. Obviously, it was shown that a subdivided mesh converges to a limit surface with C^1 or C^2 continuity except at boundary vertices if our method is applied to the Butterfly or Loop schemes respectively [Dyn90, Loop87]. A review of the continuity of surfaces for these subdivision schemes can be found in [Loop87, Dyn90, Doo78, Zorin96, Kawa06]. Besides, our method maintains the connectivity between meshes M_1, M_2 and CM because the boundaries of these meshes are kept unchanged during subdivision. On the other hand, our method can also be applied to single surfaces. The experimental results to illustrate our algorithm are shown in Section 5.

5 APPLICATIONS AND RESULTS

We implement our adaptive subdivision method for various triangular meshes of 3D object models. In this section, we give a number of experimental results. Our proposed algorithm has been implemented in Matlab.

5.1 Subdivision surfaces

In this section, the purpose of the application is to show that our method can produce less triangular faces than the others such as Linear, Butterfly, and Loop methods while keeping the surface continuity. We apply our adaptive subdivision method to the Linear, Butterfly or Loop subdivision schemes for refining the Mannequin mesh model. We also give some results of our adaptive subdivision with three these different subdivision methods. Fig. 5 shows the comparison of the Linear subdivision method and our adaptive method using the Linear scheme at level 1, where *dead* vertices consist of original boundary vertices and new *dead* vertices which are represented as red points and yellow points respectively after each subdivision step. It can be seen that our adaptive subdivision does not alter the original boundary of the Mannequin mesh while subdividing faces. The faces close to the boundary can be kept unchanged or be split into three subfaces. Moreover, our method can avoid handling cracks due to a difference in subdivision resolution of neighboring faces.

Some meshes and Gaussian curvature maps of the Mannequin model generated from Butterfly, Loop subdivision schemes and our method are illustrated in Figs. 6, 7. We can see that the quality of the surfaces subdivided by Butterfly, Loop schemes and our method is the same except in the boundary area where our method also applies the Butterfly, Loop schemes to create new vertices during subdivision process. Indeed, based on our subdivision rules and the results of our experiment, if we apply our method to the Butterfly or Loop subdivision schemes and the list of *dead* vertices is empty, our

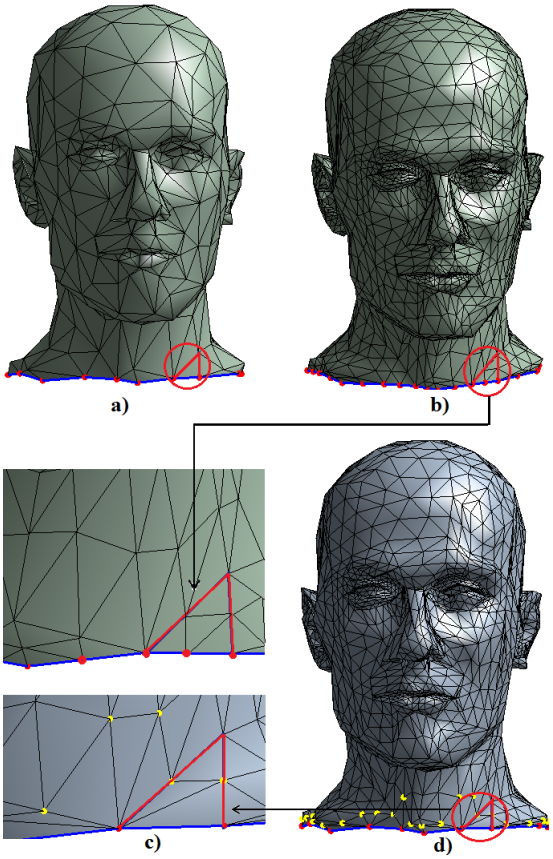


Figure 5: The Mannequin mesh generated by the Linear subdivision and our adaptive subdivision applied to the Linear subdivision scheme: a) Coarse mesh; b) Linear subdivision at level 1; c) Zooms of refined mesh by Linear subdivision and our method; d) Our adaptive subdivision at level 1.

adaptive subdivision scheme is the same with the Butterfly or Loop schemes. However, in our method the numbers of *dead* vertices and *dead* faces increase after each subdivision step as illustrated by yellow vertices in Figs. 6, 7. Consequently, it can efficiently decrease the number of produced faces in every subdivision step as listed in table 1 while keeping the continuity property of subdivision surfaces.

We made some comparisons for the basic Linear, Butterfly, Loop subdivision methods and our method using these schemes at two subdivision levels. Table 1 shows the subdivision steps, the number of vertices (V) and faces (F) of meshes and the mesh reduction rate r for the Mannequin model, where $r = \frac{n_1 - n_2}{n_1} 100$, n_1 is the number of vertices or faces of meshes subdivided by Linear, Butterfly, Loop schemes, n_2 is the number of vertices or faces of meshes subdivided by our method. The more our subdivision process is implemented, the more *dead* vertices and faces are created. Consequently, the number of newly created vertices and faces of the refined mesh will decrease and thus the reduction ratio will increase.

Based on our results and comparisons, it can be seen that our proposed method can improve the basic subdi-

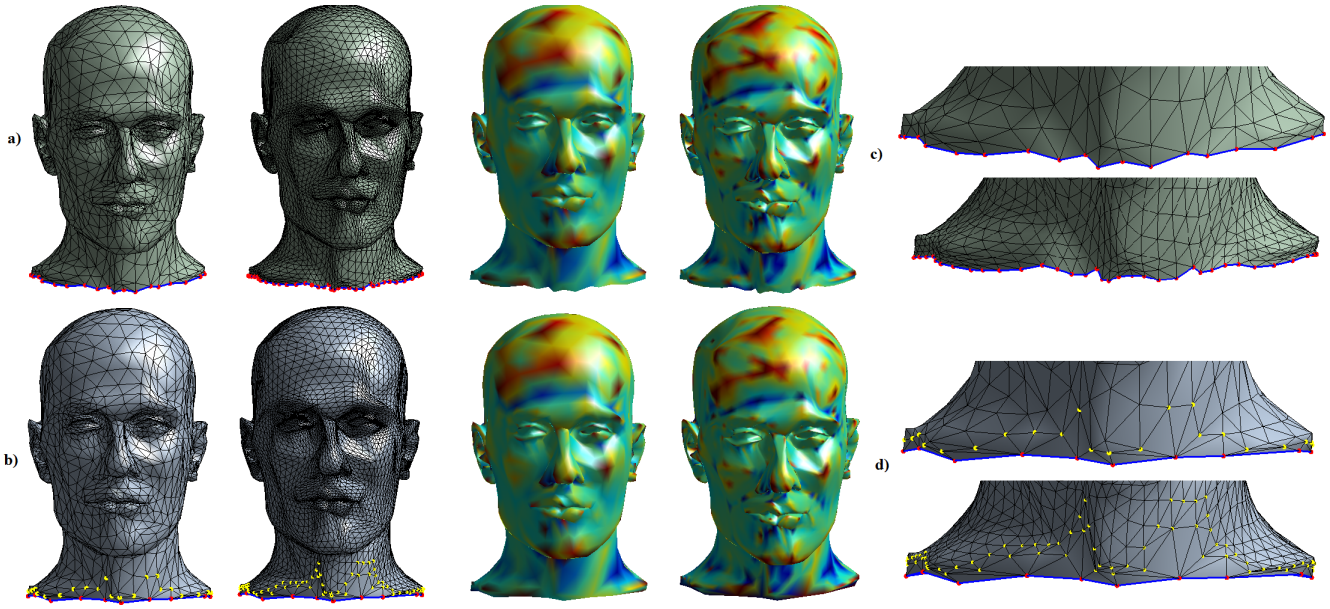


Figure 6: The Mannequin meshes and Gaussian curvature maps are produced by: a) Butterfly subdivision at levels 1 and 2; b) Our adaptive subdivision applied to the Butterfly subdivision scheme at levels 1 and 2; c) Zoom of one of the interesting parts of meshes in Fig. 6a; d) Zoom of one of the interesting parts of meshes in Fig. 6b.

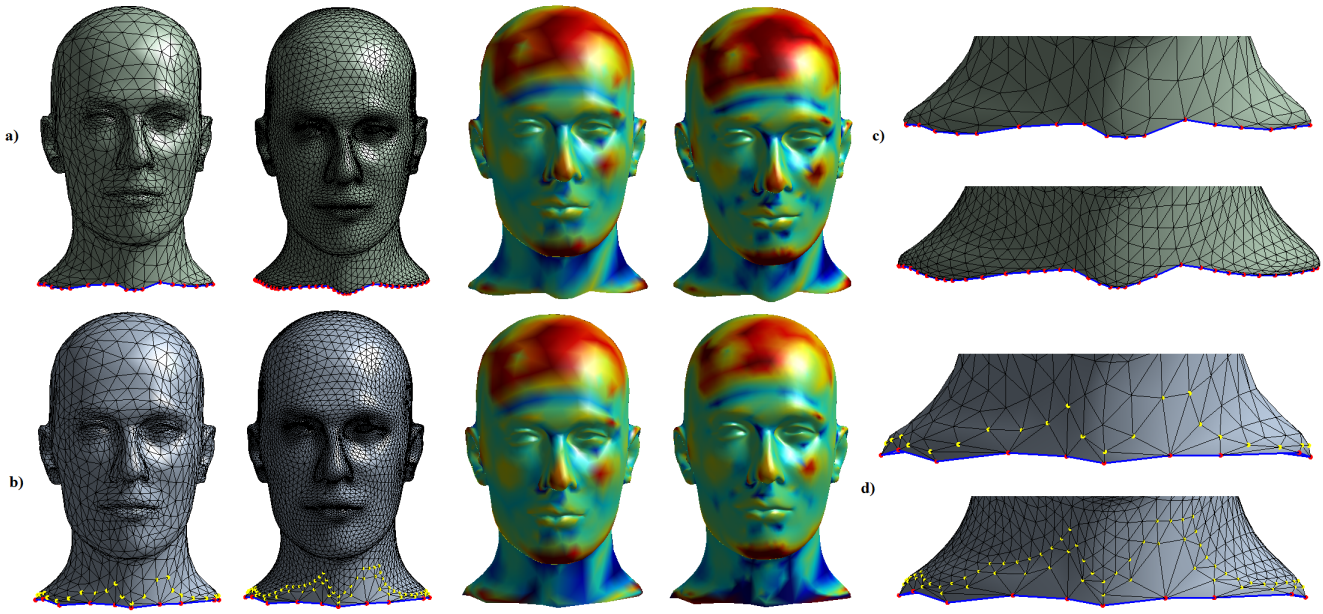


Figure 7: The Mannequin meshes and Gaussian curvature maps are produced by: a) Loop subdivision at levels 1 and 2; b) Our adaptive subdivision applied to the Loop subdivision scheme at levels 1 and 2; c) Zoom of one of the interesting parts of meshes in Fig. 7a; d) Zoom of one of the interesting parts of meshes in Fig. 7b.

Subdivision Method	**Coarse mesh**		**1 iteration**		**2 iterations**	
	V	F	V	F	V	F
1. Linear, Butterfly and Loop	428	839	1694	3356	6743	13424
2.Our adaptive subdivision	428	839	1679	3341	6598	13179
Reduction rate r (%)	0	0	0.89	0.45	2.15	1.83

Table 1: The number of vertices (V) and faces (F) in a Mannequin mesh model refined by Linear, Butterfly, Loop method and our method at different subdivision steps.

vision methods because it can reduce a number of vertices and faces while it guarantees the continuity, and locality of subdivision surfaces without causing cracks. The continuity of surfaces generated by our method depends on the applied subdivision scheme as presented in Section 4. Consequently, surfaces produced by our

 ISBN 978-80-86943-65-7

method are continuous at least C^1, except at extraordinary vertices because we implement our method on the Butterfly or Loop schemes.

5.2 Subdivision of a connecting mesh on a hole

Next, we introduce another application of our method for refining a connecting mesh on a hole. To further subdivide a connecting mesh CM on a hole, we give a simple example of the Tet model with a hole filled by HF-RBFC method [Phan13b] using a RBF local interpolation and a centroid interpolation of boundary vertices. Then, we apply our adaptive subdivision scheme for the connecting mesh CM of the reconstructed mesh model consisting of the original mesh M and CM. Our method efficiently handles subdivision of these meshes with different levels of resolution, even with different subdivision schemes. We show results in Fig. 8 by applying our method to Loop and Butterfly schemes where the first row represents meshes and the second row represents surfaces. In our method, we designed a locally-controlled subdivision scheme. Thus it allows different tension in different mesh areas of a model, where the details of the local tension are given in [Dyn90]. This is not a disadvantage, because one would often like to have different tension in different mesh areas for design flexibility in manipulating real world objects. For example, if we subdivide many times the two mesh areas of the Tet model as shown in Fig. 8, the tension will not be the same since boundary vertices and edges (also called *dead* vertices and edges) between these two meshes are kept. As a result, we will gain a mesh and surface with different tensions.

We note that the user must control the result and an additional criterion of subdivision is required to control this result when applying our method such that the generated surface has a natural shape as desired. For instance, we cannot subdivide the meshes too much to avoid a too large difference in resolution between them because the resulting surfaces are deformed with undesired shapes as illustrated in Figs.8d-e.

5.3 Incremental subdivision

Another application of our method is incremental subdivision through refining only some selected areas of the mesh. Therefore, we can create surfaces that are densely subdivided in areas of higher curvature or selected areas. As a result, the selected mesh areas become finer while the rest of the mesh is coarse. Furthermore, the refined mesh progressively changes in resolution between coarse and fine areas as shown in Figs. 9, and 10. These figures illustrate examples of incremental subdivision of the right eye area of the Mannequin mesh. This allows us to gradually increase the resolution of the selected areas while keeping the surface continuity. We first apply our method to the Loop subdivision scheme for the right eye area with two resolution levels as illustrated in Figs. 9a-c. To observe the quality of produced surfaces, we plot meshes (top row), zooms of the interest right eye area (middle row), and surfaces (bottom row). The refined mesh and surface quality of the right eye area in Fig. 9c are finer than the ones in Figs. 9a-b because the resolution is higher while still maintaining the continuity of the surface. We then incrementally subdivide the right eye area and the complementary mesh area of the Mannequin model by applying our method to two subdivision schemes, and with different resolution levels as plotted in Fig. 10. Two mesh areas consisting of the right eye area and the complementary mesh area are incrementally subdivided by applying our method to the same Loop scheme but at different subdivision levels as shown in Fig. 10a. As a result, we obtain the refined right eye area with higher resolution. We also test our method with Butterfly scheme and give a result as plotted in Fig. 10b. Besides, we incrementally subdivide two mesh areas with different subdivision schemes, such as Loop and Butterfly schemes, and at different levels as shown in Fig. 10c. We also obtain a finer mesh especially in the right eye area.

5.4 Subdivision of a connecting mesh between meshes

We give some examples of subdivision of a connecting mesh CM generated by our mesh joining method (CM2D-TPW method [Phan13a]). After joining meshes M_1, M_2 by the connecting CM, we can further subdivide CM and/or M_1, M_2 by our subdivision method. Fig. 11 illustrates examples of applying our method to Butterfly or Loop schemes.

We first join two meshes M_1, M_2 to generate the connecting mesh CM using CM2D-TPW method (see Fig. 11b). From the reconstructed model in Fig. 11b, we then subdivide CM by our subdivision scheme to generate a new connecting mesh CM_1 while we do not recompute M_1, M_2 (Fig. 11c). Next, we subdivide M_1, M_2 by our scheme to produce new meshes M_1', M_2' while CM is kept as illustrated in Fig. 11d. Finally, we apply our method to further subdivide all meshes M_1, M_2, and CM. As a result, we obtain new meshes M_1', M_2', and CM_2. Fig. 11e shows that the new mesh CM_2 remains properly join with both new meshes M_1', M_2'.

To see the quality of the resulting meshes easily, Fig. 11 shows the images of the refined meshes and zooms of the corresponding meshes, where a) two initial meshes M_1, M_2 before connecting; b) the connecting mesh CM produced by CM2D-TPW method; c) CM is subdivided by applying our subdivision method to Loop scheme at level 1; d) M_1 and M_2 are subdivided separately by applying our method to Loop scheme at level 2 and Butterfly scheme at level 1, respectively; e) M_1, M_2, and CM are subdivided separately by applying our method to Loop scheme at level 2, Butterfly scheme at level 1, and Loop scheme at level 2 respectively; f)-h) Zooms of the corresponding meshes. According to the experimental results, our adaptive subdivision method is ef-

 ISBN 978-80-86943-65-7

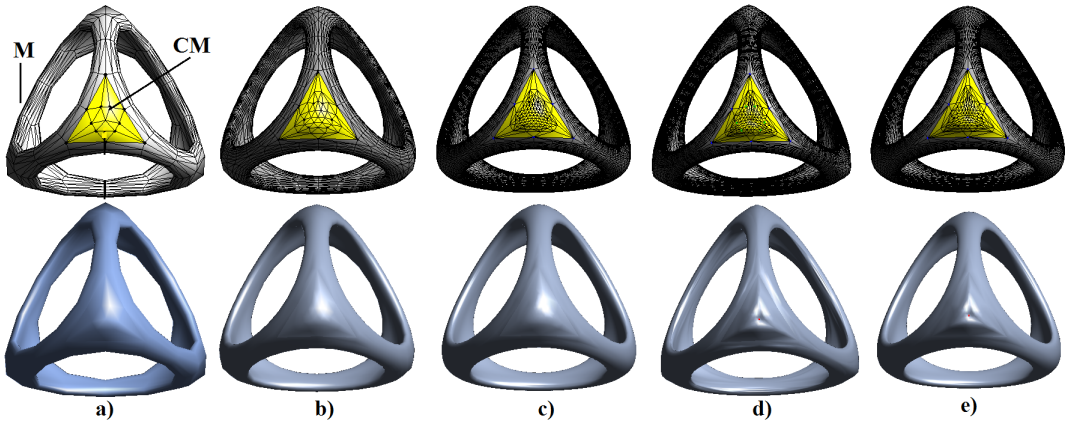


Figure 8: Zoom of subdividing the connecting mesh CM on a hole of the Tet model. The two meshes M and CM are subdivided by applying our method to two schemes respectively: a) Coarse mesh; b) Loop-Loop at level 1; c) Loop-Loop at level 2; d) Butterfly-Butterfly at level 2; e) Loop-Butterfly at level 2.

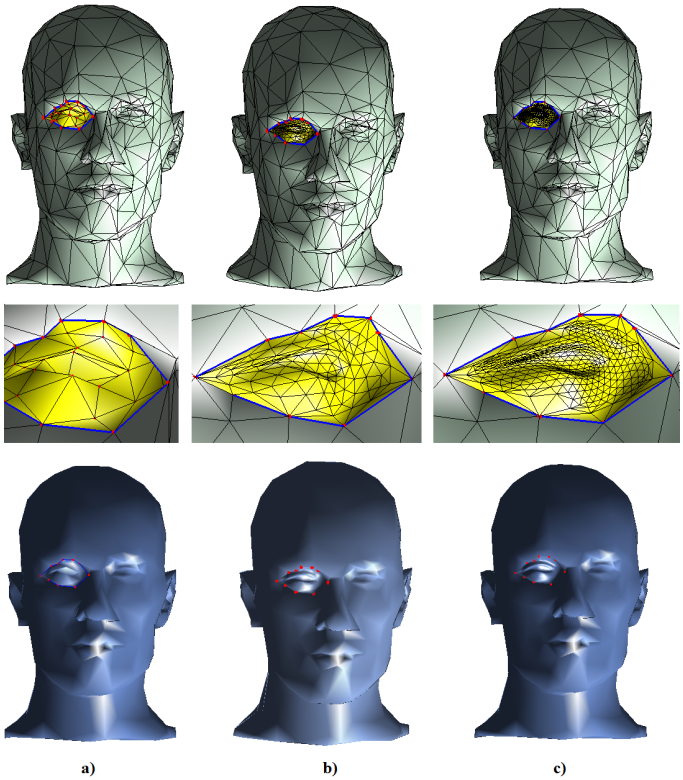


Figure 9: Incremental subdivision of the right eye area of the Mannequin model by applying our method to Loop scheme: a) Coarse mesh; b) at level 1; c) at level 2.

ficient to further subdivide the connecting mesh CM and/or meshes M_1, M_2. It can keep the boundaries of meshes and does not alter the surrounding meshes during subdivision. Therefore, our method can avoid handling cracks. As already mentioned in Section 5.2, the user must pay attention to the number of additional subdivisions. For instance in this example, a criterion to check for flatness of the group of triangles sharing the same vertex could be useful.

6 CONCLUSION

In this paper, we have introduced a new adaptive subdivision method for triangular meshes. Our method can be efficiently applied to the Butterfly or Loop subdivision schemes. It is based on the *dead* property of vertices. New vertices are created by these basic subdivision schemes. Therefore, our adaptive subdivision method produces C^1 and C^2 continuous surfaces, except at extraordinary vertices and boundary vertices. Our adaptive subdivision is a good completion work in filling holes and joining meshes because it can be applied on the connecting mesh CM and/or selected meshes M_1, M_2 so that new CM remains a correct connection with new subdivided meshes M_1 and M_2. This method is simple, reliable for adaptive subdivision to refine some selected mesh areas as needed. It allows us to separately refine selected mesh areas of a model at

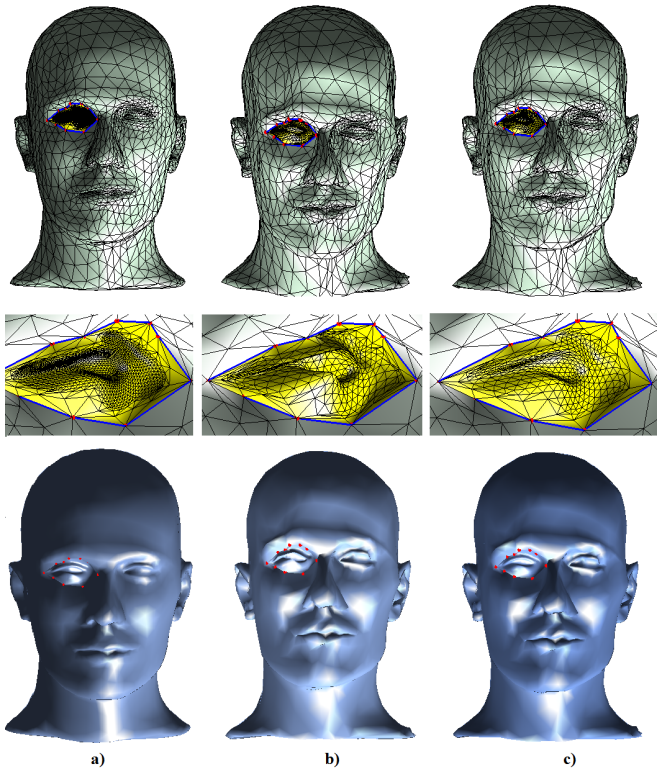


Figure 10: The right eye area and the complementary mesh area of the Mannequin model are incrementally subdivided by applying our method to two subdivision schemes respectively: a) Loop at level 3 for the right eye area and Loop at level 1 for the complementary area; b) Butterfly at level 2 for the right eye area and Butterfly at level 1 for the complementary area; c) Loop at level 2 for the right eye area and Butterfly at level 1 for the complementary area.

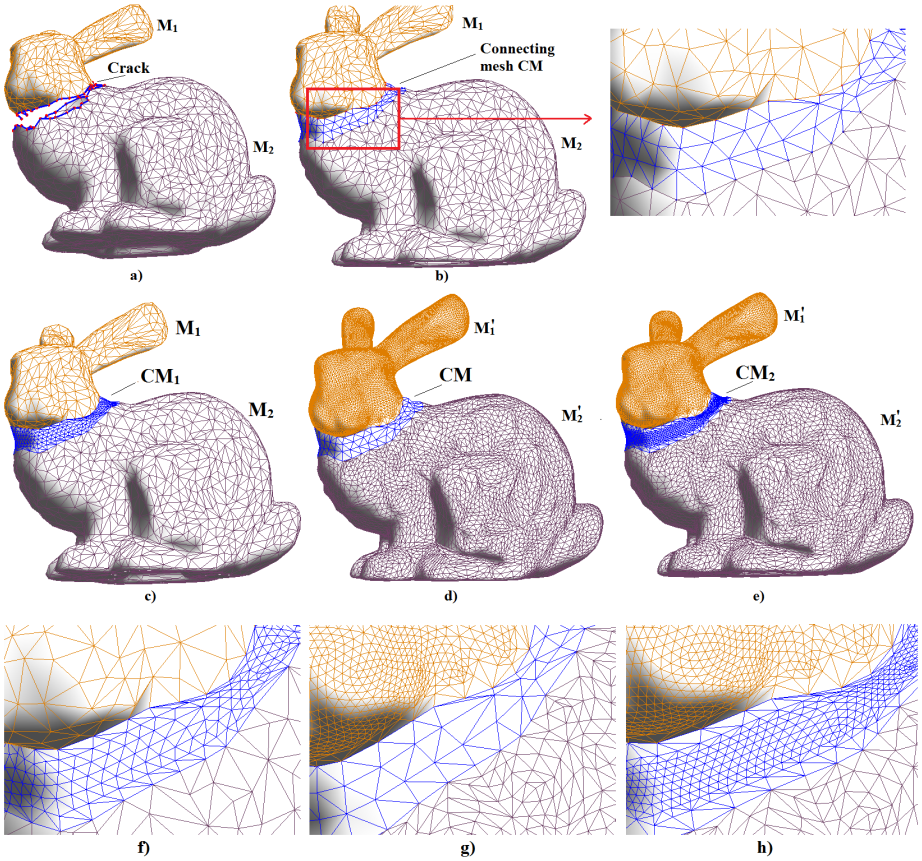


Figure 11: Two meshes M_1, M_2 and CM of the Bunny model are further subdivided by our adaptive scheme.

 ISBN 978-80-86943-65-7

different resolution levels while maintaining the valid connectivity between meshes. Our method does not change the original mesh areas around the adaptively subdivided mesh during subdivision process. It can be also a drawback as if the original object is coarsely described, the subdivision does not alter these areas and a difference of density of vertices and faces is created. Moreover, our method does not create high valence vertices near the boundaries which can lead to ripple effects on surfaces as some existing algorithms [Zorin97, Seeger01, Amresh02]. Additionally, it can keep the original boundaries between meshes, thus our adaptive subdivision rules can prevent the appearance of cracks without needing to handle them.

As the boundaries are not changed in our adaptive subdivision method, the meshes cannot be subdivided too much and therefore a reasonable number of adaptive subdivisions is advised to avoid an important difference in resolution between meshes. The user must control the result such that the generated surface has a natural shape as desired. In addition, although our method can be applied for meshes CM and/or M_1, M_2, some slim or narrow triangular faces may be formed at the boundaries between meshes. This case happens because original meshes around the adaptively subdivided mesh and the original boundaries between them are kept. In order to suppress this drawback, it could be possible to insert vertices on existing boundary edges and make possible a local re-triangulation yielding to more regular triangular faces without modifying the boundaries. As we focus in the work we presented on keeping the two resulting subdivision surfaces unchanged we did not modify the boundaries. This study will be one of our future work.

7 REFERENCES

[Amresh02] A. Amresh, G. Farin, and A. Razdan. Adaptive Subdivision Schemes for Triangular Meshes. In Hierarchical and Geometric Methods in Scientific Visualization, pp. 319-327, Springer-Verlag, 2002.

[Cas05] G. Casciola, D. Lazzaro, L. B. Montefusco, and S. Morigi. Fast surface reconstruction and hole filling using positive definite RBFs. Numerical Alg., Vol.39, No.1-3, pp. 289-305, 2005.

[Doo78] D. Doo and M. Sabin. Behaviour of recursive subdivision surfaces near extraordinary points. In Computer Aided Design, Vol. 10, No. 6, pp. 356-360, 1978.

[Dyn90] N. Dyn, D. Levin, and J. A. Gregory. A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control. ACM Transactions on Graphics, Vol. 9, pp. 160-169, 1990.

[Husain10] N. A. Husain, A. Bade, R. Kumoi, and M. S. M. Rahim. Iterative selection criteria to improve simple adaptive subdivision surfaces method in handling cracks for triangular meshes. In Pro. of the 9th ACM SIGGRAPH Conference on Virtual-Reality Continuum and its Applications in Industry, pp. 207-210, ACM, USA, 2010.

[Jiang] D. Jiang and N. F. Stewart. Reliable joining of surfaces for combined mesh-surface models. In Pro. of 21st ECMS, pages 297-303, 2007.

[Kawa06] H. Kawaharada and K. Sugihara. C^k-continuity of Stationary Subdivision Schemes. Technical reports/Mathematical engineering technical reports, The University of Tokyo, 2006.

[Kobbelt96] L. Kobbelt. Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology. In Computer Graphics Forum, pp. 409-420, 1996.

[Liu04] W. Liu and K. Kondo. An Adaptive Scheme for Subdivision Surfaces based on Triangular Meshes. Journal for Geometry and Graphics, Vol. 8, No. 1, pp. 69-80, 2004.

[Loop87] C. Loop.Smooth Subdivision Surfaces Based on Triangles. Master's thesis, University of Utah, 1987.

[Husain11] N. A. Husain, A. Bade, M. Rahim. Iterative Process to Improve Simple Adaptive Subdivision Surfaces Method with Butterfly Scheme. In Pro. of the World Academy of Science, Engineering and Technology 79, pp. 622-626, 2011.

[Pakdel04] H.-R. Pakdel and F. F. Samavati. Incremental Adaptive Loop Subdivision. In Pro. of the ICCSA 2004, pp. 237-246, Springer, 2004.

[Pakdel07] H.-R. Pakdel and F. F. Samavati. Incremental subdivision for triangle meshes. In Journal of Computational Science Engineering, Vol. 3, No. 1, pp. 80-92, July 2007.

[Phan12] Anh-Cang Phan, R. Raffin, and M. Daniel. Mesh connection with RBF local interpolation and wavelet transform. In Pro. of the SoICT '12, pages 81-90, ACM, New York, NY, USA, 2012.

[Phan13a] Anh-Cang Phan, R. Raffin, and M. Daniel. Joining Meshes with a Local Approximation and a Wavelet transform. In Pro. of the WSCG 2013, pages 29-38, Plzen, Czech Republic, 2013.

[Phan13b] Anh-Cang PHAN. Crack removal and hole filling on composite subdivision meshes. Ph.D. thesis, Aix-Marseille University, France, 2013.

[Seeger01] S. Seeger, K. Hormann, G. Hausler, and G. Greiner. A Sub-Atomic Subdivision Approach. In Pro. of the VMV-2001, Stuttgart, Germany, November 21-23, pp. 77-86, Aka GmbH, 2001.

[Zorin96] D. Zorin, P. Schröder, and W. Sweldens. Interpolating Subdivision for Meshes with Arbitrary Topology. Computer Graphics Proceedings (SIGGRAPH 96), Vol. 30, pp. 189-192, 1996.

[Zorin97] D. Zorin, P. Schröder, and W. Sweldens. Interactive Multiresolution Mesh Editing. Computer Graphics, Vol. 31, pp. 259-268, August 1997.

Single chord-based corner detectors on planar curves

Naurin Afrin, Wei Lai
Department of Computer Science and Software Engineering,
Swinburne University of Technology, Australia
nafrin@swin.edu.au, wlai@swin.edu.au

ABSTRACT

Detecting corner locations in the images plays a significant role in several computer vision applications such as motion detection, image registration, video tracking, image mosaicing, panorama stitching and object recognition. In this paper we have analyzed an existing state of art, Chord to Point Distance Accumulation (CPDA) corner detector and modified this detector in way that it uses a single chord instead of using three different chords. We named this detector as Single Chord CPDA (SCCPDA). We have also proposed a simple but effective new detector of detecting robust corner locations against different image transformations using cumulative distance calculation. The new detector has also used a single chord and we named it as Chord to Cumulative Sum Ratio (CCSR). A comprehensive performance evaluation has been performed by using Average Repeatability and Localization Error. We have found that the SCCPDA and CCSR detectors perform better than the original CPDA detector. Our experimental results show that the CCSR using simple cumulative calculation outperforms eight other existing contour based corner detectors in terms of repeatability and generates one of the lowest localization errors. In addition, the CCSR detector is also most efficient corner detector among other contour-based corner detectors.

Keywords

corners, contour-based corner detection, single chord, CPDA

1 INTRODUCTION

Feature detection is a major concern in the field of computer vision and image processing. Among different types of features, corner is one of the most stable features. There are a number of corner detectors have been proposed so far to detect repeatable corners. In this paper we concentrate on contour-based corner detectors, in particular, this paper analyses certain properties of the very popular chord-to-point distance accumulation (CPDA) detector, proposed by [Moh08]. Although CPDA has been reported to be one of the best corner detectors [Pen13, Moh10], we found that the CPDA detector uses multiple chord lengths and these chords do not give extra benefit rather it increases the overhead on the whole process. We demonstrate in the paper that, by using a single chord with accumulating distances, it is possible to get even better performance than CPDA achieving better efficiency. Finally, we propose another single chord detector of calculating the curvature values using cumulative distances among the points of a curve to detect corner locations. The detector achieves the best repeatability among existing detectors with highest efficiency.

This paper is organised as follows. Section 2 reviews the contour based corner detectors while section 3 explains particularly about CPDA detector. Section 4 analyses the CPDA detector using a single chord and introduces a new corner detector using cumulative sum. The experimental results are presented in section 5. Finally, section 6 concludes the paper.

2 RELATED WORK

The typical representative among widely used contour based corner detectors is Curvature Scale Space (CSS) [Fmo01].The main idea of this approach is to extract edges using edge detectors, such as canny edge detector [Can86], then estimate the curvature of each point of the extracted edge and finally apply gaussian smoothing scale [Dav80] to smooth the planner-curves. Thus the absolute curvature maxima points obtained using specific thresholds by removing weak and false corners which are generally known as pseudo-corners. This detector has some major drawbacks. CSS detector uses a coarse smoothing scale to identify approximate locations of the corners and then uses a finer scale to track locations to improve the localization of these corners. A coarse scale is robust to noise but it may miss many potential corners. On the other hand, finer scale is good to detect spurious corners but it is sensitive to noise.

To overcome the weakness of CSS detectors, several detectors have been proposed which use multiple scales for curvature estimation. To detect corners, Awrangjeb et al. proposed a three scale-based detector (*ARCSS*) [Moh07] which uses affine-length parametrization. Although ARCSS detects more repeatable corners, it is computationally very expensive. Zhang *et al*. [Xia07] introduced another CSS based detector, Multi-scale Curvature Product(*MSCP*), which finds corners by multiplying curvature values using three scale to discard noise and false corners. A chord based detector, CTAR [RMN11a], measure curvature values based on a triangular theory.This detector is very efficient compared to CPDA detector. Another corner detector proposed by [RMN11b], is the combination of CPDA and efficient high curvature point detector IPAN [Dmi99]. Although this detector has better repeatability, the detector is computationally expensive. He and Yung [Xia04] modified original CSS detector in two steps. First, they use an adaptive local threshold to the curvature of its neighborhood region. Next, they find the angle of proper region of support to detect the corner. Eigenvalues of the covariance matrix [Dum99] and Gradient Correlation Matrix [Zha10] have also been used to derive contour orientation. Zhang et al. [Xia09] proposed another detector which applies multiple levels of Difference of Gaussian (DoG) on a curve to obtain corresponding planer curves that are later used to detect corners. This detector is also relatively expensive to compute.

3 OVERVIEW OF CPDA CORNER DETECTOR AND ITS WEAKNESSES

CPDA detector [Moh08] follows the basic idea of css detector described in Section 2. After extracting and smoothing out the curves, three chords of different length are moved along each curve to measure discrete curvature value for each location on the curves as L_i where i ε 10, 20, 30. Here the chord L_i indicates a straight line placed *i* pixels apart on the curve.

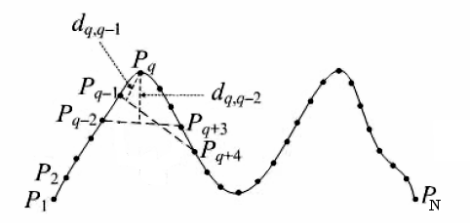


Figure 1: Curvature estimation using CPDA with chord

In Fig.1, let P_1, P_2, P_3,...., P_N be the N points on the curve. Now the value *i* of chord L_i defines the number of points of the curve segment between points P_j and P_{j+i}. To estimate the curvature value $h_{L_i(q)}$ at point P_q using a chord which is i pixels apart, the chord is moved on each side of P_q for at most i points keeping P_q as a central point and figure out the distances $d_{q,j}$ from P_q to the chord. In the end, CPDA assembles the curvature estimation using the Equation 1

$$h_{L_i}(q) = \sum_{j=q-i+1}^{q-1} d_{q,j} \tag{1}$$

Then the curvature values estimated using each chord, are normalized using Equation 2.

$$h'_{L_i}(q) = \frac{h_{L_i}(q)}{max(h_{L_i})}, \text{for } 1 \le q \le N, i \in \{10,20,30\} \tag{2}$$

The values calculated for three different chords from Equation 2 are then multiplied using Equation 3.

$$H(q) = h'_{L_{10}}(q) \times h'_{L_{20}}(q) \times h'_{L_{30}}(q), \text{for } 1 \le q \le N \tag{3}$$

Next, CPDA finds the candidate corners by rejecting weak corners using local maxima of absolute curvature and by comparing the curvature values with threshold T_h [0.2]. Then, CPDA calculates angle from a candidate corner to its two neighbouring candidate corners. The computed angle is then compared with the angle threshold ($\delta = 157^\circ$) to remove false corners.

CPDA proposes to use big neighbourhood to estimate the curvature which tends to miss corner locations which are specially close to each other [RMN11a]. CPDA detector might potentially miss obvious corners if they were located closely. This is due to using multiple chords to measure curvature values. Besides this, CPDA detector fails to detect corners if there are two or more corners on the same curve and one of them is very sharp and another is less but should be considered as corner location. In this scenario CPDA misses the less sharp corner location. There are also some scenarios where the multiplication of three chords' estimated curvature values is responsible for losing the extrema on the corner location. In summary, all the limitations are being created due to using three different length of chords.

4 PROPOSED CORNER DETECTORS

In this section, we will first propose the single chord CPDA (SCCPDA) detector and next, we will propose another approach named Chord to Cumulative Sum Ratio(*CCSR*) to estimate the curvature value based on cumulative sum analysis. Both of the detectors start with the basic idea of CSS detector of extracting and smoothing the curves as described in Section 2.

Single Chord CPDA detector

The original CPDA paper [Moh08] mentions the use of multiple chords, but does not justify the reason for doing so with any experimental results. In particular CPDA uses three chords of length 10, 20 and 30. To understand whether using multiple chords is superior to using single chords, we modified the CPDA process to use a single chord L_i, i ε [5, 30], where i has been decided experimentally from the experimental results presented in Section 5. The modified CPDA detector does not need the step shown in Equation 3, as it is only meaningful when using multiple chords. However, as described in [Moh08], apart from combining the distances of multiple chords, Equation 3 also magnifies the difference between the chord-to-point distances of weak and strong corners. To achieve a similar goal we replace the step shown in Equation 1 with Equation 4, and follow it with the normalisation shown in Equation 2. Rest of the processes of using curvature threshold and angle threshold are kept same.

$$h_{L_i}(q) = \left(\sum_{j=q-i+1}^{q-1} d_{q,j} \right)^2 \tag{4}$$

CCSR Detector

Now, we will explain another approach which also use a single chord to detect corners. One of the basic ways of finding corner locations is to measure the flatness of a curve. In other ways, corner is a location where the slope of the curve changes the direction i.e. the curve is not flat. According to this hypothesis, if we put a straight line on the curve touching two ends of the curve or curve segment, the ratio of the length of the straight line to the curve length will give the essence of the flatness of the curve. As a result, this ratio value will also carry the information of the presence of corner location on the curve or curve segment if the curve length is small.

(a) (b) (c)

Figure 2: Three different curves with three different flatness

Figure 2 (a)-(c) show three different curves of having different flatness. Now if we measure the ratio of the straight line to the curve length, we get 0.9232, 0.8039, 0.6706 respectively. The ratio values clearly indicates that lower value carries more bend characteristics of the curve. Therefore, if we calculate the ratio on a small part of the curve this might tell the presence of a corner in that segment of the curve. Next, the distances from all the interior locations of the curve segment to the respective straight line are calculated. The location of the highest distance to the straight line from the curve segment is then considered as a corner location. This also helps to localize the corner locations.

Similar to CPDA detector, the new detector also starts with extracting and smoothing curves from the image. Since the length of the curve segment needs to be measured, we have used cumulative sum for distance calculation to calculate the curve length of two given location on the curve. The cumulative sum of the distances between each point of the curve gives a sequence of a partial sum of the distances. For example, the cumulative sums of the distances between each location of the curve from the start $d_1, d_2, d_3, ...$ are $d_1, d_1+d_2, d_1+d_2+d_3, ...$ Therefore, the cumulative distance (CD)can be expressed as follows,

$$CD(i) = \sum^{i} d_i \tag{5}$$

Let $P_1, P_2, ..., P_N$ be the N points of a curve and a fixed length chord will be moved along with the curve. Please note that, the length of the chord in chord based corner detection means the number of points apart to place the chord on the curve. For example, if the chord length is $L = 5$, the chord is placed from location P_1 to P_6, next the chord is placed from P_2 to P_7 and so on. Each ratio is assigned to the middle location of the curve segment as expressed in Equation 6.

$$R(i+ceil(L)) = \frac{\sqrt{(x_{p_i} - x_{p_{i+L}})^2 + (y_{p_i} - y_{p_{i+L}})^2}}{CD(i+L) - CD(i)} \tag{6}$$

Now, we filter out the curve segments based on a threshold against the ratio, which are minima on a curve. Now the perpendicular distances from each interior points from the curve segment to the straight line is measured and the location having the longest distance is considered as a corner. We named this detector as Chord to Cumulative Sum Ratio (*CCSR*).

5 EXPERIMENT

This section presents the experimental setup and the results to evaluate the performance of the proposed detectors. First, the experimental setup and evaluation process have been described. Second, the characteristics of the SCCPDA detector and CCSR detector have been shown in term of different parameters.Third, the overall performance of these detectors have been compared with other existing corner detectors from literature. Finally, the efficiency of the detectors has been presented based on the time taken to detect corners by the detectors.

 ISBN 978-80-86943-65-7

Experimental Setup

To measure the performance of the corner detectors, the corner locations from the original images detected by the detectors are considered as the reference corners and then a known geometric transformation is applied on the original images to detect the corners again [Moh08, Moh07, RMN11a]. Now a corner of reference image is considered as repeated if the corresponding location of that corner is at a 3-pixels distance from a detected corner location in transformed image.

Two types of evaluation metrics are used to measure the robustness of the proposed detectors 1) Average Repeatability [Fmo01, Moh08] and 2) Average Localisation Error [Moh08]. These evaluation metrics have been used by [Moh08, Moh12] in order to compare the performances. The average repeatability R_{avg} measures the average number of repeated corners between original and transformed images. It is defined as,

$$AverageRepeatability = 100\% \times \frac{\frac{N_m}{N_o} + \frac{N_m}{N_t}}{2} \quad (7)$$

Where N_o and N_t are number of corners detected in the original and transformed images respectively and N_r is the number of repeated corners. The localization error L_e is defined as the amount of pixel deviation of a repeated corner. It is measured from the repeated corner locations between the original and transformed images,

$$L_e = \sqrt{\frac{1}{N_m}\sum_{i=1}^{N_m}(x_{oi} - x_{ti})^2 + (y_{oi} - y_{ti})^2} \quad (8)$$

Where (x_{oi},y_{oi}) and (x_{ti}, y_{ti}) are the positions of i-th repeated corner in original and transformed images respectively. Experiments were conducted on a 64 bit MATLAB R2009 (7.14.0.739) installation, running on Windows 7. The computer had an Intel Core i3 (2.3 GHz) processor with 4GB ram.

We have used an image dataset of 23 grey scale images to evaluate the performance of the corner detectors. The same dataset is also used by [Moh08, Moh12]. Seven Different transformations have been applied to these 23 images-Scaling, Shearing, Rotation, Rotation-Scale, Non-uniform Scale, JPEG Compression, Gaussian Noise.

Parameter Optimization

In this section, we have produced average repeatability of single CPDA detector against a series of chords from 5 to 30 and threshold from 0.15 to 0.25 with 0.01 apart. Although we have evaluated the performance of 26 chords of different lengths, we would like to show the results only for the chords from 11 to 20, because the performance of higher and smaller chords are relatively worse than the selected chords.

Figure 3 (a) - (g) show the average repeatability of SCCPDA detector for different chord lengths and different curvature thresholds against seven different transformations respectively mentioned in the previous section. Figure 3 (h) shows the average of average repeatability of all the transformations. We have not added the axis title to any of the graphs as the titles take much space and make the actual graph smaller. The vertical axis of the graphs shows the average repeatability and the horizontal axis shows the threshold values. Each curve shows the repeatability of a chord for different thresholds. From the average of the average repeatability of all transformations, we can see that chord 15 has the best average repeatability for threshold 0.22.

Similar to SCCPDA detector, the performance of CCSR detector has also been evaluated for different chord lengths and different thresholds against seven different transformations. The average repeatability of each chord is shown in Figure 4 (a)-(g). Figure 4 (h) shows the average of average repeatability of the seven transformations. The evaluation process has been conducted for the chords from 5 to 21 with 2 apart and from 0.979 to 0.990 with 0.001 apart. From the average of average repeatability in Figure 4 (h), we can see that the chord length 9 has the highest average repeatability for threshold 0.981.

Performance Evaluation

This section compares the average repeatability and localization error of proposed detectors with eight contour based detectors - DoGDetector [Xia09], CTAR [RMN11a], CPDA [Moh08], He_Yung [Xia04], MSCP [Xia07], ARCSS [Moh07], Eigenvalues [Dum99], GCM [Zha10]. Figures 5 and 6 show the average repatability and localization error respectively. Both of the figures are showing the result for seven different transformations along with the average values of the seven transformations.

In terms of average repeatability, except scale transformation, CCSR and SCCPDA have consistent better performance against different transformations. However, CPDA and MSCP have superior results in scale and shear transformations respectively. Considering all the transformations, CCSR has the best average of average repeatability followed by CTAR and SCCPDA.

In terms of Localization Errors, presented in Figure 6, the CCSR is not the best corner detector. Please note that, the lower localization error, the better the detector is. Except the shear transformation and gaussian noise, the original CPDA detector has the lowest localization error. However, similar to average repeatability, CCSR and SCCPDA detectors are always in the first four detectors in all the transformations. In summary, CPDA has the lowest average localization error against all the transformation followed by SCCPDA and CCSR detectors.

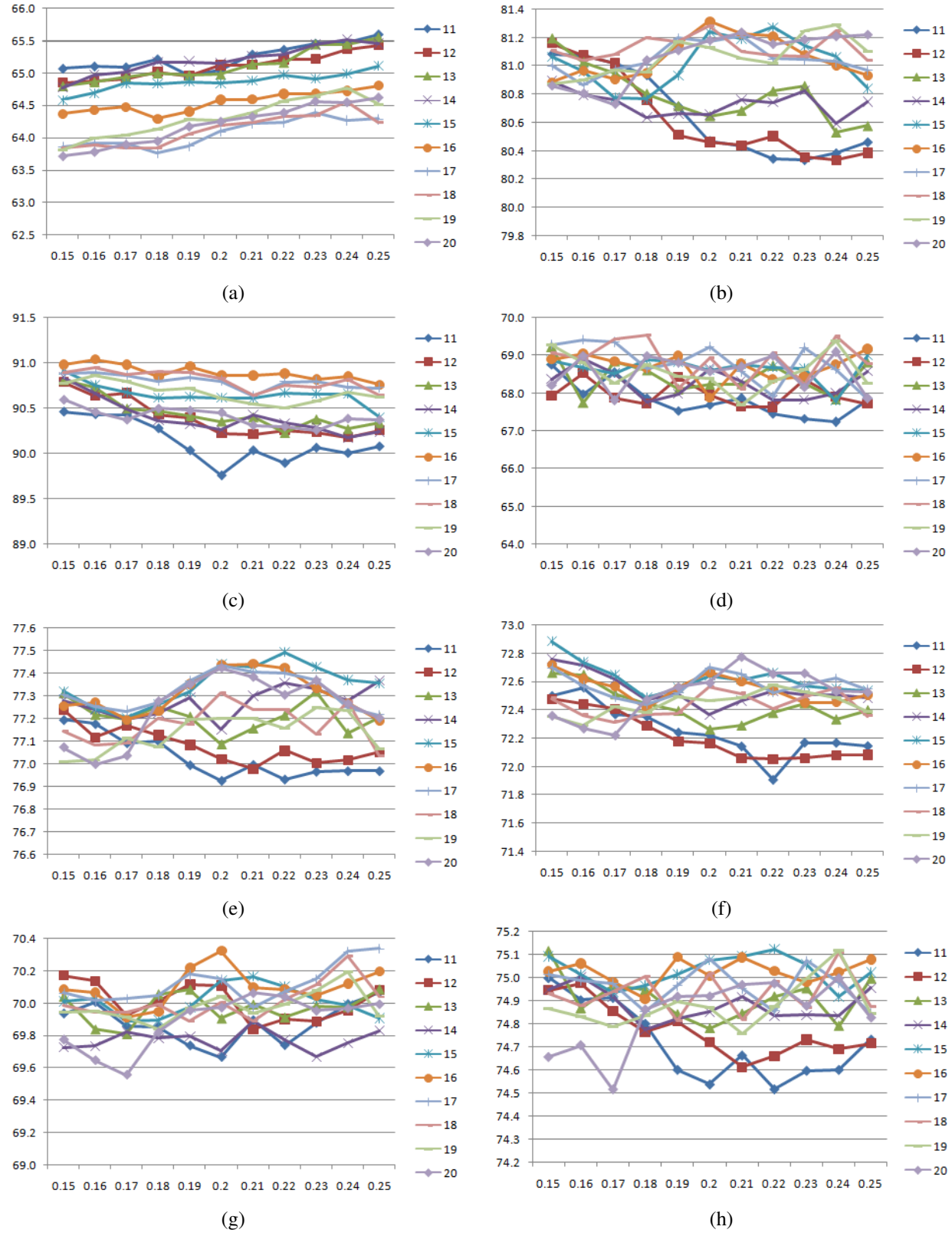


Figure 3: Average repeatability of SCCPDA detector for different chord lengths and different threshold against (a) scale transformation (b) rotation transformation (c) JPEG compression (d) Gaussian Noise (e) rotation+scale transformation (f) nonuniform transformation (g) shear transformation (h) average of all transformations

 ISBN 978-80-86943-65-7

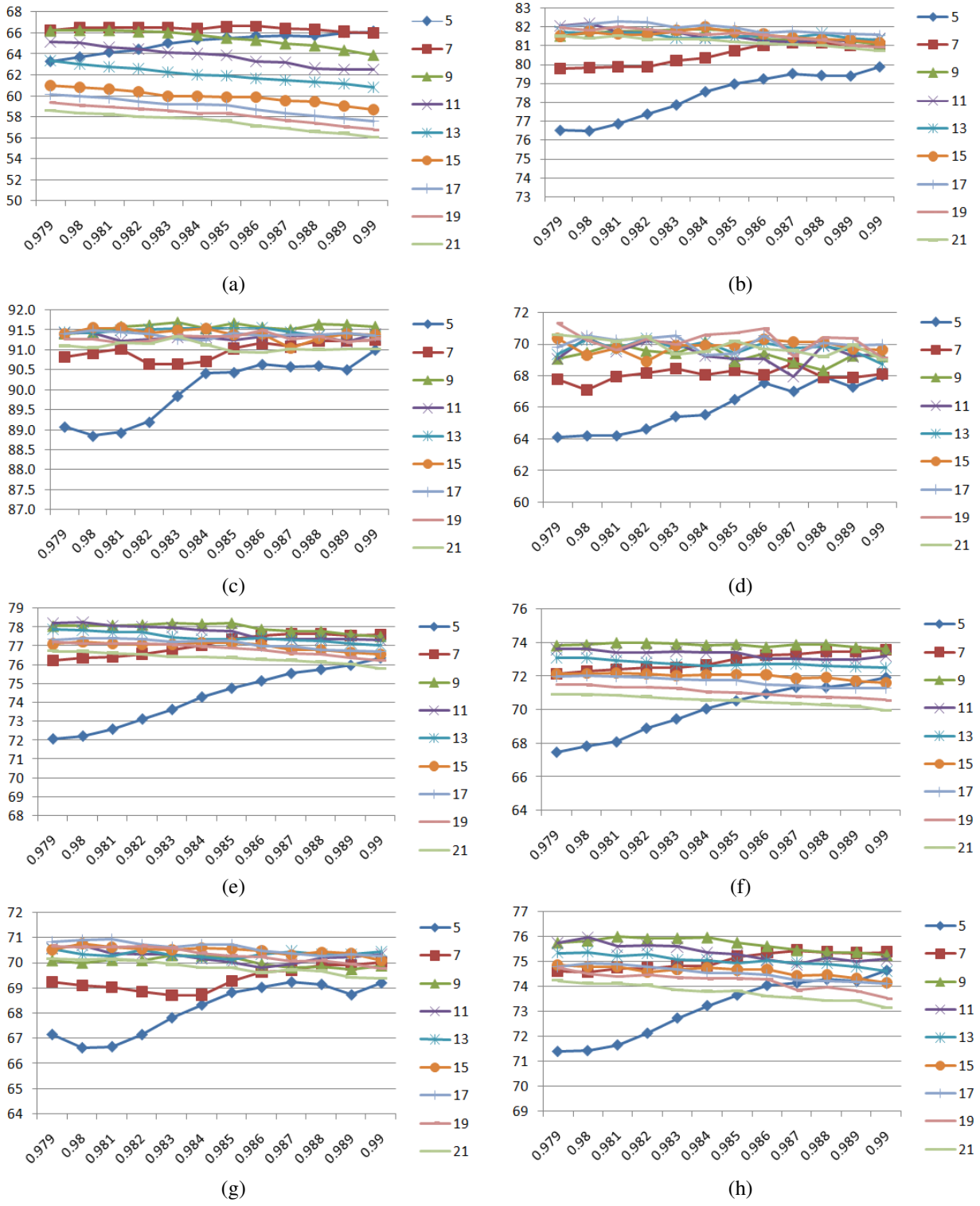


Figure 4: Average repeatability of CCSR detector for different chord lengths and different threshold against (a) scale transformation (b) rotation transformation (c) JPEG compression (d) Gaussian Noise (e) rotation+scale transformation (f) nonuniform transformation (g) shear transformation (h) average of all transformations

 ISBN 978-80-86943-65-7

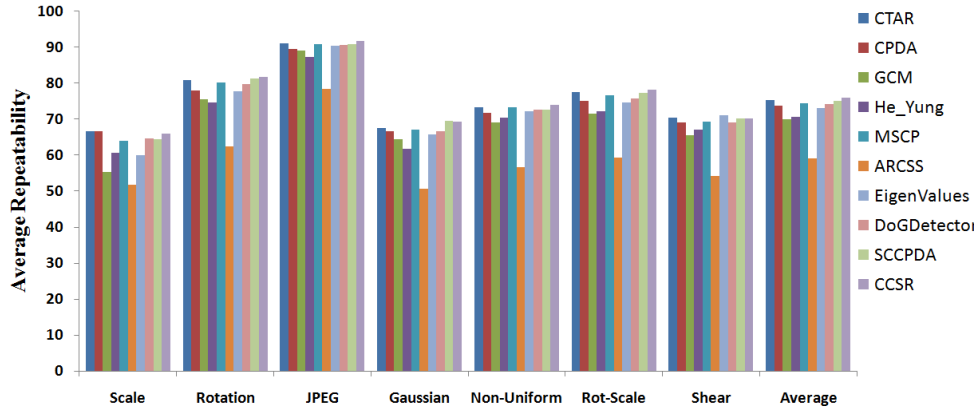


Figure 5: Average repeatability of corner detectors against different transformations

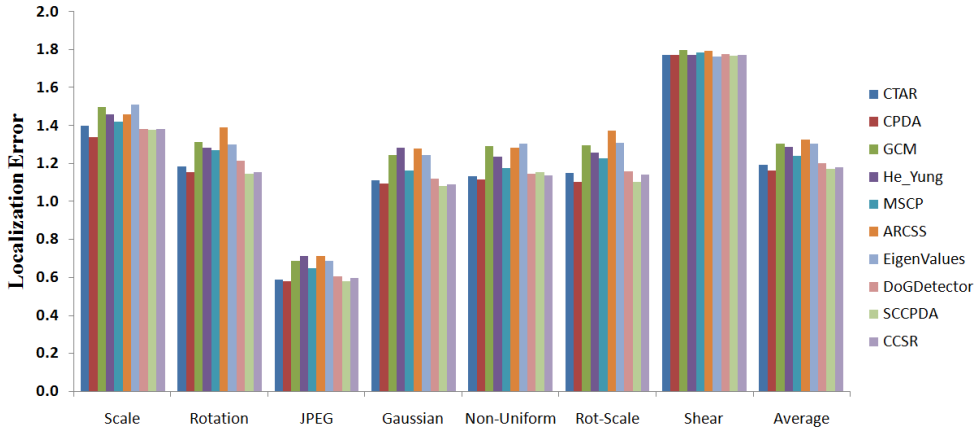


Figure 6: Localization Error of corner detectors against different transformations

Corner Detector	# of detected corners	% of CPDA corners
CPDA	922	100
SCCPDA	1000	91.398
CCSR	1264	92.437
DoGDetector	1342	88.152
CTAR	1364	94.557
ARCSS	1402	76.431
He_Yung	1617	88.167
EigenValues	1710	94.757
MSCP	1747	92.646
GCM	2714	94.878

Table 1: Number of corners detected by the detectors

Table 1 shows the number of corners in the base 23 images detected by the corner detectors. We see that CPDA detects the least number of corners followed by SCCPDA and CCSR detectors. However, both SCCPDA and CCSR detectors have better repeatability than CPDA detector (Figure 5). Higher repeatability means more corners from original image been detected in transformed image irrespective to the number of corners matched between original and transformed image. Therefore, a higher number of repeated corners with same repeatability has more advantages than less corners. Although other detectors detect more corners than CCSR, their repeatability is less than CCSR. We have also shown the percentage of CPDA corners of 23 base images detected by other corner detectors. Both SCCPDA and CCSR detectors are detecting more than 90% of the corners which are detected by CPDA detectors. Therefore, we can claim that the corners detected by these single chord detectors are qualitative and quantitative.

Complexity Comparison

Table 2 tabulates the total time taken to detect all the corners for the 23 base images by all the corner detectors. We see that CCSR is the most efficient corner detector followed by CTAR. Although SCCPDA detector is not one of the fastest corner detectors, it performs faster than the original CPDA detector. Note that, the process of extracting edges from the images has been discarded from the total time as this process is same for all the detectors. The time shown in the Table 2 is taken only to detect the corners from the extracted curves.

Corner Detector	Time(Sec)
CCSR	0.056090
CTAR	0.057176
Zhang	0.083072
MSCP	0.167033
He & Yung	0.310473
SCCPDA	0.310724
CPDA	0.384464
ARCSS	0.384809
DoGDetector	0.405147
EigenValues	1.078661

Table 2: Total time to detect corners from 23 images

6 CONCLUSION

In this paper, we have analyzed a state of art corner detector named CPDA and modified this detector in a way that the modified version not only has better average repeatability but also better efficiency. We have also proposed another contour-based corner detector, named CCSR, that outperforms CPDA and seven other existing corner detectors in term of average repeatability. CCSR also has very comparative average localization error. In addition, CCSR also detects more stable corners than CPDA detector. Another achievement of CCSR detector is that, CCSR is the fastest detector to detect corners among all the detectors compared in the experiments. In summary, discarding the use of multiple chords in CPDA detector increases not only the efficiency but also the average repeatability and CCSR, a single chord detector, is the most efficient corner detector having the best average repeatability.

7 REFERENCES

[Fmo01] F Mokhtarian and F Mohanna.*Enhancing the curvature scale space corner detector.*,pages O–M4B, 2001.

[Fan88] Fang-Hsuan and Wen-Hsing *Parallel algorithm for corner finding on digital curves.*,Pattern Recognition Letters, 8(1):47–53, 1988.

[Moh07] Mohammad Awrangjeb and Guojun Lu.*A robust corner matching technique.*, Multimedia and Expo,2007 IEEE International Conference on, pages 1483–1486. IEEE, 2007.

[Pen13] Peng-Lang Shui and Wei-Chuan Zhang.*Corner detection and classification using anisotropic directional derivative representations.*,Image Processing, IEEE Transactions, 22(8):3204–3218,2013.

[Moh08] Mohammad Awrangjeb and Guojun Lu.*Robust image corner detection based on the chord-to-point distance accumulation technique.*, Multimedia, IEEE Transactions, 10(6):1059–1072, 2008.

[Moh10] Mohammad Awrangjeb, Guojun Lu, and Clive S Fraser.*A comparative study on contour-based corner detectors.*, Digital Image Computing: Techniques and Applications (DICTA),2010.

[Can86] John Canny.*A computational approach to edge detection*,Pattern Analysis and Machine Intelligence, IEEE Transactions,(6):679–698, 1986.

[Dav80] David Marr and Ellen Hildreth.*Theory of edge detection.*,Proceedings of the Royal Society of London. Series B. Biological Sciences, 207(1167):187–217, 1980.

[Xia07] Xiaohong Zhang, Ming Lei, Dan Yang, Yuzhu Wang, and Litao Ma.*Multi-scale curvature product for robust image corner detection in curvature scale space.*,Pattern Recognition Letter, 28(5):545–554, 2007.

[Dmi99] Dmitry Chetverikov and Zsolt Szabo.*A simple and efficient algorithm for detection of high curvature points in planar curves.*,1999.

[RMN11a] RM Najmus Sadat, Shyh Wei Teng, and Guojun Lu.*An effective and efficient contour-based corner detector using simple triangular theory.*,Pacific Graphics Short Papers,The Eurographics Association, 2011.

[RMN11b] RMN Sadat, Zinat Sayeeda, MM Salehin, and Naurin Afrin.*A corner detection method using angle accumulation.*,Computer and Information Technology (ICCIT), 2011 14th International Conference,pages 95–99. IEEE, 2011.

[Xia04] Xiao Chen He and Nelson HC Yung.*Curvature scale space corner detector with adaptive threshold and dynamic region of support.*,Pattern Recognition,Proceedings of the 17th International Conference, IEEE, 2004.

[Moh12] Mohammad Awrangjeb, Guojun Lu, and Clive S Fraser.*Performance comparisons of contour-based corner detectors.*,Image Processing, IEEE Transactions, 21(9):4167–4179, 2012.

[Xia09] Xiaohong Zhang, Honxing Wang, Mingjian Hong, Ling Xu, Dan Yang, and Brian C Lovell.*Robust image corner detection based on scale evolution difference of planar curves.*,Pattern Recognition Letters, 2009.

[Dum99] Du-Ming Tsai, H-T Hou and H-J Su.*Boundary-based corner detection using eigenvalues of covariance matrices.*,Pattern Recognition Letters, 1999.

[Zha10] Zhang X., Wang H., Smith A., Ling X.,Lovell B. C., Yang D. *Corner detection based on gradient correlation matrices of planar curve* Pattern Recognition 43, April 2010.

GeoPeels: Deformation-Based Technique for Exploration of Geo-Referenced Networks

Alberto Debiasi
Fondazione Graphitech
Via alla Cascata 56/C
38123, Trento, Italy
alberto.debiasi@graphitech.it

Bruno Simões
Fondazione Graphitech
Via alla Cascata 56/C
38123, Trento, Italy
bruno.simoes@graphitech.it

Raffaele De Amicis
Fondazione Graphitech
Via alla Cascata 56/C
38123, Trento, Italy
raffaele.de.amicis@graphitech.it

ABSTRACT

Spatial relations between geographical entities are typically visualized as a node-link diagram that is depicted over a geographical layout. The node-link diagram representation is well-suitable for tasks that require the analysis of the topology and properties of a spatial graph because it is more intuitive with respect to the matrix representation. However, the readability of node-link diagrams is, especially for large sized graphs, one of the issues studied in literature. In this paper we describe an interactive technique for virtual globes, called *GeoPeels*, which uses spatial deformations to reveal spatial information otherwise hidden behind the geographical layout. The deformed geometries are rendered around the globe, therefore, occupying space that otherwise would be unused. Additionally, our approach facilitates graph-related tasks by reducing the overall number of actions required to visualize and understand the spatial relations. The evaluation of our approach is based on the analysis of the graphical performance and on a comparison between virtual globes and 2D flat maps.

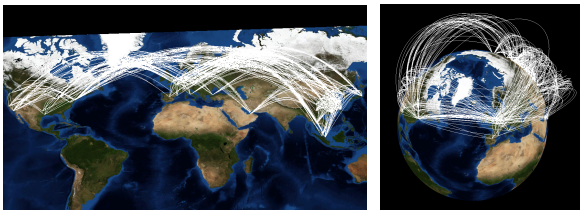
Keywords

3D geovisualization, edge congestion, interactive visualization, graph layout, geographical distortion, geo-referenced data.

1 INTRODUCTION

Large-scale map visualization systems of geographical datasets retain their traditional role of providing insight into geospatial patterns and relations, see Figure 1. On the one hand, they provide geospatial contextualization for each data point. On the other hand, they can be used to enrich data points with graphical elements like straight lines, circular arcs and curves, to give emphasis to specific spatial relations.

Virtual globes are considered mature visualization systems for the exploration of geo-referenced networks. In the past, they were used by Cox et al. [9] to visualize Internet traffic over the NFSNET/ANSnet backbone and by Munzenr et al. [27] to map the MBone routers. In climate research domain, the analysis of climate data as complex network theory is considered a powerful approach [13]. In such context the globe is commonly used [35, 29]. Globes are also used in frameworks for management and monitoring of large scale distributed systems [23].



(a) (b)

Figure 1: Geo-referenced network over imposed (a) on a flat map, and (b) on a virtual globe.

In literature, many approaches have been proposed to improve the visualization during network analysis. Brandes et al. [5] minimize clutter at important nodes by adjusting the angle of edges. A few techniques [22, 10] were provided to bundle edges around the globe. Alper et al. [1] present a visual technique to better understand the relations of the network. The globe is deformed to place the nodes connected with each other to appear closer. Debiasi et al. [11] present 3DArcLens, a focus+context technique to reveal areas occluded by undesired edges.

It is possible, through visual exploration, to interactively browse different portions of a dataset to better understand the data. On the one hand, visual exploration helps with hypothesis confirmation. On the other hand, it facilitates the discover of unexpected patterns, which can raise new questions, without the requirement

of having a complete knowledge about the data [37]. However, in virtual globes, we can visualize only one of the hemispheres, i.e. the globe occludes the nodes and the links that can be considered important for exploratory tasks.

In this paper, we propose *GeoPeels*, a distortion-based technique for the visualization of spatial relations in geographical data, where such interactive techniques remain unexplored. In particular, our technique is optimized for graphs-related tasks, such as explore node connections. Our approach is based on a virtual globe that automatically deforms the terrain surface to avoid the occlusion of key data nodes and links, i.e. based on the camera view position, the focus area, and relevant nodes and links. The terrain deformation is automatically adjusted to avoid rendering the geometries outside the screen view. In this way relevant nodes and connections are always visible.

In the exploration of spatial relations, our technique can be used in combination to zoom and pan, which by themselves are considered tedious, potentially disorienting, and often ineffective [16, 17]. Furthermore, *GeoPeels* does not require the operator to define visual node-link attributes, e.g. we do not use attributes like color and size of a node. Hence, these unused node-link attributes can be used to encode additional information. Consequently, our technique can be combined with other works that make use of such visual attributes, for example, to reduce the visual clutter.

This paper is structured as follows. Section 2 describes previous related work. Section 3 introduces our approach. Section 4 explains the list of parameters as well how to use them. Section 5 describes a use case that uses our technique. Section 6 provides an analysis of the graphical performance. Section 7 describes the advantages of our approach and makes a comparison between virtual globes and 2D flat maps. We conclude with some remarks about the usability of the proposed technique and future work in Section 8.

2 RELATED WORK

Visualizations of Geo-referenced Networks. There are two main visualization techniques to represent geo-referenced networks: the matrix-based representation and the node-link layout. Although the first visualization is free from occlusion, the latter allows layouts to encode the geographical information of nodes. Geographic node-link layouts can be classified in three categories. In the first category, both data and maps are depicted over a flat surface. The first system to automatically generate 2D visualization with arrows imposed on a map was presented by Tobler et. al. [33]. ArcMap [9] applies the same concept on a 3D space, where 3D arcs connect nodes on flat surface. In the second category everything is projected over a spherical (or ellipsoidal) surface [9, 27, 6, 23]; and in the third category we have a combination of the previous ideas depicted in multiple views [4, 8]. VisLink [8] links two or more maps using straight lines. Flowstrates [4] uses a map for the origins and another one for the destinations to represent spatio-temporal data. For an overview of visualizations used to depict geo-referenced networks see [31, 40].

Navigation to elements occluded by geographical surface. In all aforementioned categories the nodes and links may have visibility problems that may affect the exploration of the network. In 3D visualizations, where the geographical surface is a volume, graph elements may be occluded. Different focus+context techniques can be implemented to alleviate this problem. Magic Lenses [2] are lenses that are able to modify the rendering of objects seen through them. The use of these lenses was also extended to 3D graphics [39, 32]. Cignoni et al. [7] presented MagicSphere, a magic lens metaphor to volumetric lenses. Cut-away viewing [30, 12], is another approach to reveal what is occluded by volumetric objects. Exploded views have been investigated in the context of architectural visualization by Niedauer et al. [28] and for volume visualization by McGuffin et al. [25]. Their approach facilitates interactive browsing of volume data partitioned into several layers. Tory et al. [36] provide a framework for combined visualization of context and detail in volumetric data sets. For a comprehensive overview of techniques for the occlusion managements see the survey by Elmqvist et al. [15].

Navigation to Off-screen elements. Large graphs can not be entirely visualized in the screen view. However, many approaches are available in literature to overcome such problem. Pan in 2D layouts and the camera view relocation in 3D layouts are traditional tools to reveal off-screen elements. More recent techniques addressed such problem, both in terms of their visibility [18] and in navigating to them [19]. They provide visual indication of locations of off-screen objects using a simple representation at the viewport's edge. Also, they provide mechanisms to select items of interest, and to navigate to them quickly. With Bring Neighbors Lens [34], when a user moves the lens (using the mouse) towards a vertex, the neighboring vertices should be brought to the lens in a continuous fashion. To guarantee the visibility of multiple focus regions, Elmqvist et al. [14] provide a distortion technique that folds the intervening space. The context is preserved of the folds themselves.

Map Projections. The performance of the exploration of geo-referenced networks is affected by the projection that is used. The major problem of map projection is the distortion. For example, the Mercator projection, which was originally designed to display compass bearings for sea travel, depicts distortions for large areas. Azimuthal projections allows the directions to

be correct from the center of the projection to any other point. Moreover, great circles of the globe are projected to straight lines. The azimuthal orthographic projection is mainly used for illustration purposes, since it shows the Earth as seen from space infinitely far away. The availability of high-quality imagery of the Earth and other planets as seen from orbit, has caused a recent revival of interest for this projection. Virtual globes use such projection enabling a wide variety of interactive pan and zoom operations. In Azimuthal equidistant projection the distances and the bearing from the center of the map to any point are correct. The projection is frequently used to show air-route distances. The Lambert azimuthal equal-area projection preserves areas while simultaneously maintaining a true direction from the center. Recommended to the European Commission for statistical analysis and display. The latter two projections cover the entire world. Van Wijk presented Myriahedral projections [38], a class of methods for mapping the earth on a polyhedron with a large number of faces. The polyhedron is then cut open and unfolded. In such way the maps are (almost) conformal and conserve areas, but they are characterized by a large number of discontinuities. In literature, different works change dynamically the map projection [20, 3, 21]. For navigation purposes, Möser et. al. [26] propose a technique that blends geo and egocentric perspectives to combine the advantages of both viewpoints while maintaining the ease of perception of a single view. Lorenz et. al. [24] implement an interactive multi-perspective views for 3D geographic environments. It automatically configures the deformation in a view-dependent way to maintain the multi-perspective view in an interactive environment.

3 PROPOSED APPROACH

In this work we define a deformation-based technique that enables the interactive exploration of spatial relations between geographical entities; in particular to facilitate graph-related tasks, e.g. exploring the node connections.

We propose a technique that enriches the idea of a virtual globe with the following features: a list of *important nodes*, i.e. nodes that are connected to nodes *in focus* should be always occlusion-free. We define *in focus* the list of nodes that are inside the user-defined area. Remaining nodes are said to be *in context*, see Figure 2.

In order to relocate nodes *in focus* that are occluded by the globe's hemisphere, without losing their background map context, we defined the concept of *slice*, which corresponds to a portion of terrain that is occluded by the globe and that contains at least one *important node*.

In the next sections, we use the terminology *deformed slice* to describe the geometry of a slice that was peeled

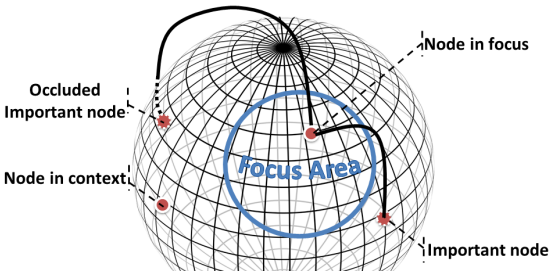


Figure 2: Nodes are classified accordingly with their position and the position of their connected nodes.

from the globe surface – which is represented as a continuous concave surface with respect to the camera view. The concept of *deformed slice* can be demonstrated with an orange: we can peel an orange to reveal its hidden parts, see Figure 3.

This approach keeps the globe's properties, i.e. nodes in the hidden hemisphere that are not *important* are intrinsically filtered out, even if located in the deformed slice.

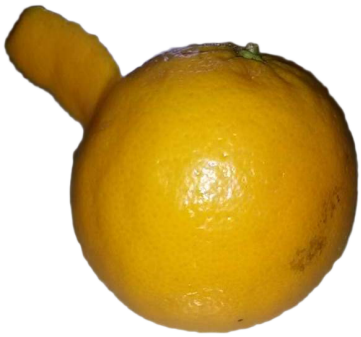


Figure 3: The orange peel is the metaphor that inspired this work.

The process to create and deform a slice is transparent to the operator. It entails the creation of *slices* for *important nodes* that are not visible. The first node that is added to a *slice* is called its *primary node*. Slices are created only if an occluded *important node* cannot be moved into an existent slice, see Figure 4. This strategy ensures that nodes are never occluded by the globe and the number of *deformed slices* is minimized.

The shape of each *slice* is defined as a rectangular geometry with the *primary node* located at the center with respect to the horizontal axis, and at a fixed position with respect to the vertical axis. During the interactive exploration, the *deformed slice* is bended to ensure the visibility of the geometry within the screen view, see Figure 5.

The focus area is represented as a circle in screen space with a radius r in pixels. The circular element remains still during the user navigation, e.g. tilt, pan and zoom,

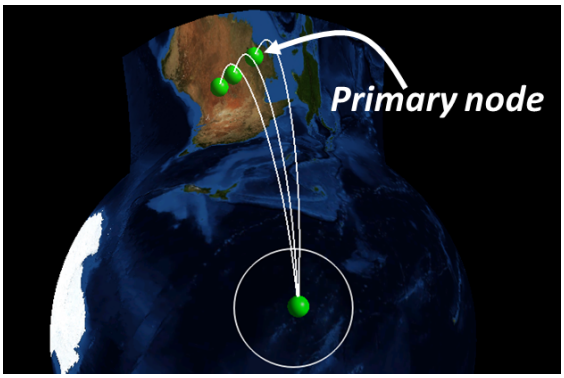


Figure 4: In this example the *deformed slice* has three nodes: the *primary node* and two nodes assigned to the existing *slice*.

hence revealing the hidden information that was initially covered by the circle. In fact, the circle is snapped to the terrain to improve the system usability; the camera movement affects the circle position in screen coordinates but not its geographical position.

GeoPeels was developed on top of *NASA WorldWind*, an open source virtual globe that approximates the earth as an ellipsoid. The proposed technique makes use of a reference system, which is described in Subsection 3.1. The overall procedure of *GeoPeels* is summarized in Subsection 3.2. In Subsection 3.3, we describe how nodes are relocated if there are multiple *slices* as candidates. Finally, in Subsection 3.4, we describe the function to manage the surface deformation. The parameters introduced in this section are described in Section 4.

3.1 Slice Reference System

The algorithm to check whether a node is inside a slice, to relocate a node, and to create a slice is based on a reference system R_S, specific to a *slice* S. R_S is defined by the planes P_d and P_α. The first plane, P_d, is identified by three points: the node n that corresponds to the point we want to reveal, the center of the earth o, and c that is the camera view point l projected over the globe, see Figure 6(a).

For every point p on the globe, γ defines the angle between the vector lp and the normal of the globe surface on p. If $\gamma > \pi/2$ then the point p is visible, otherwise it is occluded by the globe. Thus, we define x as the point on the great circle, between c and n with angle $\gamma = (\pi/2) + 0.4$. Consider a second plane, P_α, to be perpendicular to P_d and to intersect o and x, see Figure 6(b). Then, we have that the position of p is defined by $R_S(\alpha, d)$, where d is the perpendicular distance between p and P_d, and α is the angle between the plane P_α and the plane perpendicular to P_d, which intersects the points o and p. See, respectively, Figure 6(c) and Figure 6(d).

3.2 GeoPeels

Our approach can be summarized with the following procedure, which is executed every frame before the rendering phase:

```
procedure GeoPeels
    (1) List< Node > N := getNodesToRelocate();
    List< Slice > S := ∅;
    for each n ∈ N do
        (3) bool hasSlice := hasCompatibleSlice(n, S)
        if hasSlice = false then
            (2) Slice s := createSlice(n)
            S.add(s)
        end if
    end for
    for each n ∈ N do
        (3) Slice s := getChosenCompatibleSlice(n, S)
        (4) relocateNode(n, s)
    end for
    (5) drawArcs()
    for each s ∈ S do
        (4) deformSlice(s)
    end for
end procedure
```

(1) Retrieve the list of important nodes to relocate. The list of important nodes to relocate, is computed by checking their angle γ; a node is added to the list if $\gamma < (\pi/2) + 0.4$. The list considers the occluded nodes together with the nodes that are placed near to the occluded hemisphere. Afterwards, the list is sorted in descending order accordingly to each node distance to the camera view projected over the globe. This operation is useful to minimize the number of *slices*.

(2) Slice creation procedure. Let $R_S(\alpha', 0)$ be the position of a primary node, and w and k the parameters to define respectively the width and the length of its *slice*. Then, the *slice* is simply the set of all the points $R_S(\alpha, d)$ such that $d > -w$, $d < w$, $\alpha > 0$, and $\alpha < \alpha_S$, where $\alpha_S = \alpha' \cdot k$, see Figure 7(a).

(3) Retrieve compatible slices for a given node. A *slice* is compatible with a node if the latter is inside the geographical area over the globe to be deformed, see Figure 7(b). If we have more than a compatible slice for a node, then we need to use a strategy to choose the best *slice*. Such strategy is described in Subsection 3.3.

(4) Deformation of a slice and relocation of a node. Each *deformed slice* is a grid of points where we map the texture of the globe surface that corresponds to the area of interest. Both, the nodes and the grid points are relocated over the globe and consequently in altitude using the relocation function, explained in details in Subsection 3.4.

(5) Arc creation procedure. Arcs are splines composed of a sequence of control points along the great circle

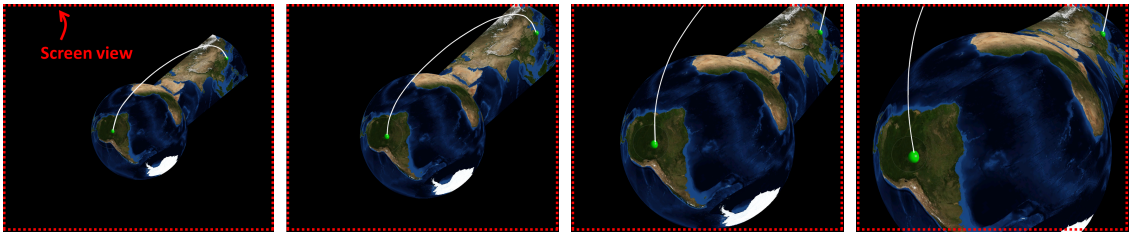


Figure 5: The *deformed slice* is bended to fit inside the screen view.

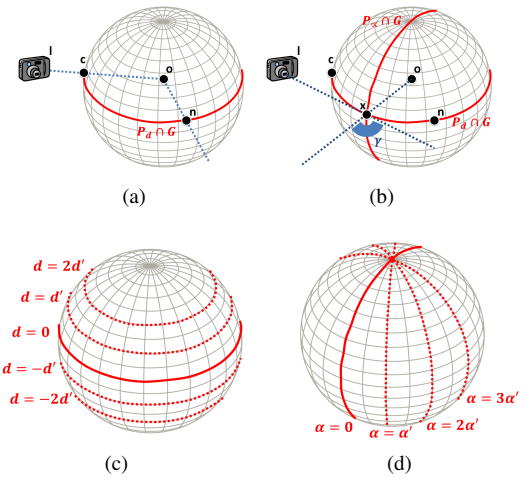


Figure 6: The planes P_α, P_d are used to define the reference system for each *slice*. α' and d' are values arbitrarily defined. Red lines are the intersection between the globe G and the planes.

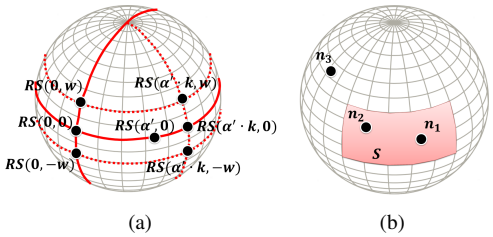


Figure 7: (a) Points used to define the *slice*. (b) n_1 is the primary node of S. S is compatible for n_2 but not for n_3.

that connects their two endpoints. Each control point has the following altitude:

$$h = 1 + h_{max} sin(\pi t) \quad (1)$$

The parameter t ranges from 0 to 1, along the arc path. h_{max} is equal to a quarter of the distance between the endpoints. Arcs that connect nodes that were relocated are composed of control points with the following altitudes:

$$h^* = h + (h_s + (h_e - h_s)t) \quad (2)$$

The parameters h_e and h_s are respectively the altitudes of the two endpoints e and s. The altitude is applied with respect to the normal vector of the globe surface, in the point along the great circle. As shown in Figure 8, arcs that connect relocated nodes maintain an aesthetically pleasing shape.

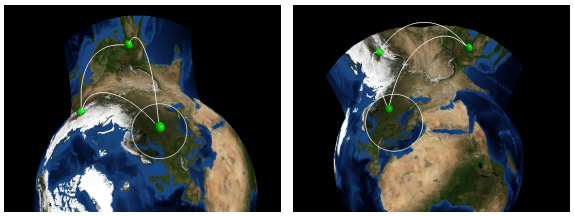


Figure 8: Examples of different arcs: arcs connecting nodes over the globe and arcs connecting relocated nodes.

3.3 Slice Selection Function

Deformed slices can occlude arcs, nodes and other slices. It is important to minimize the number of deformed slices as well as to ensure that each node is not occluded by its *compatible slices*. In this section, we describe the approach that satisfy these requirements.

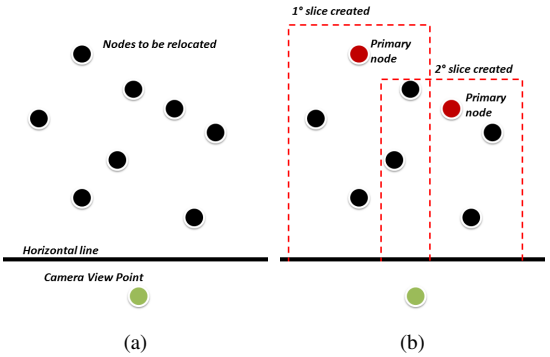


Figure 9: Our strategy to create slices and to choose the best one for a node.

Figure 9(a) shows our strategy in a projected 2D plane, for simplicity. The steps are the following:

The nodes are selected in descending order with respect to the distance with the camera view position projected over the globe. If the node has a compatible slice then

it is marked as *to relocate*. Otherwise, we create the *slice* and the node became its *primary node*, see red nodes in Figure 9(b). In a second round, we relocate the nodes. If a node has more than one compatible slice, then we choose the last created slice, because it is the most visible.

3.4 Function to Relocate Points

In this section, we describe how we relocate a point from $R_S(\alpha,d)$ to $R_S(\beta,d)$. Such point can be a node or a grid point of the deformed slice.

First, we defined the altitude of the point to relocate with the following formula:

$$h' = 1 + h'_{max} sin(\pi((\alpha-\beta)/2\alpha)) \quad (3)$$

where h'_{max} is equal to the great-circle distance between the original point and the point $R_S(0,d)$. Moreover, for a given point $R_S(\alpha,d)$ that is relocated to $R_S(\beta,d)$, the corresponding vector to move the point in altitude, is the normal vector of the surface globe at the point $R_S(\beta,0)$, see Figure 10(a). Thus, the deformed slice maintains the same width for every curvature.

The angle β is calculated with the following formula:

$$\beta = -2\alpha f(v) + \alpha \quad (4)$$

where $f(v) \in [0,1]$. The function $f(v)$ allow us to relocate points over a circular arc, see Figure 10(b). v is defined for each slice. If $f(v)$ is the identity function, the *deformed slices* will be created as shown in Figure 10(c). However, the nodes relocated on the upper area of the deformed slice may be occluded by the surface itself. For such reason, as shown in Figure 10(d), we update it as follows:

$$f(v) = v(\alpha/\alpha_S)^2 \quad (5)$$

The parameter v is initially set to 1. However, if the *primary node* is outside a defined *sub-region* of the screen view, v is decremented until the *primary node* is considered visible (i.e. inside the *sub-region*), see Figure 11.

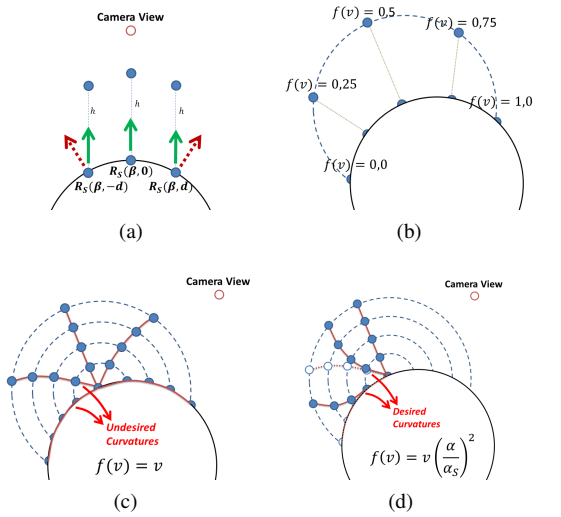


Figure 10: (a) Red arrows represent normal vectors over the globe surface. Green arrows represent the vectors used to move the points. (b) Each point is relocated over an imaginary arc. (c) The curvatures of a *deformed slice* modeled with the identity function, (d) and with our improved function.

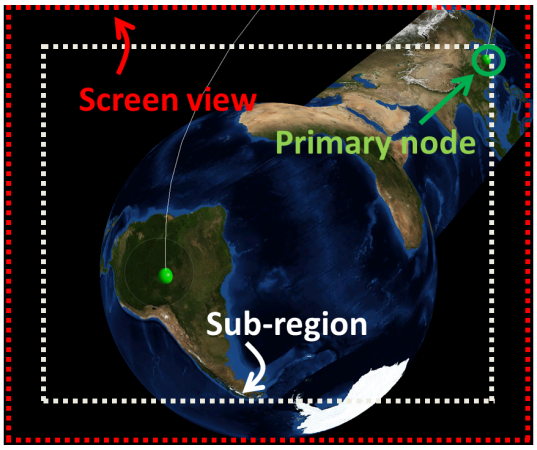


Figure 11: The *deformed slice* is bended until the primary node is inside the *sub-region* of the screen view.

4 INPUT PARAMETERS

The list of user customizable parameters is the following.

1. r is the radius of the circle, expressed in pixels. We recommend the use of a small radius if the dataset contains many nodes in the circumscribed areas, in order to reduce the number of nodes that have to be relocated. The default value is 150 *pixels*, which can be interactively changed by the user.

2. X and Y are used to define the number of points that compose the grid of a *slice*. We do not need a dense grid to give the impression of a curved surface. Hence, heuristically we set $X,Y = 7$.

3. w defines the half width of the slice. The default value is 3.500 *km* that is a good balance between area to visualize and the possibility to intersect other slices.

4. k defines the position of the relocated node with respect to the y-axis of the deformed slice. The default value is 1.2. It allows the near surface of the node to be visible. However, if the area is too circumscribed, the cognitive process to identify the location of the node can be problematic for the operator.

5. The dimension of the *sub-region* defines whether a node is considered visible or not. We set such parameter to be equal to 200 *pixels* less than the height and the width of the screen view. Since our strategy to decide the visibility of the nodes that are outside the screen view takes into account only the primary node of the *deformed slice*, other solutions can be adopted. For example, the visibility check can be performed on the two upper vertices of the geometry, and not on the *primary node*.

5 USE CASE

We used a dataset describing the connections between the main airports[1], composed of 50 nodes and 677 edges. The dataset is composed of groups of near nodes connected with long distance nodes. Hence, the circle element is appropriate to explore such dataset, see Figure 12. Due to the high arcs density, the links connected with the nodes in focus are colored in red and the remaining are colored in semi-transparent white, see Figure 13. With this combination, the *important nodes* together with their connected arcs can be easily identified.

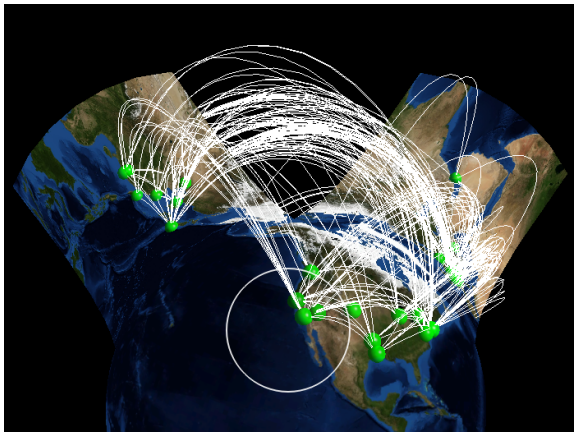


Figure 12: Airport connections dataset explored using *GeoPeels*.

6 EVALUATION

To measure the performance of our technique we take as metric the frame rate. *GeoPeels* was tested on a single processor Intel Xeon 2.26 GHz equipped with an NVIDIA GeForce GTX 280 with 1024 MB dedicated video memory.

Table 1 shows the *fps* (frame rate per second) of *GeoPeels* taking into account two datasets respectively with 154 and 4777 nodes. *#def. nodes* indicates the number of nodes that are relocated and *#slices* indicates the number of slices that are created. In this benchmark we used the values defined in Subsection 4.

[1] http://openflights.org

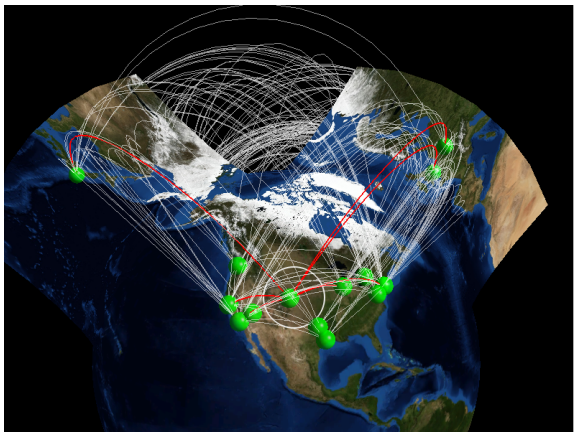


Figure 13: *GeoPeels* in combination with a color technique.

Our results take into account, for each frame, the time that is required to generate the grids and to render the geometries. In our tests we used spheres over the terrain to represent the nodes.

Table 1: Performance of *GeoPeels* for different datasets.

#nodes	#def. nodes	#slices	fps
154	0	0	120
	1	1	54
	4	1	50
	11	2	34
	9	3	28
	11	3	26
	7	4	23
4777	0	0	16
	2	1	14
	1	1	13
	6	2	12
	9	3	11
	3	3	11
	15	2	10
	22	5	8

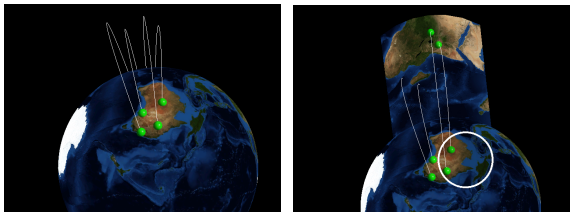
After analyzing the evaluation results, we noticed that the number of created slices and the number of relocated nodes have a considerable impact on the number of *fps*. However, for datasets with hundreds of nodes our approach gives acceptable performance.

7 DISCUSSION

In this section, we summarize our findings and the overall strength and weakness of our technique.

Advantages with respect to other virtual globes. An advantage of the proposed technique is that nodes that are considered important for a specific analysis are

never occluded by the surrounding map, see Figure 14. Additionally, we present an alternative solution to the approach of manually moving the camera view, which is a time consuming operation and may cause the loss of a previously established visual context.



(a) Virtual Globe. (b) GeoPeels.

Figure 14: Nodes occluded behind the globe.

Advantages with respect to 2D flat maps. The limitation of 2D flat maps is on their spatial boundaries. It is difficult to get insight about nodes located on the borders of the map, especially on nodes that connect with nodes on the opposite side of the map, see Figure 15. Moreover the zooming out can make difficult the analysis of the spatial relations; the visual clutter caused by unoptimized long arcs can cause visualization problems. On possible solution is a 2D map projected over an imaginary infinite plan. However, such strategy may cause redundant information related to the nodes and the arcs.

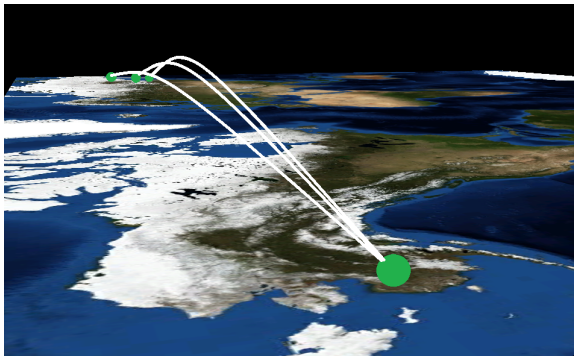


Figure 15: Fake sense of distance. These points are in reality quite close to each other.

Limitations. We identified few elements that require further investigation. First, some visual elements can be misleading for the operator because only nodes connected to nodes in the focus area are physically moved. But, our goal is to put more emphasis on a pre-selected data property, e.g. set of nodes and links.

Second, it might happen that arcs connecting visible nodes and nodes occluded by the globe's hemisphere intersect the newly deformed surfaces, e.g. when an occluded node is located behind the surface. Hence, it can mislead the operator to think that arcs end in the new deformed surface. However, this aspect can be alleviated by applying a level of transparency on the texture of each deformed geometry, see Figure 16, or applying a different color to the arcs that end over the deformed surface, see Figure 13.

Third, in a classical virtual globe approach, we can look at the arcs shape to gain insight on the distance between its endpoints - farther the endpoints are higher the arc is. In our approach, this property does not hold in general.

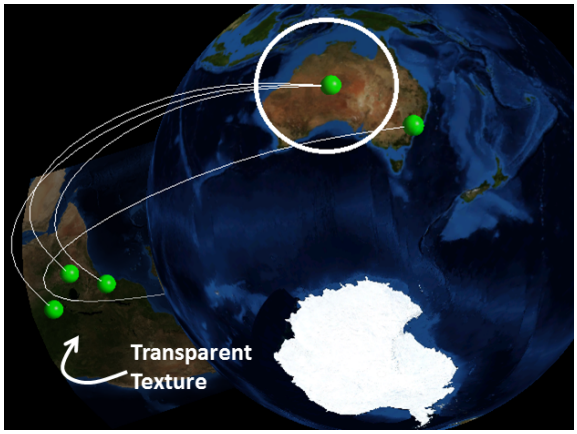


Figure 16: A transparent deformed slice. Hence, it is possible to keep trace of its intersecting arcs.

Four, as far as it concerns other deformation-based approaches, the main problem is the cognitive process associated with the identification of the deformed area. To improve this process, we can relocate together with the nodes auxiliary information, e.g. map labels.

Finally, *GeoPeels* is a deformation-based technique that requires the user to take care of the circle object.

Extensions. To improve the effectiveness of *GeoPeels*, it can be redesigned in terms of interaction. For example, it introduces an user defined selection of the nodes *in focus* as an alternative to the proximity-based selection of the circle element, see Figure 17.

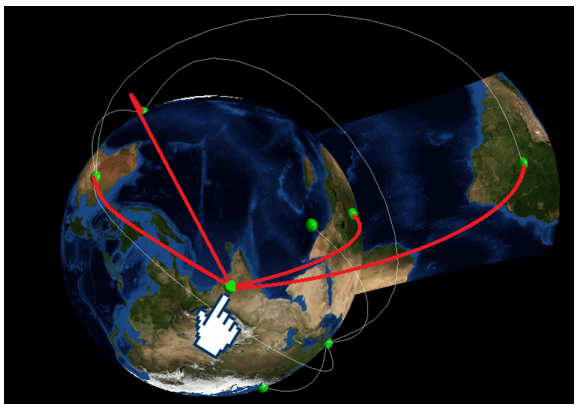


Figure 17: Extension based on node selection.

8 CONCLUSIONS AND FUTURE WORK

In this work, we have described an interactive deformation-based technique that capitalizes on dy-

namic globe deformations for an effective exploratory visual analysis of geographic networks.

Our technique provides an alternative approach to relocation of the camera view, which is a time consuming operation and may cause the loss of a previously established visual context. Another advantage of our technique is that it does not require the operator to define visual node-link attributes, e.g. we do not use attributes like color and size of a node, hence, they can be used to encode additional information.

Consequently, our technique can be combined with other works that make use of such visual attributes, for example, to reduce the visual clutter or combined with focus+context techniques applied on virtual globes, e.g. *important nodes* can be colored with red and the rest in semi-transparent white.

As future work, we plan to empirically measure the performance of the presented technique on a more comprehensive set of tasks. Moreover, a comparison has to be made with other approaches that are able to solve the raised issues. We also want to extend the orange peeling metaphor to other contexts such as the visualization of points of interest represented as icons and 3D models, or even to more complex scenarios such as the visualization of moving objects in real-time.

9 ACKNOWLEDGEMENTS

GeoPeels was applied to visually explore a georeferenced network describing the wood chain. Such data was collected from the SLOPE project.

The work presented in this paper has received funding from the EC through the 7th Framework Programme under the Grant Agreement n. 604129 (project "SLOPE"). The authors are solely responsible for this work which does not represent the opinion of the EC. The EC is not responsible for any use that might be made of information contained in this paper.

10 REFERENCES

[1] B. Alper. *Dynamic visualization of geographic networks using surface deformations*. PhD thesis, Citeseer, 2006.

[2] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and magic lenses: the see-through interface. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 73–80. ACM, 1993.

[3] J. Böttger, M. Preiser, M. Balzer, and O. Deussen. Detail-in-context visualization for satellite imagery. In *Computer Graphics Forum*, volume 27, pages 587–596. Wiley Online Library, 2008.

[4] I. Boyandin, E. Bertini, P. Bak, and D. Lalanne. Flowstrates: An approach for visual exploration of temporal origin-destination data. In *Computer Graphics Forum*, volume 30, pages 971–980. Wiley Online Library, 2011.

[5] U. Brandes, G. Shubina, and R. Tamassia. *Improving angular resolution in visualizations of geographic networks*. Springer, 2000.

[6] S. Buschmann, T. Nocke, C. Tominski, and J. Döllner. Towards visualizing geo-referenced climate networks. In *Proceedings of Workshop GeoViz Hamburg 2013*, 0 2013.

[7] P. Cignoni, C. Montani, and R. Scopigno. Magicsphere: an insight tool for 3d data visualization. In *Computer Graphics Forum*, volume 13, pages 317–328. Wiley Online Library, 1994.

[8] C. Collins and S. Carpendale. Vislink: Revealing relationships amongst visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1192–1199, 2007.

[9] K. C. Cox, S. G. Eick, and T. He. 3d geographic network displays. *ACM Sigmod Record*, 25(4):50–54, 1996.

[10] A. Debiasi, B. Simões, and R. De Amicis. Supervised force directed algorithm for the generation of flow maps. In *Proceedings of the WSCG 2014 - 22nd International Conference on Computer Graphics*, 2014.

[11] A. Debiasi, B. Simões, and R. De Amicis. 3darclens: Interactive network analysis on geographic surfaces. In *Proceedings of the IVAPP - 6th International Conference on Information Visualization Theory and Applications*, 2015.

[12] J. Diepstraten, D. Weiskopf, and T. Ertl. Interactive cutaway illustrations. In *Computer Graphics Forum*, volume 22, pages 523–532. Wiley Online Library, 2003.

[13] J. F. Donges, Y. Zou, N. Marwan, and J. Kurths. The backbone of the climate network. *EPL (Europhysics Letters)*, 87(4):48007, 2009.

[14] N. Elmqvist, Y. Riche, N. Henry-Riche, and J.-D. Fekete. Mélange: Space folding for visual exploration. *Visualization and Computer Graphics, IEEE Transactions on*, 16(3):468–483, 2010.

[15] N. Elmqvist and P. Tsigas. A taxonomy of 3d occlusion management for visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 14(5):1095–1109, 2008.

[16] G. W. Furnas. *Generalized fisheye views*, volume 17. ACM, 1986.

[17] G. W. Furnas. A fisheye follow-up: further reflections on focus+ context. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 999–1008. ACM, 2006.

[18] S. Gustafson, P. Baudisch, C. Gutwin, and P. Irani.

Wedge: clutter-free visualization of off-screen locations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 787–796. ACM, 2008.

[19] P. Irani, C. Gutwin, and X. D. Yang. Improving selection of off-screen targets with hopping. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 299–308. ACM, 2006.

[20] B. Jenny. Adaptive composite map projections. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2575–2582, 2012.

[21] H. Jenny, B. Jenny, and L. Hurni. Interactive design of 3d maps with progressive projection. *The Cartographic Journal*, 47(3):211–221, 2010.

[22] A. Lambert, R. Bourqui, and D. Auber. 3d edge bundling for geographical data visualization. In *Information Visualisation (IV), 2010 14th International Conference*, pages 329–335. IEEE, 2010.

[23] I. Legrand, H. Newman, R. Voicu, C. Cirstoiu, C. Grigoras, C. Dobre, A. Muraru, A. Costan, M. Dediu, and C. Stratan. Monalisa: An agent based, dynamic service system to monitor, control and optimize distributed systems. *Computer Physics Communications*, 180(12):2472–2498, 2009.

[24] H. Lorenz, M. Trapp, J. Döllner, and M. Jobst. Interactive multi-perspective views of virtual 3d landscape and city models. In *The European Information Society*, pages 301–321. Springer, 2008.

[25] M. J. McGuffin, L. Tancau, and R. Balakrishnan. Using deformations for browsing volumetric data. In *Visualization, 2003. VIS 2003. IEEE*, pages 401–408. IEEE, 2003.

[26] S. Möser, P. Degener, R. Wahl, and R. Klein. Context aware terrain visualization for wayfinding and navigation. In *Computer Graphics Forum*, volume 27, pages 1853–1860. Wiley Online Library, 2008.

[27] T. Munzner, E. Hoffman, K. Claffy, and B. Fenner. Visualizing the global topology of the mbone. In *Information Visualization'96, Proceedings IEEE Symposium on*, pages 85–92. IEEE, 1996.

[28] C. Niederauer, M. Houston, M. Agrawala, and G. Humphreys. Non-invasive interactive visualization of dynamic architectural environments. In *Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 55–58. ACM, 2003.

[29] T. Nocke. Basics and visual analytics of climate networks. In *Proceedings of the 2011 workshop on Climate knowledge discovery*, pages 2–2. ACM, 2011.

[30] B. Pflesser, U. Tiede, and K. H. Höhne. Towards realistic visualization for surgery rehearsal. In *Computer Vision, Virtual Reality and Robotics in Medicine*, pages 487–491. Springer, 1995.

[31] P. Rodgers. Graph drawing techniques for geographic visualization. *Exploring geovisualization*, pages 143–158, 2005.

[32] T. Ropinski and K. Hinrichs. Real-time rendering of 3d magic lenses having arbitrary convex shapes. 2004.

[33] W. R. Tobler. Experiments in migration mapping by computer. *The American Cartographer*, 14(2):155–163, 1987.

[34] C. Tominski, J. Abello, F. Van Ham, and H. Schumann. Fisheye tree views and lenses for graph visualization. In *Information Visualization, 2006. IV 2006. Tenth International Conference on*, pages 17–24. IEEE, 2006.

[35] C. Tominski, J. F. Donges, and T. Nocke. Information visualization in climate research. In *Information Visualisation (IV), 2011 15th International Conference on*, pages 298–305. IEEE, 2011.

[36] M. Tory and C. Swindells. Comparing exovis, orientation icon, and in-place 3d visualization techniques. In *Graphics Interface*, volume 3, pages 57–64. Citeseer, 2003.

[37] J. W. Tukey. Exploratory data analysis. 1977.

[38] J. J. Van Wijk. Unfolding the earth: Myriahedral projections. *The Cartographic Journal*, 45(1):32–42, 2008.

[39] J. Viega, M. J. Conway, G. Williams, and R. Pausch. 3d magic lenses. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 51–58. ACM, 1996.

[40] A. Wolff. Graph drawing and cartography. In R. Tamassia, editor, *Handbook of Graph Drawing and Visualization*, chapter 23, pages 697–736. CRC Press, Boca Raton, FL, 2013.

View-dependent Simplification using Parallel Half Edge Collapses

Thomas Odaker
Ludwig Maximilians Universitaet
München, Germany
odaker@a1.net

Dieter Kranzlmueller
Ludwig Maximilians Universitaet
München, Germany
kranzlmueller@ifi.lmu.de

Jens Volkert
Johannes Kepler University
Linz, Austria
jv@gup.jku.at

ABSTRACT

Highly detailed models are a requirement for many applications in computer graphics. The necessary level of detail, however, may vary depending on the application. To provide a tradeoff, mesh simplification is used to generate approximations of a model which can be used to reduce processing time.
We present a parallel approach to triangle mesh simplification that is designed to allow fast, view-dependent simplification of manifold triangle meshes. Our approach performs a vertex analysis on every vertex of a given triangle mesh and selects a set of vertices for removal. Vertex removal is executed using the parallel half edge collapse. Based on the half edge collapse that replaces an edge with one of its endpoints, we have devised a set of boundaries that enable parallel application of half edge collapses even on neighbouring vertices.
Since the mesh topology may not allow removal of all vertices marked for removal in one step, we apply multiple iterations of the parallel half edge collapse, reevaluating remaining vertices marked for removal for further improvement of results.

Keywords

mesh simplification, level of detail, half edge collapse, computer graphics, view dependent simplification, real-time rendering

1 INTRODUCTION

Simplification of triangle meshes has been a well researched area for several decades [Cla76a]. Mesh simplification is the process of applying a simplification operator to a given triangle mesh with the effect of reducing the number of vertices or triangles that make up the mesh [Lue02a].
Mesh simplification is widely used as an approach to reduce the resources needed for processing a mesh by reducing the stored geometry data.
The disadvantage of simplification is usually a loss of detail. The triangles of a mesh are used to represent a surface shape. Removing triangles or vertices from a triangle mesh therefore means the result cannot represent the object as accurately as the original. Simplification algorithms can be aimed either to remove a large amount of vertices or triangles, create a simplification that represents the original object well or to provide fast processing times.

Over the years a wide range of algorithms for triangle mesh simplification has been developed with varying results in terms of performance and quality of the simplified mesh. The ones most important to this paper can roughly be classified in three categories:

Vertex clustering This algorithm presented by Rossignac and Borrell in [Ros92a] is designed to work on arbitrary polygonal models. First the bounding box of an object is determined. This bounding box is then divided into a number of cells. All vertices within a cell are clustered into a single vertex and the faces of the model are updated accordingly. While this approach can be very fast, it can also cause drastic alterations to the topology of a model and create low quality simplifications. Attempting to improve these shortcomings, several variants to cell generation have been devised [Sch03a], [Low97a].

Vertex removal Presented in Schroeder et. al. [Sch92a], the vertex removal approach removes a

 ISBN 978-80-86943-65-7

vertex and all its adjacent edges and triangles from a mesh and then retriangulates the resulting hole.

Edge contraction Edge contraction provides mesh simplification by removing edges from a mesh and replacing it with a vertex [Hop93a]. This approach is usually performed iteratively, selecting one edge at a time and contracting it.

While many algorithms are designed to generate a generic simplification of a mesh, some have been devised for generating a view-dependent simplification. The latter take a current camera position into account and aim to minimize the visible differences between the simplification and the original.
One example can be found in Hu et al. [Hu09a] with the presentation of this approach being extended in [Hu10a]. It describes a bottom up approach where a simplified mesh is stored. During runtime the desired level of detail is restored by applying precalculated operations stored in a data structure.
Papageorgiou and Platis [Pap15a] present a GPU-accelerated approach that also relies on edge collapses with the goal to execute a number of edge collapses in parallel and therefore speed up the simplification. Their algorithm selects a number of independent areas. Each area can safely be simplified by an edge collapse without affecting other areas. Selection of independent areas and execution of edge collapses is repeated, until the desired simplification target is reached.
Another example is presented by DeCoro and Tatarchuk [DeC07a]. It describes a GPU accelerated implementation of vertex clustering to provide a fast, view-dependent simplification of an arbitrary mesh.
In comparison to these approaches, the parallel half edge collapse was designed not to rely on any precomputed simplification operations and provide a topology preserving, parallel approach to simplification that selects the simplification operations at runtime.

The parallel half edge collapse was designed to enable view-dependent real time simplification of manifold triangle meshes. The simplification operator of choice is the half edge collapse.
The edge collapse operator (Figure 1) is applied to a pair of vertices (V_1, V_2) connected by an edge e. It collapses V_1 and V_2 into a single vertex V', removing e and all triangles that contain it from the mesh [Hop96a]. The position of V' is chosen freely. This allows to optimize the replacement position for e and achieve a better simplification as well as improve the representation of the original mesh.
A more restrictive version of the edge collapse is the half edge collapse (Figure 2). The half edge collapse uses V_1 or V_2 as replacement for e. It therefore replaces an edge with one of its endpoints and does not change any vertex data like the stored normal.

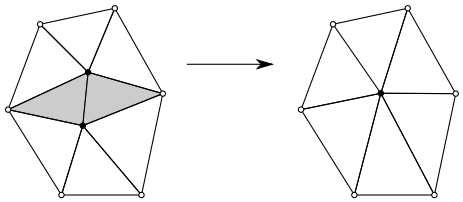


Figure 1: Edge collapse

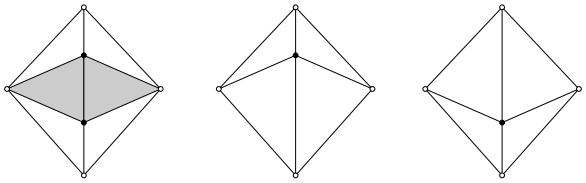


Figure 2: Half edge collapse

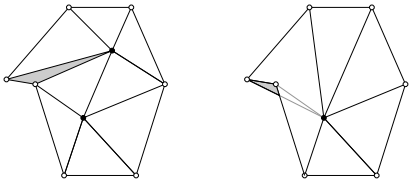


Figure 3: Edge collapse causing a mesh foldover

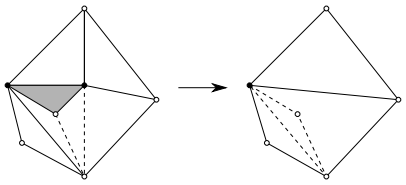


Figure 4: Edge collapse causing a topological inconsistency

Although the edge collapse and the half edge collapse are easy to implement, they can have a negative effect on the simplified mesh by causing a foldover (see Figure 3) or a topological inconsistency (Figure 4) which has to be avoided [Lue02a].

2 ALGORITHM OVERVIEW

Initially a vertex analysis is performed on all vertices of a given manifold mesh that selects a number of vertices for removal.
Given a vertex V that has been selected for removal, the algorithm determines all vertices $N = \{N_1, N_2, ..., N_q\}$ that share an edge with V. Each of these edges is considered a possible half edge collapse. V is removed by selecting a vertex pair V, N_i and performing a half edge collapse on it. The replacement position for the edge is N_i, since V is supposed to be removed from the mesh.
To speed up the simplification a parallel approach for the vertex removal is introduced. The algorithm proposed in this paper analyses all vertices $V = \{V_1, V_2, ..., V_n\}$ of a mesh with the intention of selecting a set of vertices $R = \{VR_1, VR_2, ..., VR_o\}$ that should be removed ($S = \{VS_1, VS_2, ..., VS_p\}, S = V \setminus R$,

step 1 in Figure 5). Since every vertex in R is to be removed from the mesh, not all the neighbours propose a valid half edge collapse. Only edges connecting VR_i to a vertex $N_j \in S$ are valid choices for a half edge collapse.
For each vertex $VR_i \in R$ the algorithm tries to:

- Find all neighbours $N \in S$ of VR_i that share an edge.
- Select one edge e_k connecting the pair N_j, VR_i.
- Perform the half edge collapse on e_k using N_j as the replacement position and removing VR_i from the mesh.

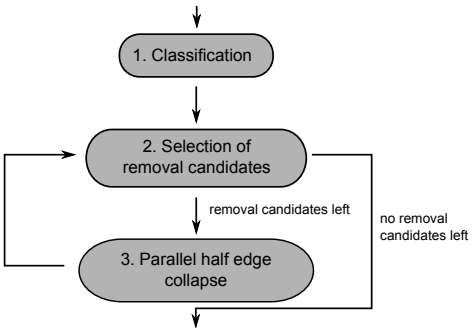


Figure 5: Algorithm overview

The algorithm does not apply a series of iterative edge collapses. It rather processes all vertices in R in parallel. Although this approach has great potential to speed up the simplification process, it causes several difficulties:

- All neighbouring vertices of a vertex VR_i being included in R can prevent VR_i from being removed instantly.
- Parallel execution of half edge collapses on neighbouring vertices may cause hard to detect mesh foldovers and topological inconsistencies.

Not all vertices $VR \in R$ may have a neighbour in S. Since the algorithm only considers an edge connecting a vertex VR_i to a neighbour $N_j \in S$ a valid half edge collapse, it may be impossible for the vertex to be removed.
The removal of the vertices in R has to be divided into several iterative steps. In each step a list of removal candidates $C = \{C_1, C_2, ..., C_n\}$ is composed, containing all $VR \in R$ that have at least one neighbour $N_j \in S$ (step 2 in Figure 5). The algorithm tries to remove all vertices in C in parallel (step 3 in Figure 5). The steps of determining the candidate list and removing those vertices is repeated iteratively, until all $VR \in R$ have been processed and no vertices remain in R.

Another problem arising due to the parallelism of the algorithm is the selection of the individual replacement positions for vertices in R. As described by Xia et al. [Xia97a] a single edge or half edge collapse performed on a mesh may cause a mesh foldover or a topological inconsistency which has to be avoided. While this is relatively easy to detect by checking for rotations of normal vectors of an affected triangle before and after a collapse, the parallel approach chosen here renders this test invalid. It is possible that a single half edge collapse is valid - it does not have a negative effect on the mesh - the parallel execution of two valid half edge collapses however, can cause a foldover or topological inconsistency.
To make sure that a possible replacement position for a vertex does not cause the aforementioned problems, a set of boundaries is defined. For each vertex $VR_i \in R$ an individual set of boundaries $B(VR_i)$ is calculated. It can be used to determine, which neighbouring vertices $N \in S$ can be safely used as a vertex pair for a half edge collapse. Any neighbour $N_j \in S$ that lies within the boundaries $B(VR_i)$ can be used as a vertex pair VR_i, N_j for a half edge collapse.

Given two vertices $VR_i, VR_j \in R$ are connected by an edge and have neighbours in S. VR_i and VR_j are to be processed in parallel and have one or more possible half edge collapses. In order to avoid foldovers and topological inconsistencies, the boundaries $B(VR_i)$ have to take all possible replacement positions for VR_j into account. The same condition is in effect for $B(VR_j)$ and the replacement positions for VR_i. The parallel half edge collapse was designed to avoid communication between neighbouring vertex removals to allow for a fast implementation. The boundaries are created to block all half edge collapses that may cause issues whenever a neighbouring vertex is removed in parallel. This may result in combinations of half edge collapses for neighbouring vertices in R that would not cause foldovers or topological inconsistencies being considered invalid.
The algorithm has to take into account that for some vertices in C no replacement position, that would not cause a foldover or a topological inconsistency, can be found. A distinction has to be made whether this is caused by the topology or by the restrictive boundary due to the execution of parallel half edge collapses and an isolated vertex removal could be safely executed.
Furthermore these restrictive boundaries may cause an issue where neighbouring vertices in C may mutually block a removal, effectively causing a deadlock and preventing a region of the mesh from being simplified. This situation has to be identified and resolved to avoid vertices remaining in R indefinitely and keep the

simplification from being completed.

3 VERTEX ANALYSIS

Given a manifold triangle mesh every vertex V_i is analysed and categorized either for being removed from the mesh or remaining. Two sets of vertices are created. $R = \{VR_1, VR_2, ..., VR_n\}$ contains a list of vertices that are to be removed while $S = \{VS_1, VS_2, ..., VS_m\}$ contains the remaining ones.
This initial classification of vertices does however not determine a final list of vertices that are removed from the mesh. Since several iterations of parallel half edge collapses may be performed on the mesh and the results of each iteration cannot be predicted, an additional step is introduced. After a half edge collapse has been executed on a vertex pair VR_i, VS_j, all vertices VR_k that were neighbours of VR_i are analysed again and may be removed from R and added to S. This step allows for an adaptation to the chosen half edge collapses and may improve the overall result of the simplification.

For classification a vertex analysis is performed. This step assigns an error value to each vertex of the mesh. Since this vertex error has to be updated during the reclassification step after each half edge collapse, the error metric was chosen with data dependency and computing time in mind.

The initial analysis is done on a per-vertex basis and only takes the position of neighbouring vertices into account. The error metric chosen here relies on the distance of the neighbouring vertices from the tangent plane of a vertex that is defined by the vertex normal stored for the vertex. Although this only takes the local effect of a vertex removal into account and ignores a possible removal of neighbouring vertices, it was chosen for two reasons. Firstly this error metric requires little computing time and since it only relies on neighbouring vertices, it does not require data collection before a calculation, which allows for fast updates after half edge collapses.
Secondly during reclassification after a half edge collapse some neighbours of vertices in C may be members of R and therefore marked for removal. These do not provide relevant data since they may be removed in subsequent iterations. The metric has to be easily adaptable to take into account data from neighbours only, that are to remain in the mesh.

Given a vertex V the tangential plane is constructed as $ax+by+cz+d=0$ with $t=[a,b,c,d]$. For each neighbouring vertex N_i with the position $n=[n_x,n_y,n_z,1]$ the quadratic distance from the tangential plane $d(V,N_i)=(t \bullet n)^2$ is calculated (Figure 6).

The final error value for a vertex $e(V)$ is defined as

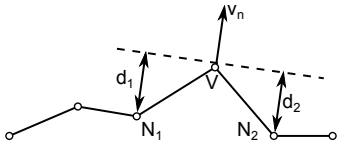


Figure 6: Vertex error calculation

the average distance between the tangent plane and the neighbouring vertices:

$$e(V) = \frac{\sum_{i=1}^{m}(d(V,N_i))}{m} \quad (1)$$

For a view-dependent simplification $e(V)$ is scaled using the angle between the view vector and the vertex normal as well as the distance between vertex and camera. This step has to be performed at runtime and computes the per-vertex classification. While the error $e(V)$ is computed from the static mesh data, the influence of the camera data allows for a view-dependent simplification as the camera data is updated before the simplification is computed.

Vertex classification into R and S is performed by comparing the calculated vertex error $e(V)$ with a user defined error threshold u. Any vertex with an error value smaller than or equal to u is marked for removal and stored in R. The remaining ones form S.
The problem that arises when using this approach is a potential selection of a majority of vertices (if not all) of a mesh for removal. Since the algorithm performs a half edge collapse on an edge that connects a vertex in R with one in S, the number of parallel operations could be limited severely. In case of all vertices being marked for removal, the execution of the parallel half edge collapse would be impossible.
To guarantee the algorithm is always functional and improve parallelism, an artificially modified vertex error is introduced for some vertices.
This approach selects a number of vertices from the mesh and assigns them a very high vertex error e_m. The artificially assigned error is defined as $e_m > u$. This guarantees these chosen vertices to always remain in the mesh and the algorithm remains functional.
For further refinement of this approach several "layers" of error manipulation are introduced. The first layer L_0 contains the aforementioned vertices that are assigned $e_m(V) > u$. Each additionally created layer L_i (i > 0) selects additional vertices and assigns them a predefined error value $e_m(L_i)$. This value is user defined. Modifying the user threshold u can hence be used not only to control the level of detail of the simplified mesh, but also to select a set of vertices that is to remain in the mesh with the purpose of accelerating the execution of the simplification.
For this approach a number of vertices has to be selected for each layer:

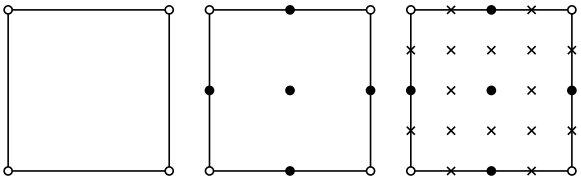


Figure 7: Example point generation

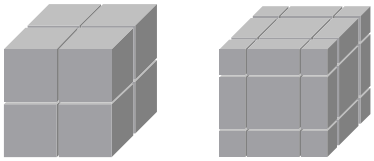


Figure 8: Example volume generation

- The bounding box of the mesh is determined and a 3-dimensional grid of points $G_0 = \{P_1^0, P_2^0, ..., P_n^0\}$ for layer L_0 is created within and with an equal axial distance from each other.

- Addition of layers L_i with sets of points $G_i = \{P_1^i, P_2^i, ..., P_o^i\}$. The new points are generated at halfway points between points in G_{i-1}. The distance between two points in G_i is $d(G_i)$. This distance is separate for each axis (2-dimensional example in Figure 7).

- For each point: generation of a volume $B(P_j^i)$ centered around P_j^i with a side length of $d(G_i)$ (trimmed to the bounding box). Figure 8 shows the volumes for the first two examples in Figure 7.

- For each volume: determination of all vertices $V(P_j^i)$ within the volume

- For each volume: selection of one vertex from $V(P_j^i)$ and manipulation of the vertex error

This approach creates a number of points and the corresponding volumes. Then a number of vertices has to be selected. For each point $P_i \in G_j$ the algorithm selects a vertex that lies within the assigned volume - if any can be found. The magnitude of the error manipulation depends on the layer L_j the point was created for and is selected by the user for each L_j.
The vertex selection takes the distance between a point P_i^j and the vertex as well as the vertex error into account. This is done to favour vertices close to the generated points and simultaneously consider the vertex error to choose vertices that are more likely to remain in the mesh.
Given a point P_i^j in set G_j all vertices V_k within the volume around P_i^j are determined. Then the distance between V_k and P_i^j is calculated $d(V_k, P_i^j) = |p_i^j - v_k|$.
The value $m(V_k, P_i^j)$ is calculated using the maximum side length l of the volume around P_i^j and the error $e(V_k)$.

$$m(V_k, P_i^j) = (l - d(V_k, P_i^j))^2 * e(V_k) \qquad (2)$$

The result of this calculation is a weighted vertex error. For a point P_i^j the vertex with the largest $m(V_K, P_i^j)$ within the corresponding volume is selected.
For each layer a minimum error is defined by the user. If a vertex is selected by a point P_i^j and the stored error $e(V)$ is smaller than the user selected value $e(L_j)$, the error is replaced by the user value.
Since the vertices are classified by comparison of the vertex error with a threshold, the user can select which set of vertices determined by this approach is to be used for the simplification.

4 PARALLEL HALF EDGE COLLAPSE

The result of the vertex classification is all the mesh's vertices being divided into two sets: R containing all vertices to be removed from the mesh and S with all vertices to remain.
The algorithm only considers half edge collapses on edges connecting a vertex in R to one in S. So the first step of the actual simplification is to determine a list of vertices C that contains all vertices $C_i \in R$ that have at least one neighbour $N \in S$.
For each C_i all possible half edge collapses are determined and one of them has to be selected and executed. The problem with half edge collapses lies in topological inconsistencies and mesh foldovers that may occur.
In order to allow for a fast implementation, the execution of half edge collapses on neighbouring vertices in parallel has to be carried out without any exchange of data. A simple approach to prevent negative effects on the mesh would be to compare triangle normals before and after the execution of the half edge collapse. A maximum angle between the triangle normal of a triangle before and after the collapse can be defined. If the angle is greater than a defined threshold, the collapse is considered invalid.
While this difference in angles works for isolated (half) edge collapses, it cannot be applied to the parallel half edge collapse, since two valid half edge collapses executed on neighbouring vertices may cause foldovers or topological inconsistencies. Figure 9 shows an example for this effect. Executing just the half edge collapse V_4, V_2 or V_3, V_1 creates a valid mesh. When both are applied in parallel however, the result is a folded triangle that has to be avoided. Here the original triangle V_3, V_4, V_5 and the modified triangle V_1, V_2, V_5 are overlapping.

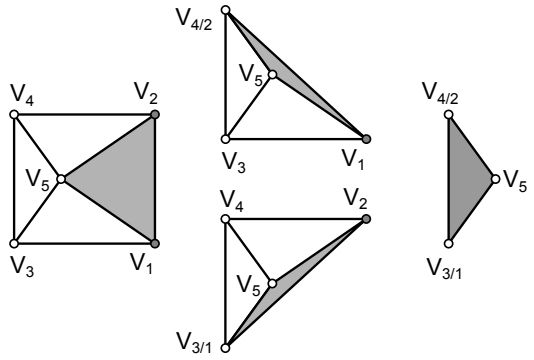


Figure 9: Two valid half edge collapses cause an inconsistency

The approach used in the algorithm presented here defines a set of boundaries for each vertex in C. This takes any neighbouring vertices to be removed in parallel into account and prevents foldovers and topological inconsistencies. It is also designed for view-dependent refinement and defines the boundaries using the view vector of the camera.

Given a vertex VR that is a candidate for removal, all triangles that contain VR have to be determined. For each triangle a number of planes is calculated that form the per-triangle boundaries and that are added to the set of boundaries $B(VR)$. Any potential half edge collapse has to be checked against the entire boundary $B(VR)$ to make sure that none of the triangles containing VR are affected by undesired effects. Since the result of any half edge collapse executed in parallel on neighbouring vertices is not known, the boundaries are defined in a way that prevents the aforementioned effects, no matter which half edge collapse a neighbouring vertex chooses (if any at all). This has the negative side effect of restricting the boundaries and possibly blocking combinations of half edge collapses that would not have negative effects on the mesh. In order to allow a higher degree of freedom when choosing collapses, the boundaries are created with regard to how many vertices of a triangle are to be removed in parallel. For a triangle, one of three sets of boundaries can be created, dependent if one, two or all three vertices of the triangle are current candidates for removal.

4.1 Test 1

The first case covers a triangle with only one vertex subjected to a half edge collapse. One edge of the triangle remains unchanged. While the aforementioned simple test that checks for large variations in triangle normals would suffice for this situation, the boundaries are created here as well to provide a uniform test and allow for a fast implementation.

Given the view-dependent approach of this algorithm a foldover or a topological inconsistency is created when the normal of a triangle before and after the manipulation changes orientation in relation to the view vector (the dot product between the view vector and the triangle normal changes sign). Therefore all boundaries are created taking the camera position E into account.

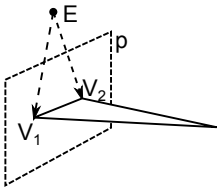


Figure 10: Boundary 1

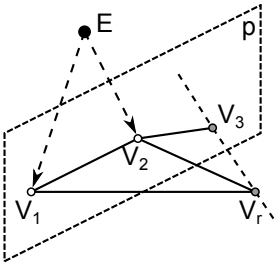


Figure 11: Boundary test 1

For a single vertex V_r to be removed in a triangle composed of V_r, V_1, V_2, a plane p is defined. Since V_1 and V_2 as well as the edge between them remain unchanged, removing V_r rotates the triangle normal around this edge. The plane p is defined using both V_1 and V_2, as well as the vectors $\overrightarrow{V_1 - E}$ and $\overrightarrow{V_2 - E}$.

Figure 10 shows an example for such a boundary plane with the camera looking down at the triangle from above. In order to test whether a half edge collapse is valid, the edge that is to be removed is intersected with the plane. If an intersection can be found, the endpoints of the edge lie on opposite sides of the plane and the half edge collapse is considered invalid.

Figure 11 shows an example for such a test. The edge V_r, V_3 is to be collapsed into V_3. While the vector $\overrightarrow{V_3 - V_r}$ has an intersection with the plane, the edge V_r, V_3 does not and the half edge collapse is considered valid.

This first boundary for an isolated half edge collapse always computes the correct result. As mentioned earlier, some boundaries can block valid half edge collapses in order to allow for parallel execution. While this is true for the following boundaries for two or three candidates in one triangle, it does not apply here.

4.2 Test 2

Given a triangle where two vertices are candidates for removal, the boundaries above are not valid anymore since they do not take the half edge collapse executed on neighbouring vertices into account. They would only prevent collapses that individually cause foldovers

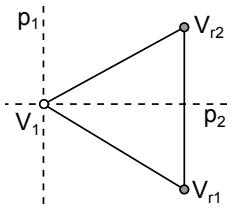


Figure 12: Boundary 2

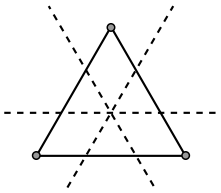


Figure 13: Boundary 3

or inconsistencies. See Figure 9 for an example.
Given a triangle with the vertices V_{r1}, V_{r2}, V_1 with V_{r1}, V_{r2} being candidates for removal, there is no edge that would remain unchanged and therefore no predictable rotational axis for the triangle normal. This results in a boundary consisting of two planes (Figure 12 shows an example from the viewpoint of the camera). They are designed to assign each removal candidate an individual area where a replacement position can be found. Any overlap between these two areas would enable a change of orientation between the triangle normal and the view vector.
The first plane p_1 is defined using the vectors $\overrightarrow{V_{r1} - V_{r2}}$ and $\overrightarrow{E - V_1}$ as well as the point V_1. It is therefore parallel to the edge between the two removal candidates and lies through the remaining vertex of the triangle.
The second plane p_2 is defined to intersect the edge between V_{r1} and V_{r2}. For this purpose a median was chosen. Plane p_2 uses the vectors $\overrightarrow{E - V_1}$ and $\overrightarrow{\frac{V_{r1} + V_{r2}}{2} - V_1}$ as well as the point V_1.
The test for a valid half edge collapse works similarly to the previous one. Given an edge that has to be tested for possible foldovers, it is intersected with both p_1 and p_2 to check if the collapse would cause in issue with this triangle. If an intersection between the edge and either of the planes can be found, it is considered invalid and will not be executed.
These boundaries can block valid half edge collapses and even combinations of valid half edge collapses for V_{r1} and V_{r2}. This is however accepted in order to allow for a parallel execution.

4.3 Test 3

The last case handles triangles where all 3 vertices V_{r1}, V_{r2}, V_{r3} are to be removed. Test 2 places both planes p_1 and p_2 to contain the remaining vertex V_1 of the triangle. Since all three vertices are a candidate for removal here as well, these boundaries are no longer valid.

Given a triangle V_{r1}, V_{r2}, V_{r3} test 3 creates two planes per removal candidate as well. All planes contain the centroid S of the triangle. Plane p_1 for vertex V_{r1} uses the vectors $\overrightarrow{V_{r2} - V_{r1}}$ and $\overrightarrow{E - S}$. Plane p_2 uses $\overrightarrow{V_{r3} - V_{r1}}$ and $\overrightarrow{E - S}$.
Both these planes are parallel to the edges that contain the removal candidate and are again defined using the view vector of the camera. While for test 2 the two planes were identical for V_{r1} and V_{r2}, here two removal candidates only share a single plane. As a result two planes have to be constructed for each removal candidate, while three are necessary for the entire triangle (Figure 13 shows an example from the viewpoint of the triangle).
Testing of a half edge collapse is done as described in test 2. Given a removal candidate V_{r1} both necessary planes are constructed and an edge e is intersected with both p_1 and p_2. If either p_1 or p_2 intersect e, the edge is not valid for a half edge collapse as it would pose a risk of causing a mesh foldover or a topological inconsistency.

In order to find all valid half edge collapses for a vertex VR, first all triangles have to be found. Then for each triangle the number of removal candidates is determined and the corresponding set of boundaries is constructed. At this point, a set of planes P is on hand. Each edge VR, VS_j is then intersected with all $p_k \in P$. If an edge intersects any plane in P, it is considered invalid.

4.4 Half edge collapse selection

After a list of possible half edge collapses has been compiled, one of them has to be chosen and executed. We devised a slightly modified version of the quadric error metric presented by Garland and Heckbert[Gar98a] for this selection.
[Gar98a] presents an approach that uses pair collapse to replace a vertex pair V_1, V_2 with a single vertex V'. For this purpose it defines a vertex error $\triangle(V')$ in relation to V_1 and V_2. The algorithm then tries to find a position for V' that minimizes $\triangle(V')$. The authors however also suggest that a simple solution would be to use either V_1, V_2 or $\frac{V_1+V_2}{2}$ and choose the position with the lowest value of $\triangle(V')$.
The parallel half edge collapse makes use of this error metric. As suggested by the authors of [Gar98a], we use the replacement position with the lowest vertex error according to the quadric error metric. Given a removal candidate VR and a set of neighbours N where all $N_i \in S$ with edges VR, N_i form a valid half edge collapse, the error value according to the quadric error metric is calculated. Then the N_i with the lowest

error value $\triangle(N_i)$ is chosen and the half edge collapse performed.
Since half edge collapses are not executed isolatedly when using the parallel half edge collapse and several edges may be collapsed into a single vertex in parallel, the quadric error metric is not updated with each collapse as suggested by [Gar98a]. Instead the QEM is recalculated for each removal candidate during boundary computation, using the current intermediate result of the mesh.

5 DEADLOCK PREVENTION

While the boundaries described in the previous section allow for parallel execution of half edge collapses, they create the disadvantage of possibly blocking combinations of parallel half edge collapses that would not cause a foldover or a topological inconsistency. This problem cannot only occur for two neighbouring vertices. Several vertices could enter a state where they mutually block half edge collapses, effectively causing an area of the mesh that cannot be simplified even though for each vertex one or more half edge collapses could be performed individually.
If a vertex is to be removed, but all possible half edge collapses are being blocked by the boundaries, an additional computation has to be performed. For each edge e_i that was tested against the boundaries, the blocking planes have to be determined.
If all planes that blocked the half edge collapses are constructed by test 1, the current topology of the mesh around the removal candidate VR does not allow any half edge collapse without causing either a foldover or a topological inconsistency. In this first case the algorithm aborts the search for a valid half edge collapse for VR. The vertex remains in the mesh and is part of the simplified result.
The second case needs a more complex handling. Here one or more planes blocking a collapse are constructed from a triangle with two or three vertices to be removed. As described above, these boundaries could prevent valid half edge collapses from being executed. To avoid that, the algorithm always computes a separate set of planes for VR, using only test 1 for all triangles, effectively ignoring possible parallel half edge collapses. This leads to two results for each half edge collapse. The first one taking parallel half edge collapses into account and potentially blocking valid collapses, the second one to check if the topology allows for a removal of VR at all. If the second result fails as well as the first one, no half edge collapse can be safely performed for VR and the vertex remains in the mesh and no further attempts of removal will be taken.
If at least one half edge collapse is allowed by the second result, the vertex remains a candidate for removal for the next iteration i_{n+1}.
During iteration i_{n+1} the status of all neighbouring vertices of VR is compared to that of the previous iteration i_n. If none of the removal candidates have been subjected to a half edge collapse, a deadlock is assumed. To resolve this the stored vertex error $e(VR)$ is compared to the one of the neighbouring removal candidates. Unless $e(VR)$ is greater than the vertex error of all neighbouring removal candidates, VR is marked as "to be ignored". Ignored vertices are skipped during the selection of removal candidates and although neighbouring vertices do not consider them valid targets for a possible half edge collapse, they are not considered as "to be removed" and therefore do not cause the application of test 2 or 3. This approach allows neighbouring vertices to be subjected to a half edge collapse and so a possible deadlock can be resolved.
Once a neighbouring vertex is reclassified or removed by a half edge collapse, the ignore flag is deleted and VR can be selected as a candidate for removal again.

6 RECLASSIFICATION

As described in section 3, the initial classification performs calculations based on neighbouring vertices but does not consider the possible removal of those. For this reason an additional step is performed in between iterations.
After a vertex VR_i is removed by a half edge collapse, the vertex error of all neighbours $N_j \in R$ of VR_i is invalidated due to the changes to the mesh. A new vertex error that is used to validate the classification is then calculated for all N_j.
After an iteration has been completed, the algorithm determines a list $C = \{C_0, C_1, ..., C_n\}$ containing all vertices $VR \in R$ that have at least one neighbour in S. Since half edge collapses change the edges, this list needs to be regenerated after each iteration. Then the vertex error $e(C_i)$ is updated taking these changed edges into account. If the new vertex error $e(C_i)$ is greater than a certain threshold, C_i is reclassified, removed from R and C and added to S. It is therefore no longer to be removed from the mesh.
This operation may be executed very often during runtime. In addition this error metric for reclassification should not deviate from the metric used in the initial vertex analysis to prevent a potential reclassification of a majority of vertices in R. For those reasons a variation of the metric presented in section 3 is applied here.

The initial classification uses the average of distances between the tangent plane of a vertex V that is determined by the stored vertex normal and not changed during the simplification and its neighbours N. At this point it has to be taken into account that one or more neighbouring vertices may be marked for removal and therefore do not provide useful data, since

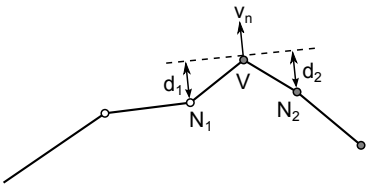


Figure 14: Reclassifying vertices

they may not be part of the final, simplified mesh.
The metric used here still relies on the distance between the tangent plane of V and N_i, it does not rely on the average though and calculates the maximum value instead.

Figure 14 shows an example for a possible reclassification. Given that all vertices to the left of V are elements of S, while the ones to the right are contained in R, only the neighbour N_1 is included in the error metric for the recalculation. d_2 calculated using $N_2 \in R$ is omitted.
Since this newly calculated error value differs from the initial classification, an adapted threshold has to be defined for the reclassification. If the newly calculated error value is greater than the threshold, the vertex is reclassified as described above, otherwise it remains in R and the next iteration tries to find a possible half edge collapse to remove it from the mesh.

7 RESULTS

For early testing a GPU accelerated implementation of the parallel half edge collapse using CUDA on a Nvidia Geforce GTX 670 GPU was used.
For testing purposes, the model of the Stanford Bunny was simplified. The original version of this mesh is made up of 35 947 vertices forming 69 451 triangles.
Three different simplifications of the original mesh were calculated by adapting the threshold for the initial classification. For these early results the main focus was the overall processing time for the simplification, the resulting triangle count of the simplified mesh and the number of iterations necessary until all vertices marked for removal had been processed.
Figure 15 shows the comparison between the wireframe of the three simplified models derived from the original.
The three cases resulted in models made from 48 831, 29 014 and 17 565 triangles respectively. The first case was completed in 1.9 ms, while cases 2 and 3 required 3.3 and 4.4 ms.
As to be expected, the necessary number of iterations increases as more vertices are marked for removal and hence removed from the mesh. This can be explained by the restriction that only edges between vertices in R and S are considered possible half edge collapses. Marking more vertices for removal, therefore reducing the amount of vertices in S, reduces the number of possible half edge collapses and results in additional

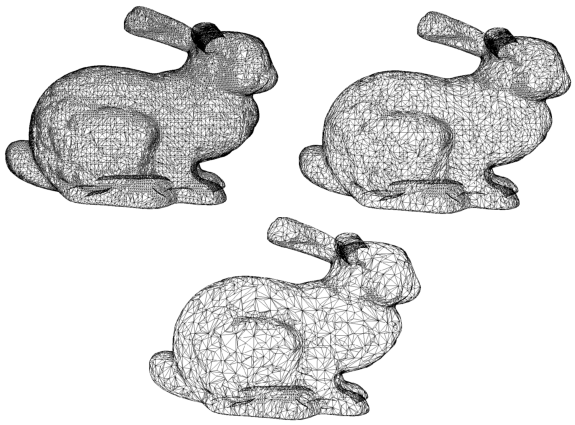


Figure 15: Simplification results

iterations.
While the parallel half edge collapses managed to remove all marked vertices in 2 iterations for case 1, 5 and 8 are respectively needed in the other two cases, causing the increase in processing time.
Another potentially limiting factor that could cause additional iterations are deadlocks. Since these are not resolved until the following iteration, a small number of mutually locking vertices can have a large impact on the runtime. For runtime critical applications of the parallel half edge collapse a threshold for a minimum amount of vertices in R should be considered. Since the half edge collapse and therefore the parallel half edge collapse preserves manifold connectivity at every step, the simplification can be safely aborted after each iteration.

These early tests have also shown that the use of the artificially introduced vertex error can have a great impact on runtime. While there was no difference in runtime measurements for the first test case, there is a clearly visible difference for test cases with a higher number of necessary iterations. Executing test case 3 without the modified vertex error increased the number of iterations by a factor of 4, simultaneously causing a great increase in overall runtime for the simplification as well.

7.1 Future work

The parallel half edge collapse can provide fast simplification with good results. A main bottleneck of the approach can be posed by the restriction, that only edges between vertices in R and S are considered a possible half edge collapse. A majority of the vertices of a mesh being marked for removal or a disadvantageous topology can cause a reduction in parallel half edge collapses and force the execution of additional iterations. Future work can focus on several approaches to resolve these issues.

An improved vertex analysis/classification could provide a way to improve parallelism and reduce processing time. A more precise initial classification could provide additional removal candidates in each iteration, therefore reducing the number of necessary iterations and improving performance.
The difficulty caused by this approach lies within the prediction of chosen half edge collapses. Preselecting a precise set R without adapting to the outcome of simplification operations has the potential to result in lower quality simplifications than adapting after each iteration.
Another significant improvement could be made by allowing vertices in R that do not have a neighbour in S to be collapsed rather than having to wait for one or more iterations until a neighbouring vertex $N \in S$ is provided. This would also increase the freedom of the simplification, possibly offering a greater number of half edge collapses per vertex to choose from and improve the overall result of the simplification.

8 CONCLUSION

The parallel half edge collapse is designed to provide fast simplification with good results. The emphasis on lack of communication between half edge collapses as well as the focus on fast error metrics potentially allows for short processing times, enabling the parallel half edge collapse for real time applications using view-dependent simplification and thus further improve the result of the simplification.
The parallel design enables an implementation on modern GPUs, further reducing the time needed to calculate the simplified meshes.

On the other hand however the usage of the half edge collapse instead of the more generic edge collapse can prove to be a limiting factor. The possibly limited number of selectable half edge collapses and the focus on error metrics designed for fast computation and updating can impact the result. This can have a negative influence on the quality of the overall result of the simplification, which is a trade-off that has to be accepted for real time use.

9 REFERENCES

[Cla76a] Clark, J. H. Hierarchical geometric models for visible surface algorithms, Com. of ACM 19, No. 10, pp.547-554, 1976

[Ros92a] Rossignac, J., and Borrell, P. Multi-resolution 3D Approximations for Rendering Complex Scenes, Modeling of Computer Graphics: Methods and Applications, pp.455-465, 1992

[Sch03a] Schaefer, S., and Warren., J. Adaptive vertex clustering using octrees, Proceedings of SIAM Geometric Design and Computing 2003, Vol. 2, pp.491-500, 2003

[Low97a] Low, K.-L., and Tan, T., S., Model simplification using vertex-clustering, SI3D Proceedings 1997, pp.75-ff., 1997

[Sch92a] Schroeder, W., J., Zarge, J., A., and Lorensen, W., E. Decimation of triangle meshes, ACM SIGGRAPH Computer Graphics Vol. 26, No. 2, pp.65-70, 1992

[Hop93a] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., A., and Stuetzle, W. Mesh optimization, ACM SIGGRAPH Proceedings 1993, pp.19-26, 1993

[Hu09a] Hu, L., Sander, P., V., and Hoppe, H. Parallel view-dependent refinement of progressive meshes, I3D 2009 Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games, pp.169-176, 2009

[DeC07a] DeCoro, C., and Tatarchuk, N. Real-time mesh simplification using the GPU, I3D 2007 Proceedings of the 2007 Symposium on Interactive 3D Graphics Vol. 2007, pp.161-166, 2007

[Hop96a] Hoppe, H. Progressive meshes, ACM SIGGRAPH 1996 Proceedings, pp.99-108, 1996

[Lue02a] Luebke, D., Watson, B., Cohen, J., D., Reddy, M., and Varshney, A. Level of Detail for 3D Graphics, 2002

[Xia97a] Xia, J., C., El-Sana, J., and Varshney, A. Adaptive real-time level-of-detail-based rendering for polygonal models, IEEE Transactions on Visualization and Computer Graphics Vol. 3, No. 2, pp.171-187, 1997

[Gar98a] Garland, M., and Heckbert, P., S. Surface simplification using quadric error metrics, SIGGRAPH Proceedings 1997, pp.209-216, 1997

[Hu10a] Hu, L., Sander, P., and Hoppe, H. Parallel view-dependent level of detail control, IEEE Transactions on Visualization and Computer Graphics Vol. 16, No. 5, pp.718-728, 2010

[Pap15a] Papageorgiou, A., and Platis, N. Triangular mesh simplification on the GPU, The Visual Computer: International Journal of Computer Graphics Vol. 31, Issue 2, pp.235-244, 2015

IGFTT: towards an efficient alternative to SIFT and SURF

Ícaro Oliveira de Oliveira
Federal Technological University of Paraná
Brazil
icaroo@utfpr.edu.br

Keiko Veronica Ono Fonseca
Federal Technological University of Paraná
Brazil
keiko@utfpr.edu.br

Eduardo Todt
VRI Research Group, Federal University of Paraná
Brazil
todt@inf.ufpr.br

ABSTRACT

The invariant feature detectors are essential components in many computer vision applications, such as tracking, simultaneous localization and mapping (SLAM), image search, machine vision, object recognition, 3D reconstruction from multiple images, augmented reality, stereo vision, and others. However, it is very challenging to detect high quality features while maintaining a low computational cost. Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF) algorithms exhibit great performance under a variety of image transformations, however these methods rely on costly keypoint's detection. Recently, fast and efficient variants such as Binary Robust Invariant Scalable Keypoints (BRISK) and Oriented Fast and Rotated BRIEF (ORB) were developed to offset the computational burden of these traditional detectors.

In this paper, we propose to improve the Good Features to Track (GFTT) detector, coined IGFTT. It approximates or even outperforms the state-of-art detectors with respect to repeatability, distinctiveness, and robustness, yet can be computed much faster than Maximally Stable Extremal Regions (MSER), SIFT, BRISK, KAZE, Accelerated KAZE (AKAZE) and SURF. This is achieved by using the search of maximal-minimum eigenvalue in the image on scale-space and a new orientation extraction method based on eigenvectors.

A comprehensive evaluation on standard datasets shows that IGFTT achieves quite a high performance with a computation time comparable to state-of-the-art real-time features. The proposed method shows exceptionally good performance compared to SURF, ORB, GFTT, MSER, Star, SIFT, KAZE, AKAZE and BRISK.

Keywords

IGFTT, feature detectors, keypoint, computer vision, repeatability

1 INTRODUCTION

Feature detectors have become an essential component in contemporary computer vision research. The main goal is to find salient image keypoints that can be repeatedly detected under various image transformations and then construct distinctive and robust representations for them.

This paper aims to tackle this problem by developing an improvement on the well-known GFTT keypoint detector [21]. This technique is attractive because of its good performance. For feature detection, scale invariant and stable keypoints are selected in the scale space according to maximal-minimum eigenvalues responses. Furthermore, it was built a fast accurate orientation estimation by the eigenvector's orientation.

We have validated the IGFTT detector whose precision, recall and the execution time figures. The experiments show that IGFTT achieves quite a high performance with a computation time comparable to state of the art, e.g. ORB and it is more efficient in terms of repeatability than KAZE, BRISK, SURF and SIFT. Also, the orientation extraction method developed in this paper presented less errors than other methods. The rest of this paper is organized as follows: First, an overview of related work is given in Section 2. Section 3 describes a review about GFTT detector. Section 4 describes the improvement to GFTT developed here. The experimental evaluation is carried out in Section 5, and finally Section 6 presents the discussion and conclusions.

2 BACKGROUND

2.1 Feature Detection

Corners are points in an image where two lines meet perpendicularly [10]. They can be any points between two lines with different directions or between two points with strong image gradients. Corners are particularly important since they can be used to locate

 ISBN 978-80-86943-65-7

and make the registration of objects and to provide measures of their dimensions, for example, knowledge about orientation will be vital for a robot finds the best way of picking up an object, inspection and other applications [8].

Thereby, corners are interesting for identifying parts of an image, efficient to match image to image and have great relevance to the accuracy and efficiency of machine vision. The corners with local maximum or minimum intensity are called interest points or keypoints.

SIFT [12] is a pioneering method which produces high quality features based in gradients and requires high computational effort. SURF [5] detects keypoints faster than the original SIFT, without loss in performance and, recently, other methods were proposed improving either processing time or repeatability, e.g. ORB [20], KAZE [4] and BRISK [23].

SIFT [12], SURF [5], BRISK [23] and ORB [20] have the same structure: they detect best keypoints in a scale pyramid type like shown in Figure 1 and extract orientation using the directed gradients or moments.

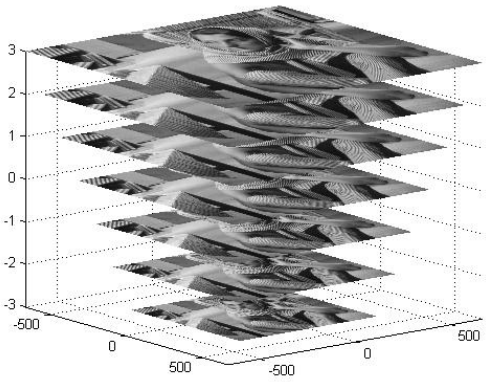


Figure 1: Image Pyramid [9]

Each detector applies an algorithm to detect stable keypoints in each scale level. For example, SIFT [12] detects in a Difference of Gaussians applied in each level, SURF [5] detects in wavelet and integral images, BRISK [23] detects in a scale space with interpolation and ORB [20] detects in a simple pyramid scales. Other detectors like AGAST [13], FAST [18] and SUSAN [22] detect stable keypoints in an unique image and don't use scale pyramids or extract orientation.

SIFT [12] extracts scale-invariant features through detecting local extremas of the Difference-of-Gaussian (DoG) over scale space applied in the image. The DoG is a faster approximation of Laplacian of Gaussian (LoG). SURF [5] proposes the Fast-Hessian detector. This method approximates the second order Gaussian derivatives of Hessian matrix from rectangle filters. Futhermore, the Fast-Hessian detector works with integral images to compute 3 times faster than DoG. Recently, Features from Accelerated Segment Test (FAST) [18] detector is based on the SUSAN corner detector [22]. The circular area center is used to determine brighter and darker neighboring pixels on the circle describing the segment. FAST uses a Bresenham's circle [18] of diameter 3.4 pixels as test mask. Thus, for a full accelerated segment test, 16 pixels have to be compared to obtain the value of the nucleus. In the sequence, FAST selects the 9 top segments (FAST-9) that get high repeatability between 16 pixels. For each keypoint, the FAST calculates a sum of the differences between each 9 circular pattern's pixels and the nucleus' pixel and selects the keypoint with a sum lower than a threshold. This parameter controls the sensitivity of the corner response. A large t-value results in few but therefore only strong corners, while a small t-value yields also corners with smoother gradients. The alternative FAST-ER [19] generalizes FAST by allowing it to learn the 9 top segments with the best repeatability and small loss of efficiency for any scene.

AGAST [13] improves FAST performance by combining specialized decision trees. Since these FAST-based detectors do not deal with scale change, BRISK [23] takes AGAST to detect feature candidates and searches for the maximum FAST score over scale space to achieve scale invariance. KAZE [4] proposes an automatic feature detection in nonlinear scale spaces using efficient AOS techniques and variable conductance diffusion. Next, AKAZE [17] proposes to use recent numerical schemes called Fast Explicit Diffusion (FED) embedded in a pyramidal framework to dramatically speed-up feature detection in nonlinear scale spaces. The STAR [2] keypoint detector uses an approximation that allows to preserve rotational invariance applied in scale space formed by a bi-level approximation of the Laplacian of Gaussians (LoG) filter. ORB [20] detects FAST features filtered by Harris at each level in the scale pyramid of the image and calculates a new fast and accurate orientation component to FAST. A Maximally Stable Extremal Region (MSER) [14] is a connected component of an appropriately thresholded image. The word 'extremal' refers to the property that all pixels inside the MSER have either higher (bright extremal regions) or lower (dark extremal regions) intensity than all the pixels on its outer boundary. The 'maximally stable' in MSER describes the property optimized in the threshold selection process.

2.2 Feature Description

SIFT [12] creates a histogram of local gradient orientations and locations, where the gradient orientations are quantized into 8 orientation bins and the space locations are quantized into a 4x4 grid.

SURF [5] works with the Haar wavelet responses as the local features. Since the above gradient-based descriptors can only deal with linear illumination changes, some other methods have been proposed to tackle more general illumination changes by using relative intensity orders of pixels rather than the original intensities. BRIEF [7] uses a relatively small number of binary tests between pixels to represent the local patch as a binary string. ORB [20] develops a rotation invariant by the intensity centroid that extends the descriptor BRIEF, and it is more discriminative by learning a good subset of binary tests than BRIEF original. BRISK [23] introduces a Gaussian weighted pattern for sampling the neighborhood of keypoints. The long-distance pairs are used to estimate the local dominant orientation and the short-distance pairs are used to build a binary descriptor. FREAK [1] proposes a retinal sampling pattern based on the human visual system, and computes the binary descriptor by comparing image intensities over the retinal patterns. KAZE [4] uses the M-SURF descriptor adapted to a nonlinear scale space framework. AKAZE [17] introduces a Modified-Local Difference Binary (M-LDB) descriptor that is highly efficient, exploits gradient information from the nonlinear scale space, is scale and rotation invariant and has low storage requirements.

In practice, it is very challenging to obtain a high quality feature whilst maintaining a low computational cost on several transformations. Therefore, this work aims to improve the GFTT detector developing a novel orientation estimation and creating by simple scale pyramid to GFTT detector.

3 GOOD FEATURES TO TRACK: REVIEW

In this section the Good Features to Track [21] will be briefly described. Consider an image sequence $I(x,t)$, with $x = [u,v]^T$ where u and v are the coordinates of an image point. If a point of time sampling t is substantially high, then the points of the image I are displaced, however their intensities remain unchanged:

$$I(x,t) = I(\delta(x), t+\tau) \tag{1}$$

where $\delta(\cdot)$ is the motion field that specifies the transformation applied to image points. The authors approximate the transformation to a translation through the fast-sampling hypothesis, in other words, $\delta(x) = x + d$, where d is a displacement vector. Variable d is used to search keypoints in the frame's sequences. The image motion model can keep some noises becoming not perfect, so the problem is to find d which minimizes the Sum of Squared Differences (SSD) to find the displacement d residuals, from this the equation below is computed:

$$min_d(\sum_W [I(x+d,t+\tau) - I(x,t)]^2) \tag{2}$$

where W is an image window around the keypoint and t is the frame in t time. If we apply first-order Taylor expansion of $I(x+d,t+\tau)$ into (2) we can obtain a simple linear system formed by

$$Gd = e \tag{3}$$

where

$$G = \sum_w \begin{bmatrix} I_u^2 & I_u I_v \\ I_u I_v & I_v^2 \end{bmatrix}, \; e = -\tau \sum_w I_t \, [I_u \; I_v]^T$$

with $[I_u \; I_v] = \nabla I = [\frac{\partial I}{\partial u} \; \frac{\partial I}{\partial v}]$ and $I_t = \frac{\partial I}{\partial u}$. From Eq. (3): d is the solution of (3), that is, $d = G^{-1}e$, and is used to predict a new (registered) frame. This method is iterated through of the Newton-Raphson scheme to converge the displacement d. So, assuming that λ_1 and λ_2 are the eigenvalues of G, the feature is detected if $\min(\lambda_1,\lambda_2) > \lambda$; where λ is an user-defined threshold.

Thereafter, it is assuming the image I and next image J. If I and J are dissimilar images then the feature is dropped. The dissimilarity images I and J is measured by the equation

$$sum(I - J) > threshold \tag{4}$$

Between consecutive frames a translation model is sufficient for tracking, however an affine model is necessary when frames are far.

It is more expensive to calculate an affine model for each frame than use a simple model, for example, a scale model. In this case, the IGFTT used a simple scale model and obtains better results related to precision than SIFT, SURF, AKAZE, KAZE, FAST, ORB, MSER, STAR and GFTT.

4 IMPROVE GOOD FEATURES TO TRACK (IGFTT)

Focusing on computation efficiency, our detection methodology is inspired by the work of Shi Tomasi et al. [21] for detecting keypoints in the image. Aiming the achievement of invariance to scale which is crucial for high-quality keypoints, we go a step further by searching for keypoint not only in the image plane, but also in scale-space.

The IGFTT has the same GFTT's parameters like *minDistance*, *qualityLevel*, *blockSize* and *N*. *minDistance* filters the keypoints with maxima-minimal eigenvalues lower than the *minDistance*.

Variable *qualityLevel* is used to threshold the minimum eigenvalues. In other words, the maximum global eigenvalue is multiplied by the variable *qualityLevel*. We select the eigenvalues smaller than *qualityLevel*. Variable *blockSize* is the block's length used to calculate the minimum eigenvalues in the image. Variable N is maxima amount of detected keypoints. Further, we add the variables *scaleLevels* and *scaleFactor* to generate the scale space. Variable *scaleLevels* is the amount of levels in the scale space and variable *scaleFactor* is the applied factor in the image at each level of the scale space.

In the literature, there are three scale-space types: wavelet, gaussian pyramid and simple pyramid. The wavelet is linear scale space representation that analyzes the signal from scale and resolution. A Gaussian pyramid is a scale-space that creates a set of images scaled down from a image. Each image is also weighted by a Gaussian blur. This pyramid is used to create a scale space with blur invariant, however the eigenvalues already are blur invariants. The simple pyramid is a scale-space that creates a set of scaled down images from an original image. In this case, we used the simple pyramid to make easier implementation.

```
Data: the pyramid scale space from images with
      various scales
Result: KeyPoints
initialization;
foreach image in pyramid do
    calculate gradient matrix;
    calculate the covariance in the gradient matrix;
    extract eigenvectors and eigenvalues in the
    directions x and y;
    select the minimal eigenvalues between directions
    x and y and the eigenvector for each pixel;
    calculate the maximum eigenvalue local;
    select the keypoints with eigenvalues bigger than
    maximum value local * qualityLevel;
    sort the selected keypoints by the eigenvalue;
    foreach point in keypoints do
        if total_KeyPoints == N then
            break;
        end
        if distance between point and all keypoints <
        minDistance then
            KeyPoints.add(point);
            total_KeyPoints = total_KeyPoints + 1;
        end
    end
end
```

Algorithm 1: IGFTT

4.1 Fast Orientation by Eigenvector

Our approach uses a simple but effective measure of corner orientation, based on the eigenvector, because the eigenvector represents the direction preserved by a linear transformation applied in the gradient matrix. From $min(\lambda_1, \lambda_2)$ we selected the eigenvector of G with the minimal eigenvalue. The coordinates (x, y) of the eigenvector are used to extract the interest point orientation.

$$angle = atan2(y, x); \tag{5}$$

where atan2 is the quadrant-aware version of arctan.

4.2 Descriptor

We use the FREAK descriptor applied in each scale space. For a detected keypoint k at scale σ_i, the FREAK descriptor extracts the pattern formed by Difference-of-Gaussians inspired in the retinal pattern of the eye. Furthermore, FREAK extracts this pattern in various scales and orientations. the FREAK has integral images to accelerate the description process. Finally, the descriptor is interpolated and normalized.

5 EXPERIMENTS

We have evaluated our method on the standard Oxford dataset [15] and on two new datasets [24, 11] with geometric and photometric transformations like rotation, scale, viewpoint, image blur, JPEG compression and illumination. The implementations for all detectors came from OpenCV 3.0 beta [6].

5.1 Detector evaluation

We compare IGFTT detector with SURF, ORB, GFTT, MSER, STAR, SIFT, KAZE, AKAZE and BRISK detectors. We test the precision and recall scores of different detectors for gradually increasing transformation. In these experiments we set $minDistance = 1$. Through repeatability criterion introduced in [16], we analyze the overlap error of two correspond regions in various transformations. This error is defined as

$$1 - \frac{R_{\mu_a} \cap R_{H^T \mu_b H}}{R_{\mu_a} \cup R_{H^T \mu_b H}} \tag{6}$$

where R_μ represents the elliptic region defined by $x^T \mu x = 1$. H is the homography between the two images. The intersection of the regions is $R_{\mu_a} \cap R_{H^T \mu_b H}$ and $R_{\mu_a} \cup R_{H^T \mu_b H}$ is their union. The intersection of the regions must be greater than 0. The area of these regions are computed numerically. The repeatability score for a given pair of images is defined as

$$\frac{true\ keypoints}{detected\ keypoints} \quad (7)$$

where the amount of keypoints with $overlap\ error < \varepsilon_t$ is represented by $true\ keypoints$. $Detected\ keypoints$ is the amount of keypoints detected and the overlap error threshold represented by ε_t is 40%. We take into account only the regions located in scene's parts present in both images. In this case, the repeatability represents the recall score. The precision score is

$$\frac{true\ keypoints}{matched\ keypoints} \quad (8)$$

where matched keypoints are the amount of the keypoints with intersection of the regions greater than 0. The true keypoints are based in putative matches, in other words, a putative match is formed by a single pairing of keypoints, where a keypoint cannot be matched to more than one other. IGFTT detector consistently outperforms SURF, ORB, GFTT, MSER, STAR, SIFT, KAZE, and BRISK in most cases, which can be attributed for the minimal eigenvalues features used by the detector like shown in Figure 2 and Figure 3.

In this section, we compare the computation times of IGFTT to SURF, ORB, GFTT, HARRIS, MSER, Star, SIFT, KAZE, and BRISK. The experiments are carried out on desktop Intel Ivy Bridge Core i5-3450 3.10GHz CPU and 16GB DRAM, using the first image of the each dataset's sequence compared with other images. The results are averaged on 1000 experiments runs.

In general presented the IGFTT better averages than other detectors using minimal eigenvalues to select keypoints in each level of the scale-space. Furthermore, this feature, applied in each scale space level, is faster than the corresponding algorithms applied by the SURF, MSER, SIFT, AKAZE or KAZE. We used the average of the precision, recall and execution time compared to those obtained from the reference detectors [17, 4, 12, 5, 14, 3, 20, 23, 21].

The SIFT exhibits lower averages on precision (67.69% vs 93.21%) and recall (55.00% vs 81.78%), but more execution time (209.794ms vs 57.52ms) than IGFTT. The IGFTT eigenvalues and keypoints selection methods are faster and more repeatedly than Difference-of-Gaussian used by SIFT. The Difference-of-Gaussian used by SIFT did not detect the same keypoints of the first image in the other images from the various transformations in all datasets.

The SURF exhibits lower average on precision (85.13% vs 93.21%) and on recall (71.74% vs 81.78%) than IGFTT. The Hessian Matrix used by SURF, it did not detect the same keypoints of the first image in the other images from various transformations. In some cases, the SURF shows better invariance than IGFTT in the datasets trees (blur) and venice (zoom), because of the Frobenius score and the box filter used by SURF. The Hessian Matrix applied in the integral images and Wavelet show more time values (83.61ms vs 57.52ms) than IGFTT, due to the complexity of both algorithms.

The KAZE shows lower average on precision (78.34% vs 93.21%) and on recall (69.57% vs 81.78%) than IGFTT. The nonlinear diffusion filtering did not detect the same keypoints of the first image in the other images with various transformations. Furthermore, KAZE is more complex than IGFTT, the KAZE showing a more average time (409.64ms vs 57.52ms) than IGFTT.

The AKAZE shows lower average on precision lower (78.52% vs 93.21%) and on recall (65.16% vs 81.78%) than IGFTT. The Fast Explicit Diffusion filtering did not detect the same keypoints of the first image in the other images from various transformations. Furthermore, the AKAZE is more complex than IGFTT, showing more average time (119.08ms vs 57.52ms) than IGFTT.

The BRISK shows lower average on precision (78.95% vs 93.21%) and on recall (58.21% vs 81.78%) than IGFTT. The pyramid with octave levels and octave in-between levels did not detect the same keypoints of the first image in the other images from various transformations. But, BRISK is faster (23.15ms vs 57.52ms) than IGFTT in average of the execution time, because it has AGAST detector that is more efficient than GFTT.

The ORB presents lower average on precision (88.30% vs 93.21%) and on recall (78.46% vs 81.78%) than IGFTT. The Harris score used by ORB did not detect the same keypoints of the first image in the other images from various transformations. In some cases ORB shows better invariance than IGFTT in the Reichstag (various transformations) and ubc (jpeg compression) datasets, due to the Harris score. But, the ORB is faster (16.58ms vs 57.52ms) than IGFTT in average of the time, because the FAST detector is more efficient than GFTT detector.

The GFTT shows lower average on precision (74.71% vs 93.21%) and on recall (55.81% vs 81.78%) than IGFTT. The GFTT detector is not invariant scale, viewpoint, rotation and zoom. In other words, the GFTT detector did not detect the same keypoints of the first image in the other images with various transformations. But, the GFTT is faster (19.38ms vs 57.52ms) than IGFTT in average time.

The STAR shows lower average on precision (71.90% vs 93.21%) and on recall (53.80% vs 81.78%) than IGFTT. The Star detector is not fully invariant to scale, viewpoint, rotation and zoom. In other words, the STAR detector did not detect the same keypoints of the first image in the other images with various

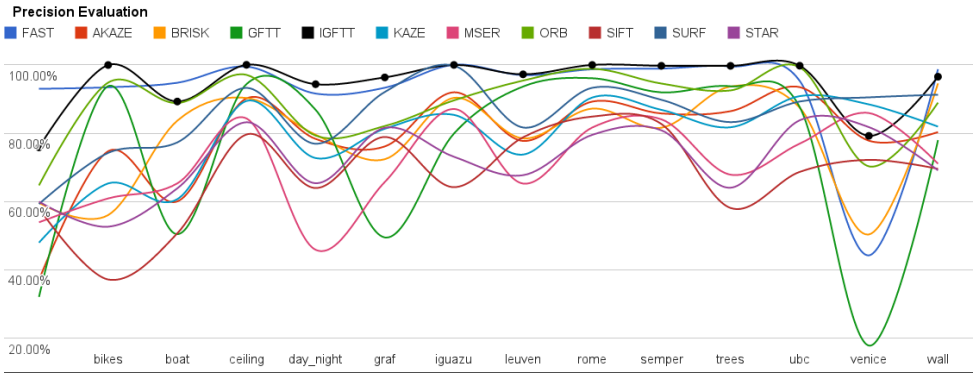


Figure 2: The precision of the detectors on various datasets with affine transformations.

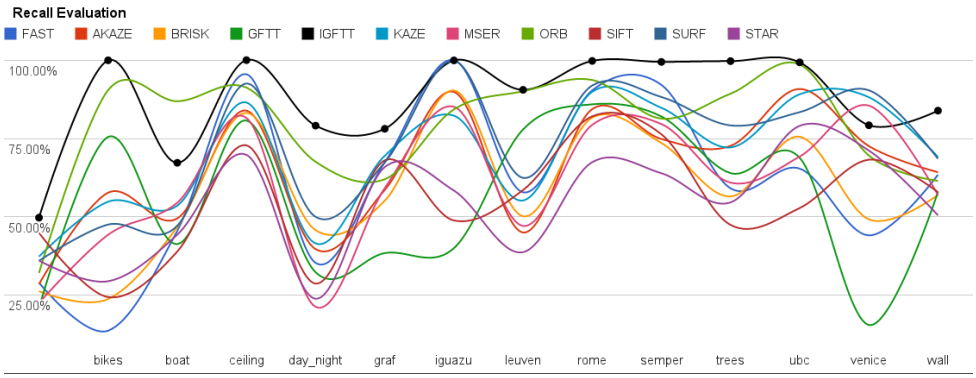


Figure 3: The recall of the detectors on various datasets with affine transformations.

transformations. But, the STAR is faster (18.52ms vs 57.52ms) than IGFTT in average time.

The MSER shows lower average on precision (71.14% vs 93.21%) and on recall (60.51% vs 81.78%) than IGFTT. The MSER methods to detect regions are based on extremal regions did not detect the same keypoints of the first image in the other images with various transformations, although the detector been fully invariant to these transformations. Furthermore, the MSER was more complex than IGFTT, the MSER was more average time (89.88ms vs 57.52ms) than IGFTT.

The FAST shows lower average on precision (92.71% vs 93.21%) and on recall (61.34% vs 81.78%) than IGFTT. The FAST detector is not invariant to scale, viewpoint, rotation and zoom. In other words, the FAST detector did not detect the same keypoints of the first image in the other images with various transformations. But, the FAST is faster (4.73ms vs 57.52ms) than IGFTT in average time.

5.2 Descriptor evaluation

We compare IGFTT detector combined with SURF, ORB, SIFT, FREAK, BRIEF and BRISK descriptors with SURF, ORB, SIFT, KAZE and BRISK. Furthermore, we used the brute-force matching developed in OPENCV 3.0 to make the matching between the images. We used the precision and recall scores. The overlap error threshold represented by ε_t is 50%. We take into account only the regions located in the part of the scene present in both images. This way, the recall score for a given pair of images is defined as

$$\frac{true\ keypoints}{visible\ keypoints} \tag{9}$$

where the amount of keypoints with $overlap\ error < \varepsilon_t$ is represented by $true\ keypoints$, $visible\ keypoints$ is the amount of keypoints in the part of the scene present in both images and the overlap error threshold represented by ε_t is 50%. In this case, the repeatability represents the recall score. The precision score is 8 where matched keypoints are the amount of the keypoints matched between the images pair. The true keypoints are based in putative matches, in other words, a putative match is formed by a single pairing of keypoints, where a keypoint cannot be matched to more than one other.

In this section, we compare the computation times between IGFTT combined with various descriptors SURF, ORB, SIFT, KAZE, and BRISK. The experiments are performed on desktop Intel Ivy Bridge Core i5-3450 3.10GHz CPU and 16GB DRAM, using the first image of the each dataset's sequence compared with other images. The results are averaged on 1000 experiments runs. We used the average of the precision, recall and execution time to compare the all detectors and descriptors used here.

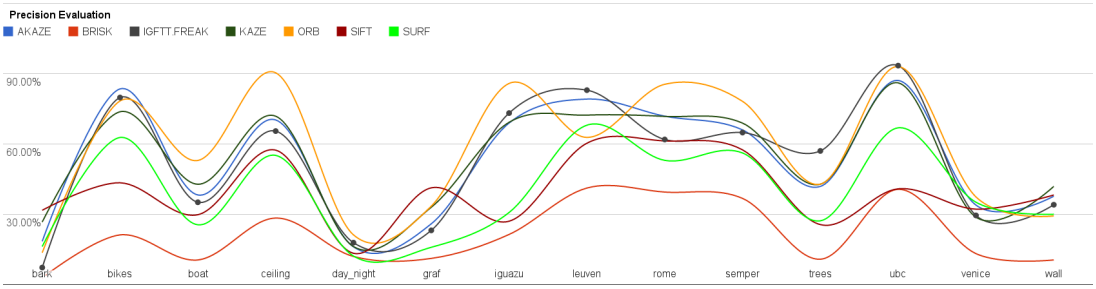


Figure 4: The precision of the detectors combined with descriptors on various datasets with affine transformations.

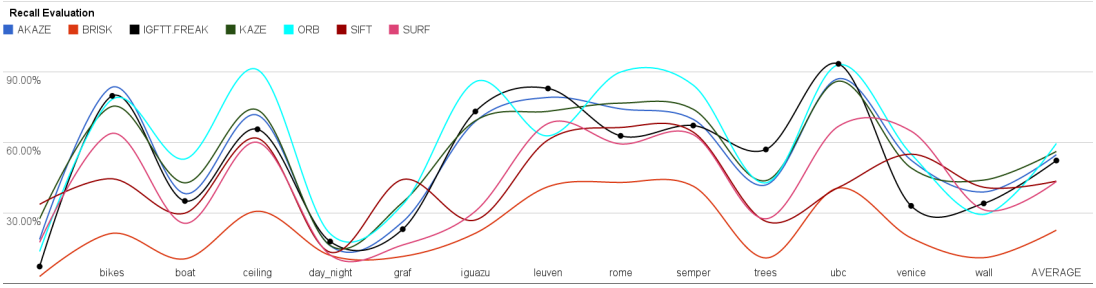


Figure 5: The recall of the detectors combined with descriptors on various datasets with affine transformations.

In some cases the IGFTT with FREAK descriptor (called IGFTT.FREAK) has better averages in precision and recall than SIFT, SURF and BRISK. Furthermore, IGFTT.FREAK is faster than SIFT, SURF, KAZE, AKAZE and BRISK.

The SIFT exhibits lower average on precision (40.02% vs 51.85%) and on recall (43.57% vs 52.34%), furthermore it is greater time average (957.85ms vs 64.77ms) than IGFTT.FREAK. The FREAK descriptor using retinal binary pattern describes very well the regions around the keypoints detected by the IGFTT and this combination show better results than the Difference-of-Gaussian and the descriptor based in gradients' orientations. Furthermore, in [1] the descriptor FREAK generate a binary string formed by a sequence of one-bit Difference-of-Gaussians. In other words, this descriptor is more discriminant than SIFT descriptor.

The SURF exhibits lower average on precision (39.66% vs 51.85%) and on recall (43.44% vs 52.34%) than IGFTT. The FREAK descriptor is faster and robust than SURF descriptor through cascade of binary strings. The Hessian Matrix applied in the integral images, Wavelet and shows more time average (657.60ms vs 64.77ms) than IGFTT.

The KAZE shows greater average on precision (53.31% vs 51.85%) and on recall (56.21% vs 52.34%) than IGFTT. The nonlinear diffusion filtering extracts some features more discriminant than the FREAK descriptor like shown in bark, boat, ceiling, graf, rome, semper and wall datasets. However, the KAZE is more complex than IGFTT, the KAZE shows more time average (629.55ms vs 64.77ms) than IGFTT.

The AKAZE shows lower average on precision (52.81% vs 51.85%) and on recall (54.82% vs 52.34%) than IGFTT. The Fast Explicit Diffusion filtering extracts some features more discriminant than the FREAK descriptor like shown in bark, bikes, boat, ceiling, graf, rome, semper, venice and wall datasets. However, the AKAZE is more complex than IGFTT, the AKAZE shows more time average (331.06ms vs 64.77ms) than IGFTT.

The BRISK shows lower average on precision (21.47% vs 51.85%) and on recall (22.75% vs 52.34%) than IGFTT. The descriptor BRISK based in a bit-string descriptor from intensity comparisons is lower discriminant than the retinal binary pattern used in the FREAK descriptor. Furthermore, BRISK was more complex than IGFTT, the BRISK shows more time average (74.33ms vs 64.77ms) than IGFTT.

The ORB results in greater average on precision (57.51% vs 51.85%) and on recall (59.59% vs 52.34%) than IGFTT. The Oriented BRIEF descriptor and the ORB detector are more discriminant in some cases than IGFTT with FREAK descriptor. The ORB shows a better invariance than IGFTT in the bark, boat, ceiling, day_night, graf, iguazu, rome, semper and venice datasets. Furthermore, the ORB is faster (24.05ms vs 64.77ms) than IGFTT in time average, the FAST detector used in the ORB is faster than IGFTT detector.

6 CONCLUSIONS

We have presented a novel method named IGFTT, which tackles the classic Computer Vision problem of detecting image keypoints for cases without sufficient a priori knowledge on the scene, camera poses

and transformation. In contrast to well-established algorithms with proven high performance, such as SIFT and SURF, the method at hand offers a dramatically faster alternative at comparable precision performance - a statement which we base on an extensive evaluation using an established framework.

IGFTT relies on an easily configurable, the unique properties of IGFTT can be useful for a wide spectrum of applications, in particular for tasks with hard real-time constraints or limited computation power: IGFTT finally offers the quality of high-end keypoints in such time-demanding applications.

Amongst avenues for further research into IGFTT, we aim to explore alternatives to the GFTT search to yield higher repeatability whilst maintaining or improve the speed. Furthermore, we aim at analyzing both theoretically and experimentally other efficient detectors applied in the scale space or other filters like ORB applied.

7 REFERENCES

[1] R. O. A. Alahi and P. Vandergheynst. Freak: Fast retina keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2012.

[2] M. Agrawal, K. Konolige, and M. R. Blas. Censure: Center surround extremas for realtime feature detection and matching. *European Conference on Computer Vision 2008*, 2008.

[3] M. Agrawal, K. Konolige, and M. R. Blas. Censure: Center surround extremas for realtime feature detection and matching. In D. A. Forsyth, P. H. S. Torr, and A. Zisserman, editors, *ECCV (4)*, volume 5305 of *Lecture Notes in Computer Science*, pages 102–115. Springer, 2008.

[4] P. F. Alcantarilla, A. Bartoli, and A. J. Davison. Kaze features. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ECCV'12, pages 214–227, 2012.

[5] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, pages 404–417. Springer, 2006.

[6] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[7] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. BRIEF: Computing a Local Binary Descriptor Very Fast. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1281–1298, 2012.

[8] E. R. Davies. *Computer and machine vision theory, algorithms, practicalities*. Elsevier, Waltham, Mass., 2012.

[9] C. R. del Blanco Adán. Computer vision, 2014. [Online; acessado em 01-April-2014].

[10] K. Demaagd, A. Oliver, N. Oostendorp, and K. Scott. *Practical Computer Vision with SimpleCV - Making Computers See in Python.* O'Reilly Media, 2012.

[11] E. D. J. Heinly and J. M. Frahm. Comparative evaluation of binary features. *European Conference on Computer Vision*, 2012.

[12] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, pages 91–110, 2004.

[13] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Proceedings of the European Conference on Computer Vision (ECCV'10)*, September 2010.

[14] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. *Proc. of British Machine Vision Conference*, pages 384–396, 2002.

[15] K. Mikolajczyk. Affine covariant features, 2014. [Online; acessado em 01-Abril-2014].

[16] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1):43–72, Nov. 2005.

[17] J. N. Pablo F. Alcantarilla and A. Bartoli. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *In British Machine Vision Conference (BMVC)*, 2013.

[18] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006.

[19] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *Pattern Analysis and Machine Intelligence*, pages 105–119, 2010.

[20] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. *Proceedings of the 2011 International Conference on Computer Vision*, 2011.

[21] J. Shi and C. Tomasi. Good features to track. *Computer Vision and Pattern Recognition*, 1994.

[22] S. M. Smith and J. Brady. Susan-a new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, 1997.

[23] M. C. Stefan Leutenegger and R. Siegwart. Brisk: Binary robust invariant scalable keypoints. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.

[24] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. *Computer Vision and Pattern Recognition*, 2008.

A Machine Learning Approach to Automate Facial Expressions from Physical Activity

Tarik Boukhalfi
École de technologie supérieure
Montreal, Canada
tarik.boukhalfi.1@ens.etsmtl.ca

Christian Desrosiers
École de technologie supérieure
Montreal, Canada
christian.desrosiers@etsmtl.ca

Eric Paquette
École de technologie supérieure
Montreal, Canada
eric.paquette@etsmtl.ca

ABSTRACT

We propose a novel approach based on machine learning to simulate facial expressions related to physical activity. Because of the various factors they involve, such as psychological and biomechanical, facial expressions are complex to model. While facial performance capture provides the best results, it is costly and difficult to use for real-time interaction during intense physical activity. A number of methods exist to automate facial animation related to speech or emotion, but there are no methods to automate facial expressions related to physical activity. This leads to unrealistic 3D characters, especially when performing intense physical activity. This research highlights the link between physical activity and facial expression, and to propose a data-driven approach providing realistic facial expressions, while leaving creative control. First, biological, mechanical, and facial expression data are captured. This information is then used to train regression trees and support vector machine (SVM) models, which predict facial expressions of virtual characters from their 3D motion. The proposed approach can be used with real-time, pre-recorded or key-framed animations, making it suitable for video games and movies as well.

Keywords

Facial Animation - Biomechanics - Physical Activity - Machine Learning.

1 INTRODUCTION

Facial animation remains one of most tricky, time-consuming, and costly aspects of 3D animation. Facial expressions are difficult to model because of the numerous factors underlying them: emotions (joy, sadness, etc.), mouth movements (speech, deep breath, etc.), eye and eyelid movements (blinking, gaze direction, etc.) and physiological (fatigue, pain, etc.).

Different approaches for automating facial expressions related to emotion or speech exist, but none are available to automate expressions related to physical activity. In the visual effects and computer animation communities, facial animations are most often key-framed or motion-captured. Even though this is a relatively long and costly procedure, it is understandable for main characters. For secondary characters, such as a crowd, the facial animation related to physical activity will often be disregarded. In video games, although characters often have to provide significant physical exertion, facial animation related to this component is somewhat neglected. It is sometimes present during cinematic sequences, but it suffers from a crude approximation during gameplay, and it is often simply overlooked. According to discussions we have had with video game companies, current approaches for gameplay facial animations related to physical activity rely on ad hoc techniques based on linear functions and thresholds. Such approaches are far from the complexity of human facial animations, in relation to physical activity. In this paper, we propose a novel approach based on machine learning to simulate facial expressions related to physical activity, in order to improve the realism of 3D characters. The approach is based on the analysis of motion capture data acquired from real exercise sessions. Given the captured animations and physiological data, specific machine learning techniques are selected to enable the synthesis of facial expressions corresponding to physical activity. The main contributions of the proposed approach can be summarized as:

- A machine learning framework to derive facial expressions from physical activity;
- An approach to link mechanical, physiological, and facial measurements;
- An analysis of the most effective way to compute energy values for machine learning purposes;
- A set of empirical rules relating physical activity to specific facial expressions;
- A normalization procedure to make better use of heart rate and blend shape data.

With this machine learning framework and captured data, we are able to synthesize realistic facial expressions. The approach can be used for real-time as well as off-line facial animation. Furthermore, the method allows for control over stylization, as key-framed data could be used instead of captured data. It enables control over expressiveness, as the animator can adjust various parameters that have an impact on the facial expression of the character. Finally, the models developed in this work also provide metabolic data that could be used for purposes other than facial animation.

2 RELATED WORK

Previous works are categorized in four topics: the description of facial expressions, the synthesis of speech-related expressions, the synthesis of emotion-related expressions, and the description of facial expressions related to physical activity.

2.1 Facial Expression Coding

Several objective and systematic approaches to encode facial expressions have been proposed. Although facial expressions are due to a wide range of factors, only facial changes due to emotions, intentions or social communication are taken into account [17]. Various coding systems have been developed mainly for psychological studies, including FACES (Facial Expression Coding System) [14] and FACS (Facial Action Coding System) [8], which are presented in a survey paper [19]. FACS is an anatomically-based expression space grouping together facial muscle groups as AUs (Action Units), whose combination can be used to form any possible expression [26]. The MPEG-4 standard proposes a similar approach using FAPs (Facial Action Parameters), which has been used in various research projects. In the proposed approach, the facial expressions are built using blend shapes that correspond to facial muscle groups similar to the FACS approach.

The automation of the coding process generally relies on video tracking software or motion capture systems that require complicated setup. In recent years, novel techniques emerged using depth cameras such as *FaceShift* [4] or *Brekel ProFace* [6]. Other software use simple webcams, such as *Mixamo Face Plus* [22], *di-o-matic Maskarad* or *Emotient* software [18]. While most of the available software extract a set of facial features or blend shapes, *Emotient* software extracts Ekman's facial expressions and a set of Action Units.

2.2 Speech-Related Facial Expressions

Animation synthesis is generally done by analyzing an audio input, extracting phonemes, and then animating the 3D face model's visemes (phoneme's visual counterpart) [15]. Different approaches have also been developed to enhance realism in animation, such as blending speech-driven animation into emotion-driven animation and using anatomically-based structures [16]. Other works [31] have focused on improving the visual behavior related to speech. Some works use machine learning methods such as SVMs [33] or neural networks [7, 21]. All of these methods help to achieve more realistic results in facial expressions related to speech, and substantially reduce manual animation time. They give good results, given that there is a single input (the audio) that captures all of the required information to adjust the facial animation.

When considering physical activities, a character's motion involves several limbs, as well as potential and kinetic energy, torques, etc., which results in a broader set of inputs. Furthermore, simulating speech-related animation from audio is synchronized to one input signal, while the facial expression of the character's motion might be the result of both its instantaneous motion and the movements or activities performed by the character in the past few minutes. Finally, the character's motion triggers facial expressions that will influence parts of the face that are not related to speech, such as the region around the eyes. For these reasons, works dealing with speech cannot fully solve the problem of generating the facial animation related to physical activity.

2.3 Emotion-Related Facial Expressions

Researchers in psychology studied emotions and came up with classifications based on a limited number of emotions. To further simplify the relationships between emotions, they can be represented in simpler 2D expression spaces [24, 28]. Computer graphics researchers have taken advantage of such approaches and proposed different two dimensional emotion layouts that allow a meaningful blending between emotions [25]. Other approaches have relied on coding systems such as FACS to provide realistic transitions between emotion expressions [1]. While these works provide interesting approaches for the transition and blending of facial expressions, they work when the emotion is already known, and when a set of face poses is provided. Animating the right combination of emotions through time remains a complex problem. It is similar to the challenge involved in this work: developing an approach that can predict the facial expression from observations and models describing how a human subject reacts in different circumstances.

2.4 Physical Activity-Related Facial Expressions

Even though 3D characters often perform intense physical activities, we could not find any research addressing the automatic and realistic facial animation related to physical activity. Outside of the computer graphics field, the work of McKenzie [20] describes the facial expressions related to substantial effort, exhaustion,

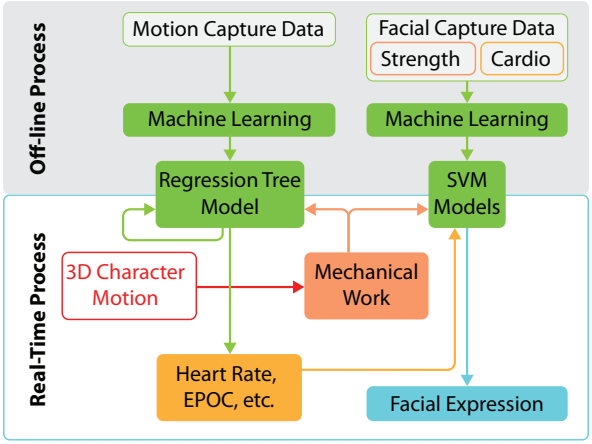


Figure 1: Machine learning facial expressions

and fatigue. In the computer graphics field, facial animation literature related to physical activity is found mostly in art books [9, 11]. These contain numerous facial expressions, some of which are related to physical activity. While these works provide useful information for manually animating expressions, they are not useful for the automatic facial animation from the character's motion. Zordan's work [34] target the modeling and control of 3D models in relation to respiration.

Numerous works address facial animation in the context of speech and emotion, but they are not adapted to the synthesis of physical activity expressions. Based on machine learning and captured data, the proposed approach derives a model, to animate facial expressions.

3 FACIAL EXPRESSION SYNTHESIS

Fig. 1 shows an overview of the proposed approach. The first step is to acquire real-life motion capture data, providing information on the facial expressions observed under various types of exercise. These data are used to train machine learning models, which are then used to generate realistic facial expressions.

3.1 Data Acquisition

Various capture sessions were conducted in order to gather the information required to develop a model that gives realistic results. During these sessions, information describing the type of activities and physical state (heart rate and facial expressions) were recorded.

A full-body motion capture was done in a large room where 15 participants of different age (20 to 46 years old) and training level (0 to 7.5) exercised freely without training devices. The resolution of the motion capture cameras did not allow facial capture along with the full-body capture. Furthermore, using training devices would have led to markers occlusion. For these reasons, the capture was split in two sessions: full-body and facial. The goal of the full-body session was to provide data to establish the relationship between the motion and physiological measures. The participants were asked to alternate between exercises of low and high levels of intensity, and to slow down to ensure that a large range of data was acquired. While participants were training, both full-body motion and heart rate data were recorded. The software used to record the heart rate provided an estimation of other metabolic indicators, such as metabolic energy consumption, breathing rate and EPOC (Excess Post Oxygen Consumption).

The second capture session was done at a fitness center, where 17 participants from the same age groups and training level as the previous session, were asked to exercise on either a cardiovascular training machine or a mixture of strength training machines and free weights. The goal of this capture session was to establish the relationship between motion, heart rate, and facial expression. Again, this capture involved exercising at different levels of intensity and a slow-down period. Using this procedure, the data collected for each exercise included repetitions for the same participant as well as for different participants. Facial expressions were filmed, while heart rate data were recorded following the same procedures and with the same material as during the first session. Together with the height and weight of the participants, the specific loads used with the strength training machines and free weights allowed for a good approximation of the involved work and forces.

3.2 Biomechanical Model

One of the key inputs to both the off-line and online phases of the proposed approach is the mechanical work resulting from the motion. Different methods were evaluated to approximate the work: potential energy, translational kinetic energy, and rotational kinetic energy. Different ways of evaluating the mechanical energies were tested: using the center of mass of the whole character, using the lower/upper body, and computing these values for each limb. Potential energy, translational and rotational kinetic energies were used.

Tests were conducted to find an approach that would be efficient to compute, while providing good results for both the learning and synthesis phases. While separate inputs for each limb intuitively seemed to provide better knowledge about the type of exercise and effort, they resulted in noisy facial animations with blend shape weights that changed too rapidly compared to the real data. An explanation for this phenomenon is that even though there are several captures, the amount of input data is still too small to correctly capture the intricate interrelations between specific limbs and facial expressions. Ultimately, what provided the best result was using the sum of the mechanical work (potential, translational kinetic and rotational kinetic) for all limbs.

3.3 Machine Learning Facial Expressions

To get a better understanding of the underlying mechanisms and relations between the exercises and the facial expressions, a preliminary analysis of the data was conducted. As the relations between the metabolic, mechanical and facial parameters are too complex to model using simple polynomial equations, it made sense to use machine learning. Given the type of captured data and the kind of predictions required, regression techniques were the most appropriate. Several models were trained using different sets of features as input, and the quality of these models was evaluated. Likewise, appropriate model parameters were selected using cross-validation. The data flow, learning approaches and models are presented in Fig. 1.

3.3.1 Metabolic Parameters Prediction

To predict the heart rate from the character's motion, various learning techniques were tested with different combinations of features as input. The heart rate increases or decreases depending on the intensity of the exercise: for each person, there is a certain threshold in exercise intensity that results in an increase or decrease in the demand for oxygen. To model this behavior, regression trees were found to give an accuracy comparable to more complex models such as SVM. Furthermore, this technique was also selected for its ability to provide a human-interpretable model, which can be used to get more artistic control on the final result.

Since the range of input values affects learning techniques, and as the range of heart rates varies among the participants, the data were transformed to the $[0,1]$ interval resulting in the normalized heart rate (NHR):

$$\text{NHR} = \frac{\text{current heart rate} - \text{resting heart rate}}{\text{maximum heart rate} - \text{resting heart rate}}$$

The maximum and resting heart rates are found in standard training charts based on the age and training level.

A regression tree model was built using the UserClassifier in the *Weka* software [12]. Several combinations of inputs were tested (mechanical work as input and heart rate as output, mechanical work and heart rate difference as input and heart rate difference as output, etc.). Among the tested models, the one providing the best results was to predict the difference in heart rate using the current heart rate and the instantaneous mechanical work. By using the last predicted NHR in the subsequent prediction, the model considers the temporal information and the accumulated fatigue.

While a model trained using the data from a single participant could accurately predict the heart rate of this participant (correlation coefficient of 0.88 and root-mean-square error – RMSE – of 19%, see Fig. 2(a, c)), combining the data from every participant in a single model resulted in significant errors (correlation coefficient of 0.21 and RMSE of 79%, see Fig. 2(b, d)). To improve the results, simpler models were derived.

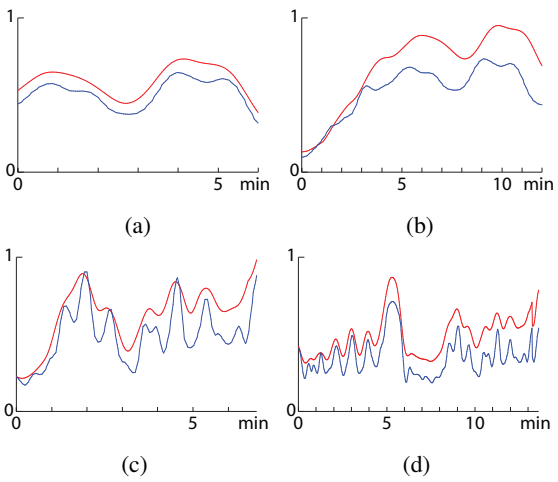


Figure 2: Comparison between real (blue) and predicted (red) NHR as a function of time (in minutes). (a-b) cardiovascular and (c-d) strength training exercises. (a,c) the same and (b,d) different participants.

Using appropriate regularization parameters and tree pruning yielded simple regression trees with only two leaves. By analyzing these simpler models for each participant, a common structure emerged: all regression trees had the same test at the root node, comparing the mechanical work w to a threshold value t_{root}, which broke the predictions into increases or decreases in the heart rate. Moreover, the main difference between these personalized models was the threshold value used in the test, this value depended largely on the fitness level of the participant. Based on these observations, a linear regression between the training level and t_{root} was used to improve the model. The resulting model based on the linear regression and regression tree is as follows:

$$t_{root} = 7.13 + 0.42 \times \text{training level}$$

$$\Delta(\text{nhr}) = \begin{cases} c_{inc} \times (w - t_{root}) \times (1 - \text{nhr})^2 & t_{root} < w \\ c_{dec} \times (w - t_{root}) \times \text{nhr}^2 & w \leq t_{root} \end{cases}$$

$$c_{inc} = 0.0056 - 0.00043 \times \text{training level}$$

$$c_{dec} = 0.0009 + 0.00025 \times \text{training level}$$

The threshold t_{root} determines when the heart rate starts to increase while c_{inc} and c_{dec} are the factors of increase or decrease. These values were obtained by calculating a linear regression between the individual values obtained in the regression tree of each participant.

This model provided a prediction that was almost as good as the one for separate participants (correlation coefficient of 0.87, RMSE of 24%). Furthermore, the training level can be used to control the response level of characters to various types of exercises.

The normalized heart rate is used to approximate the oxygen consumption (VO_2) and respiration rate. Both the VO_2 and respiration rate have to be normalized relatively to their minimal and maximal values. Given the normalized values, the VO_2 and respiration rate are proportional to the normalized heart rate [27]. With the VO_2 estimation, EPOC can be approximated [10]. These estimations, together with the mechanical work and mechanical power, are then used to predict the facial animations as will be described in the next section.

3.3.2 Predicting Expression Components

To animate a virtual character, the four weights corresponding to the blend shapes associated with the basic components identified in the preliminary analysis (see Section 4.1.1) should be obtained from the movement of the character. Compared to the metabolic parameters, the facial expressions in our capture data exhibited more sudden and frequent changes. Because of this behavior, regression trees did not provide adequate results to predict blend shape weights.

Instead, we opted for SVM regression, which provided better prediction results and had already been used successfully for facial animation [32]. Tests were conducted with multiple participants, for multiple exercises as well as for single participant and single exercise (see Fig. 3). For a single participant and exercise, the prediction of the participant's blend shapes corresponding to exercises not used to train the model was accurate (Fig. 3 (a), (b)). Compared to what was observed for the metabolic parameter, the prediction from strength training exercises (Fig. 3 (b), (c), and (d)) lines up quite well with the real data, while the prediction for cardiovascular exercises (Fig. 3(b)) follows the general trend of the curve, but presents variations of smaller amplitude due to the regularization of the model.

For multiple participants, training with a single exercise enabled a good prediction of the same exercise for a participant not used in the training data (Fig. 3(c)). Nevertheless, generalization across all participants and exercises was relatively poor. Again, the data had to be normalized, but this time with respect to the expressiveness of the participant. This can be seen in Fig. 3(c), as the curves are well aligned, but the blend shape weights are on a different scale. Some of the participants could endure incredible exertion with a relatively neutral expression while others depicted pronounced expressions. The expressiveness could not be linked to any of the parameters collected about the participants (age, training level, etc.). It still can be computed for each participants by finding the maximal weight for each blend shape, resulting in a four-dimensional expressiveness vector. The captured blend shape weights of the participants were then normalized. This expressiveness vector allows to control the facial expressions by increasing or reducing the expressiveness values.

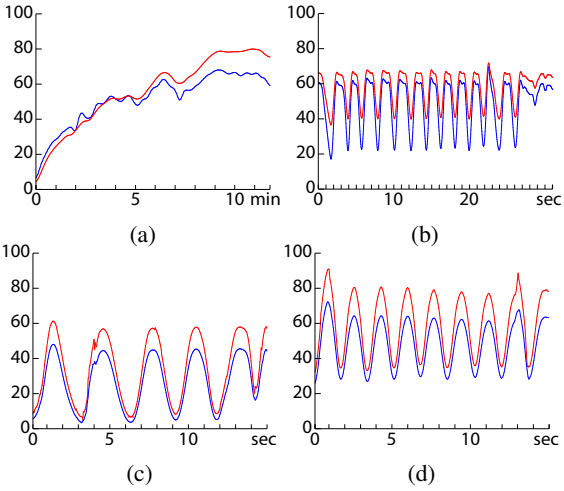


Figure 3: In each case, the acquired data (blue) was not used to train the model for the prediction (red) of the blend shape weight. In (a), (b), and (c), the prediction and SVM model are for the same exercise, while in (d) we tested the prediction for an exercise not used to train the SVM model. The prediction works for (a) cardio and (b) strength training exercises. The motion of a participant not used to construct the SVM model is used for the prediction in (c). In (d), the prediction is for an exercise not used to train the SVM model.

Given these normalized blend shape weights, the whole dataset could be learned using an SVM. To select the best-suited set of inputs and model parameters, several combinations were evaluated on a test data set and using cross validation. The combination that gave the best results was mechanical work, mechanical power, normalized heart rate and EPOC as inputs, and blend shape weight as output. As the predicted heart rate is used as an input for the next prediction (see Fig. 1), the models consider the temporal information and do not only model the correlation at a single-frame level.

With respect to the selection of the SVM parameters, the radial basis function (RBF) kernel was selected, and several combinations of parameters were tested: regularization parameter and gamma of the RBF varying independently from 10^{-10} to 10^{10}. The values of these parameters that produced the best results were different from one blend shape to the other. The regularization parameter ranged from 10^3 to 10^6 while the gamma of the RBF ranged from 10^{-5} to 10^{-2}.

Since different areas of the face reacted in different ways depending on the physical activity and the participant, the predictions use four SVM models. Although these are independent, the predictions were consistent in all of our tests. The training RMSE of the models was in the $[17\%, 26\%]$ range. The final models enabled a good generalization of the captured data, which indicates that our method could be used to generate realistic facial animations on other types of movements.

(a) $\frac{\text{Mouth}}{\text{Exertion/EPOC}}$ (b) $\frac{\text{Mouth}}{\text{Breathing/EPOC}}$ (c) $\frac{\text{Eyes}}{\text{Exertion/EPOC}}$ (d) $\frac{\text{Eyes}}{\text{Fatigue}}$ (e) $\frac{\text{Eyebrows}}{\text{Exertion}}$ (f) $\frac{\text{Eyebrows}}{\text{Fatigue}}$

Figure 4: The physical effort resulted in a broad range of facial expressions

(a) (b) (c)

Figure 5: Results for five minutes of running at medium speed with different training levels: (a) 10, (b) 5, (c) 0

4 RESULTS

In this section, the approach and its experimental evaluation are discussed in the context of the expected use of the models. The accompanying video presents results for different types of exercises. The results present believable expressions based on the physical activity of the character, and these expressions improve the realism of the 3D character (see Fig. 5).

The learned regression tree and SVM models were used to animate facial expressions of a 3D head model. Using the LibSVM implementation to perform the predictions showed that the approach can easily achieve real-time frame rates. On an average computer, 60 to 600 FPS were obtained depending on the render settings and mesh complexity. The prototype can be used either by triggering pre-recorded animation clips or by using a live input from a Kinect device. The user can specify the age, height, weight, fitness level and expressiveness vector of the 3D character, and can also add weights lifted in each hand as well as on the back. The facial expressions change relatively to the motion of the character and the specified parameters. The user can change the character parameters and get a real-time feedback. Fig. 5 shows different results achieved with different training levels and Fig. 6 shows different results obtained by changing the expressiveness vector.

4.1 Discussion

4.1.1 Observations on the Captured Data

A qualitative analysis of the captured data (full-body capture, video and heart rate) was conducted. While these observations helped us in the selection of the appropriate machine learning methods, they should also benefit artists in animating 3D characters. Two types of relations were discovered: general relations that hold for most facial expressions, and specific relations that apply to particular expressions.

The first general rule is that the intensity of expressions is proportional to the displaced mass and inversely proportional to the mass of the muscle. The expression related to the instantaneous exertion is proportional to the mechanical power. The evolution of the expression intensity is proportional to the change in heart rate and metabolic energy. Finally, the recovery time is proportional to the effort intensity and inversely proportional to the training level.

Rules specific to individual components of facial expression were also observed. It was determined that the facial expression features associated with physical activity were concentrated around the eyes and the mouth (see Fig. 4). Regarding the expressions related to the mouth, the stretching is induced by two factors: instantaneous physical exertion and fatigue level. The mouth remains closed at the beginning of the training session. After a certain time, it starts to open, and the opening is linked to the respiration rate and the fatigue level.

Other observations were related to the region of the eyes. Eye squinting is mainly induced by instantaneous physical exertion until a certain fatigue level. At a higher level of fatigue, the eyes tend to relax in connection with the fatigue level with a remaining constant squinting value. The behavior of the eyebrows is a combination of a downward movement related to the physical exertion and an upward movement related to fatigue.

Finally, some observations were made with respect to both the breathing and the swallowing. The frequency of occasional breathing movements related to loud and quick expiration is induced by two factors: fatigue level and respiration rate. The frequency of the occasional gulping is proportional to the fatigue and to the regular respiration rate. These observations helped in defining blend shape selection greatly inspired by the muscular groups of the human face, as described in FACS [8]. The model consisted of a neutral face and four blend shapes that can be used in various expressions linked to physical activity (see Fig. 7).

4.1.2 Manual Blend Shapes Recovery

To simplify the capture sessions, only a video recording of the face was used for the facial expression capture.

(a) (b) (c) (d) (e)

Figure 6: Results obtained with different expressiveness vectors: (a-b) softened expressions, (c) predicted expression, and (d-e) exaggerated expressions. The red lines are added as guides to help in comparing the expressions

(a) Neutral (b) Mouth Stretch (c) Mouth Opening (d) Eyes Squinting (e) Eyebrows

Figure 7: The four blend shapes used in our implementation.

Different techniques were evaluated for retrieving facial animation data, but it was found that manual animation provided more reliable results. To recover objective values that can be used with machine learning approaches, a virtual character's face was key-frame animated to match the expression of the participants. To ensure the results are reproducible, blend shapes were key-framed, one at a time, and always in the same order. Furthermore, to measure how perceptually meaningful the values were, three different people independently adjusted the blend shape weights for a selection of eighteen representative poses. Even though the blend shape weights were not identical, the error remained limited to 11% on average and was considered to be quite sufficient for the purposes of this work.

4.1.3 Limitations

As shown in Fig. 7, the blend shape model used in this work is sufficient, but it does not cover the whole range of expressions. Since each blend shape is predicted separately, there could be inconsistencies in the face of the character. Resolving such inconsistencies and providing a better correspondence could be achieved through a constrained weight propagation [23]. While the mesh deformation used in this paper is based on blend shapes, the models could be learned with the use of other control mechanisms, such as bone systems.

The generated facial expressions are generalizations of the observed data. They correspond to mean values and sometimes lack expressiveness (see Fig. 8). The models sometimes output results that deviate significantly from the observations. As they happen quite infrequently, they can be easily filtered out.

The metabolic prediction model uses its last prediction as input. It is thus subject to error accumulation and could diverge from the observed values over time. Approaches to steer the values back to the observed range should be used to solve such problems.

(a) (b) (c)

Figure 8: Comparison between real and predicted facial expression from one participant to another: (a) generated, (b-c) different participants doing the same exercise.

5 CONCLUSION

By analyzing two sets of captured data, this paper reveals several important observations about what triggers specific facial expressions. A combination of two machine learning techniques was used in order to automatically synthesize some metabolic parameters as well as the facial animation of a 3D character. While being automatic, this approach provides meaningful parameters that animators can change to deliver realistic and compelling facial animations that automatically adjust to the motion of the character. Furthermore, the metabolic parameters provided by the approach could also be helpful in animating other aspects of the character, such as breathing and sweating. Finally, the approach can be used for real-time applications as well as

off-line high quality rendering. The approach provides more realistic characters while reducing the burden of capturing or hand animating the facial expressions resulting from physical activity.

As the manual blend shape animation was a time consuming process, the method was limited to four blend shapes. Automating this process by using novel techniques [5, 29, 30] would allow for a larger number of blend shapes or Ekman's AU's by using the *Emotient* software for even more realistic results. Like other methods described in Section 2, the proposed approach addresses a single aspect of facial animation (only from physical activity). A future work would be to provide a framework that allows mixing different types of expressions through various methods [2, 3, 13]. The proposed approach is deterministic in nature: given the same control parameters and motions, it will result in the same facial animation. An interesting future research would be to incorporate the probabilistic and stochastic nature of human reactions into the models.

6 ACKNOWLEDGMENTS

This work was funded by the GRAND NCE.

7 REFERENCES

[1] A. Arya and A. DiPaola, S. Parush. Perceptually valid facial expressions for character-based applications. *Intl. Journal of Computer Games Technology*, 2009.

[2] T. Beeler, F. Hahn, D. Bradley, B. Bickel, P. Beardsley, C. Gotsman, R. W Sumner, and M. Gross. High-quality passive facial performance capture using anchor frames. *ACM Trans. Graph.*, 30(4):75, 2011.

[3] A. H. Bermano, D. Bradley, T. Beeler, F. Zund, D. Nowrouzezahrai, I. Baran, O. Sorkine-Hornung, H. Pfister, R. W Sumner, B. Bickel, and M. Gross. Facial performance enhancement using dynamic shape space analysis. *ACM Transactions on Graphics (TOG)*, 33(2):13:1–13:12, 2014.

[4] S. Bouaziz and M. Pauly. Dynamic 2d/3d registration for the kinect. In *ACM SIGGRAPH 2013 Courses*.

[5] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *Visualization and Comp. Graph, IEEE Trans. on*, 20(3):413–425, March 2014.

[6] J Chow, K Ang, D Lichti, and W Teskey. Performance analysis of a low-cost triangulation-based 3d camera: Microsoft kinect system. In *Intl. Society for Photogrammetry and Remote Sensing Congress (ISPRS)*, volume 39, pages 175–180, 2012.

[7] P. Eisert, S. Chaudhuri, and B. Girod. Speech driven synthesis of talking head sequences. In *3D Image Analysis and Synthesis*, pages 51–56, 1997.

[8] P. Ekman and W.V. Friesen. *Facial action coding system: investigator's guide*. Consulting Psychologists Press, 1978.

[9] G. Faigin. *The Artist's Complete Guide to Facial Expression*. Watson-Guptill Publications, 1990.

[10] Firstbeat Technologies. Indirect epoc prediction method based on heart rate measurement. White paper.

[11] E. Goldfinger. *Human Anatomy for Artists: The Elements of Form*. Oxford University Press, 1991.

[12] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11:10–18, November 2009.

[13] M. Kapadia, I. Chiang, T. Thomas, N. I. Badler, and J. T. Kider, Jr. Efficient motion retrieval in large motion databases. In *Proc. of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pages 19–28, 2013.

[14] A.M. Kring and D.M. Sloan. The facial expression coding system (faces): Development, validation, and utility. *Psychological Assessment*, 19(2):210–224, June 2007.

[15] S. Kshirsagar and N. Magnenat-Thalmann. Visyllable based speech animation. *Computer Graphics Forum*, 22(3):631–639, 2003.

[16] S. Kshirsagar, T. Molet, and N. Magnenat-Thalmann. Principal components of expressive speech animation. In *Proc. of Computer Graphics International 2001*, pages 38–44, 2001.

[17] S.Z. Li, A.K. Jain, Y.-L. Tian, T. Kanade, and J.F. Cohn. Facial expression analysis. In *Handbook of Face Recognition*, pages 247–275. Springer New York, 2005.

[18] M. Malmir, D. Forster, K. Youngstrom, L. Morrison, and J. R. Movellan. Home alone: Social robots for digital ethnography of toddler behavior. In *Computer Vision Workshops (ICCVW), 2013 IEEE Intl. Conf. on*, pages 762–768, 2013.

[19] I.B. Mauss and M.D. Robinson. Measures of emotion: A review. *Cognition & Emotion*, 23(2):209–237, 2009.

[20] R.T. McKenzie. The facial expression of violent effort, breathlessness, and fatigue. *Journal of anatomy and physiology*, 40:51–56, October 1905.

[21] W. Zhen P. Hong and T.S. Huang. Real-time speech-driven face animation with expressions using neural networks. *Neural Networks, IEEE Transactions on*, 13(4):916–927, Jul 2002.

[22] C. Piña, E. Gambaretto, and S. Corazza. Live real-time animation leveraging machine learning and game engine technology. In *ACM SIGGRAPH 2014 Talks*.

[23] L. Qing and D. Zhigang. Orthogonal-blendshape-based editing system for facial motion capture data. *Computer Graphics and Applications, IEEE*, 28(6):76–82, 2008.

[24] J.A. Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178, 1980.

[25] Z. Ruttkay, H. Noot, and P. Ten Hagen. Emotion disc and emotion squares: Tools to explore the facial expression space. *Computer Graphics Forum*, 22(1):49–53, 2003.

[26] M. Sagar. Facial performance capture and expressive translation for king kong. In *ACM SIGGRAPH 2006 Sketches*, 2006.

[27] D.P. Swain and B.C. Leutholtz. Heart rate reserve is equivalent to %vo2 reserve, not to %vo2max. *Med Sci Sports Exerc*, 29:410–4, March 1997.

[28] R.E. Thayer. *The biopsychology of mood and arousal*. Oxford University Press, 1989.

[29] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Kinect-based facial animation. In *SIGGRAPH Asia 2011 Emerging Technologies*.

[30] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. *ACM Trans. Graph.*, 30(4):77:1–77:10, July 2011.

[31] Y. Xu, A. W. Feng, S. Marsella, and A. Shapiro. A practical and configurable lip sync method for games. In *Proc. of Motion on Games*, MIG '13, pages 109:131–109:140, 2013.

[32] P. Faloutsos Y. Cao, W.C. Tien and F. Pighin. Expressive speech-driven facial animation. *ACM Trans. Graph.*, 24:1283–1302, October 2005.

[33] Y.Q. Xu E. Chang H.Y. Shum Y. Li, F. Yu. Speech-driven cartoon animation with emotions. In *Proc. of ACM Intl. Conf. on Multimedia*, MULTIMEDIA '01, pages 365–371, 2001.

[34] V. B. Zordan, B. Celly, B. Chiu, and P. C. DiLorenzo. Breathe easy: Model and control of human respiration for computer animation. *Graph. Models*, 68(2):113–132, March 2006.

Image-based Object Modeling by Fitting Salient Lines and Geometric Primitives

Meng-Hong Cho
Inst. of Multimedia Engineering,
National Chiao Tung University
Hsinchu City, Taiwan
combo1966@gmail.com

I-Chen Lin
Inst. of Multimedia Engineering and
Dept. of Computer Science,
National Chiao Tung University
Hsinchu City, Taiwan
ichenlin@cs.nctu.edu.tw

ABSTRACT

With modern vision techniques and depth sensing devices, it becomes possible for common users to acquire the shape of an object from a set of color or depth images from different views. However, the estimated 3D volume or point clouds, disturbed by noise and errors, cannot directly be applied for graphics usage. This paper presents a two-stage method for reconstructing 3D graphics models from point clouds and photographs. Unlike related work that immediately fitted primitives for the point clouds, we propose finding the primary planes through salient lines in images in advance, and extracting auxiliary planes according to the symmetric properties. Then, a RANSAC method is used to fit primitives for the residual points. Intuitive editing tools are also provided for rapid model refinement. The experiments demonstrate that the proposed automatic stages can generate more accurate results. Besides, the user intervention time is less than that by a well known modeling tool.

Keywords

Image-based modeling, primitive fitting, line features.

1 INTRODUCTION

Reconstructing 3D models is an essential and important topic in the computer graphics and vision fields. The reconstructed models can be used for various applications, from object analysis, interior and urban planning to visual special effects, and so forth. With the rapid advances in depth estimation techniques, users can now purchase consumer-level depth cameras, such as Microsoft Kinect, ASUS Xtion Pro. Software tools, such as ARC 3D, 123D Catch, and my3Dscanner, can help users obtain their three-dimensional models by taking a set of photographs. These tools mainly employ structure-from-motion technologies to compute camera poses (extrinsic parameters) and 3D point clouds of a target object. However, these point clouds usually contain holes and defects resulted from noise, view occlusion, and inaccurate point correspondences. They require additional processing or considerable manual adjustment before applied to graphics usage, e.g. model editing, graphics rendering, and 3D printing.

Primitive fitting is a popularly used approach for modeling from point clouds. It decomposes point clouds into multiple parts and finds the most likely primitive fit for each part. The modern primitive fitting performs satisfactorily for accurate laser-scanned point clouds. Nevertheless, when we applied primitive fitting to point clouds generated by photogrammetric methods, the larger noise makes the fitting process of higher degrees of ambiguity, resulting in unstable and noise-sensitive results. From another aspect, Debevec et al. [Deb96a] involved more user intervention and proposed a pioneering architectural modeling method. They required users to compose a target scene by rough geometric models, and to draw corresponding lines from images. Their system then adjusted the geometric models by aligning the lines of primitives with lines assigned by users.

Our work is inspired by the aforementioned approaches, but we would like to lessen user intervention and lower the chances of ambiguity in fitting. We consider that the lines and planes are valuable cues in modeling of man-made objects, and propose a novel two-stage modeling method. Given a set of photographs and corresponding 3D point clouds, the proposed system first extracts salient lines from photographs, and then finds planes which are associated with these lines and conform to our objective rules. While taking these planes as foundation, the ambiguity situations are alleviated. Partial primitive fittings

 ISBN 978-80-86943-65-7

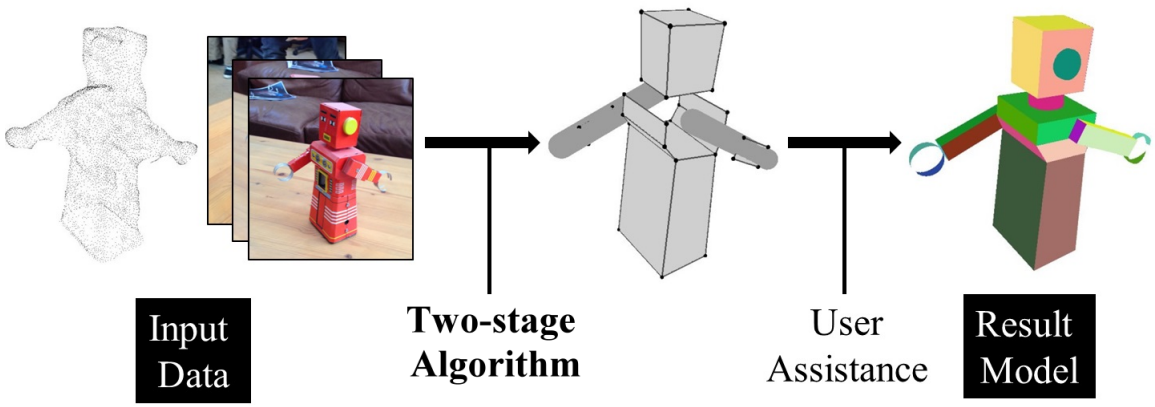


Figure 1: Overview of the proposed method.

is then applied to estimate the residual regions. As shown in Fig.1, the proposed system automatically generated models by fitting salient lines, planes and partial primitives. It allows user assistance to refine the reconstructed model. The user study demonstrates that the proposed method requires less user intervention time than that by a known modeling tool.

2 RELATED WORK

Reconstructing three-dimensional geometric models has attracted considerable interests. This section reviews research on model reconstruction from images and point cloud.

Cipolla and Robertson [Cip99a], Werner and Zisserman [Wer02a], and Schindler and Bauer[Sch03a] used multiple images as input, and generated a coarse model consisting of orthogonal planes. The properties of vanishing points are applied to obtain planes and camera parameters. Snavely et al. [Sna06a] proposed a Photo tourism system. It computed the view points of photographs, and found the correspondences between reconstructed 3D points and images. Izadi et al. [Iza11a] proposed a GPU pipeline to track and fuse a sequence of the point clouds into a volume structure. The estimated 3D scenes are of less noise and can be used for augmented reality (AR) applications. Nguyen et al. [Ngu13a] focused on high-resolution texture mapping for image-based modeling. They projected best-fitting images onto surface segments and merged the segments by using a graph-cut method.

Li et al. [Li11a] introduced a method to reconstruct a man-made object from scanned point sets. They focused on discovering global relations among parts of model to correct primitives, and applied several iterations of RANSAC fitting [Sch07a] and constrained optimization. Lafarge and Alliez [laf13a] adopted a two-step method for point-cloud-based modeling. They first detected a set of planar primitives from the input point set, and then they used min-cut formulation to reconstruct the surface of model. By contrast, our method extracted image salient lines and associated planes and combined a RANSAC fitting framework. Nan et al. [Nan10a] proposed an interactive modeling system for buildings with repetitive structural elements. Arikan et al. [Ari13a] introduced an intelligent snapping algorithm to best fit the input data and maintain the planarity of the polygons.

Several related articles compared different image-based methods or tools. Azevedo et al. [Aze09a] [Aze10a] analyzed the accuracy of volume-based modeling and shape-from-motion from multiple-view images. Bernardes et al. [Ber14a] discussed using image-based modeling tools as a replacement of laser scanning in the archaeological process.

3 PREPROCESSING AND OVERVIEW

The inputs of the proposed system are photographs and point clouds of a target object. The point clouds can be estimated by different active or passive vision techniques. In this paper, we acquired point clouds and photographs from the Autodesk 123D Catch gallery [Aut14a]. In the preprocessing stage, we currently utilized the Grabcut method [Rot04a] to extract foreground objects from the input photographs. The user intervention in preprocessing can be further lessened by using recent segmentation methods, e.g. [Lin15a]. The line segment detector (LSD) method [Von10a] is used to obtain line segment features from the foreground regions of photographs and masked images. Based on the epipolar geometry and constraint, our system finds the potential correspondences of line segments between different views, and it converts the two-dimensional line segment pairs into three-dimensional line segments. Besides, 3D line segments that are far from the point clouds are removed from the valid set. Fig. 2 depicts the result of this step.

The proposed system consists of two automatic stages, *selection of planes associated with salient lines* and *residual point cloud fitting with primitives*. After two

automatic processing stages, a friendly interface is provided for rapid user-assistance refinement. Fig. 3 shows the flowchart of the automatic stages. Section 4, 5 and 6 present the L-plane selection, primitive fitting and refinement stages, respectively.

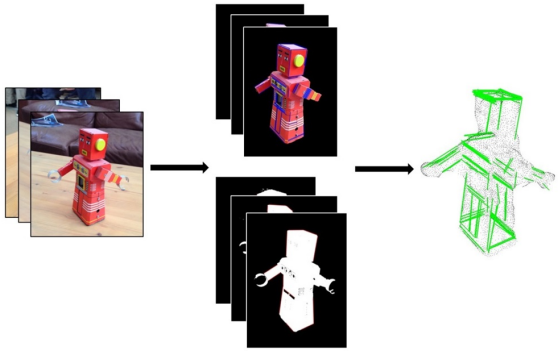


Figure 2: Result of salient line feature extraction.

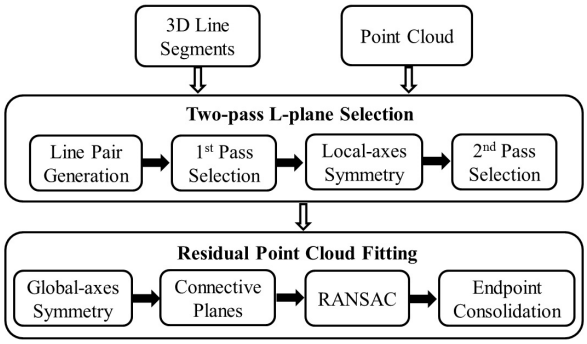


Figure 3: The flowchart of the automatic modeling stages.

4 FUNDAMENTAL FRAME MODELING WITH L-PLANES

4.1 Finding valid L-planes

We use the term *L-plane* to concisely represent a quadrangular plane derived from two or more three-dimensional salient lines. As shown in Fig. 4, either parallel or perpendicular line segments can form an L-plane candidate. The next step is to form a valid L-plane set from the naive candidates. We found the overlap ratio between an L-plane candidate and points of point clouds is useful for this task. For a point, we project it onto an L-plane candidate. If the projection (point-to-plane) distance is within a predefined range, we inspect whether the point is inside the L-plane region, as shown in Fig. 5. For an L-plane, the surface area that is covered by the input point clouds is named $Area_{points}$. Since there are always noise and measurement errors, in practice, we divided an L-plane area into regular cells, and evaluated the sum of area of cells that have close and valid point projections.

For each L-plane, we compute the filled ratio between the surface area of included points $Area_{points}$ and the rectangular area of L-plane $Area_{rect}$:

$$FilledRatio = \frac{Area_{points}}{Area_{rect}} \quad (1)$$

We exclude candidates of which *FilledRatio* values are less than a threshold and form the valid L-plane set.

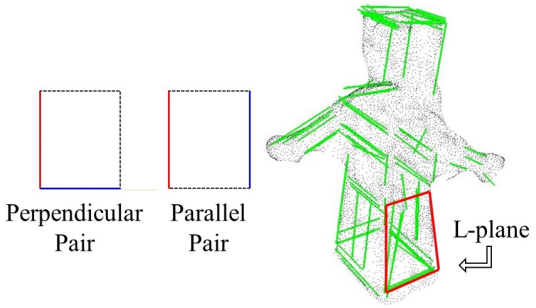


Figure 4: L-planes from pairs of salient lines.

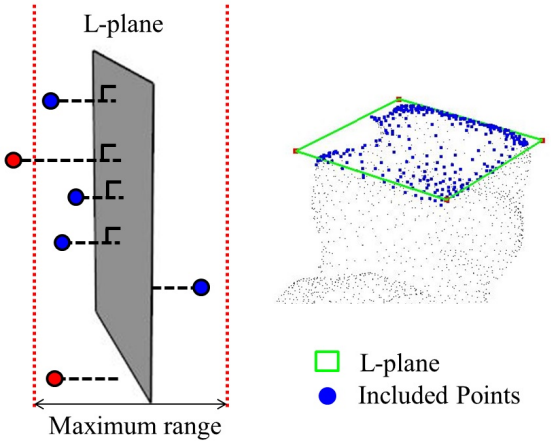


Figure 5: Check the coverage ratio between an L-plane candidate and point clouds.

4.2 First-pass selection with an objective function

The valid L-plane set is derived from salient lines in photographs, which can be influenced by texture on the object, discretized error, and other noise. An L-plane extracted in the above subsection is possibly overlapped with others, and therefore, we have to further extract a more compact L-plane set for the fundamental frame of a target model.

Before we introduce our objective function, we first measure the quality of a valid L-plane by its distance to point clouds. Assume an $L\text{-}plane_i$ includes N_i valid projection points. The projection distance between the included points V_j to the $L\text{-}plane_i$ is named $Dist(V_j, L\text{-}plane_i)$ (as the dashed lines in Fig. 5). The quality of

 ISBN 978-80-86943-65-7

an $L\text{-}plane_i$ is defined as the average of valid point-to-plane distances.

$$Quality_i = \frac{1}{N_i}\sum_j Dist(V_j,\ L\text{-}plane_i) \qquad (2)$$

We propose an objective function to select a set of L-planes, S, which covers the maximum surface area of model accurately (of high quality) and with less overlap. The objective function E_{sel} is defined as:

$$E_{sel} = Area_{ent} - \sum_{s\in S}(Area_s - \omega \cdot Quality_s), \qquad (3)$$

where $Area_{ent}$ is the surface area of the entire point cloud set, $Area_s$ represents the covered surface area of $L\text{-}plane_s$, and ω is a weight to adjust the importance of quality. $Quality_s$ in Eq.3 can be regarded a penalty of fitting error for plane s. The optimum is substantially a complex combinatorial problem. In this paper, we adopt a heuristic algorithm to efficiently approximate this problem. At each iteration, the proposed system finds a single L-plane s that can minimize the E_{sel} value, and it then removes the $Area_s$ from $Area_{ent}$. The selection procedure continues until there is no adequate L-plane.

4.3 Second-pass selection by local-axes symmetry

Since symmetry characteristics occur in a large portion of man-made objects, we intend to find more valuable L-planes (which may be missed during salient line extraction) according to symmetry of the primary L-planes selected above.

Given a pair of primary L-planes which are parallel or perpendicular to each other, we evaluate their local symmetric center and axis. This pair of L-planes are then flipped and shifted to check whether there are rotational or mirror symmetry situations. The filled ratio in Eq. 1 is again used to verify the new L-plane candidate. For error tolerance, when an L-plane is flipped or shifted, the proposed system also moves it backward and forward within a small offset, and takes the position with the highest filled ratio as the new L-plane. Fig. 6 shows the symmetric L-plane candidates derived from the primary L-planes.

With a set of symmetric L-plane candidates, We also apply the objective function in Eq.3 to extract a set of auxiliary L-planes. The entire area $Area_{ent}$ for the second pass selection is now the final value of the first pass selection. The symmetric L-planes and results of the two-pass selection are shown in Fig. 7.

5 RESIDUAL POINT CLOUD FITTING

By the two-pass selection, we have estimated the fundamental frame of the point cloud. However, there

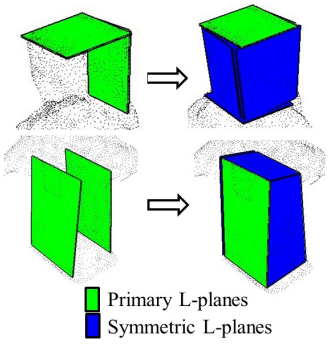


Figure 6: Symmetric L-plane candidates.

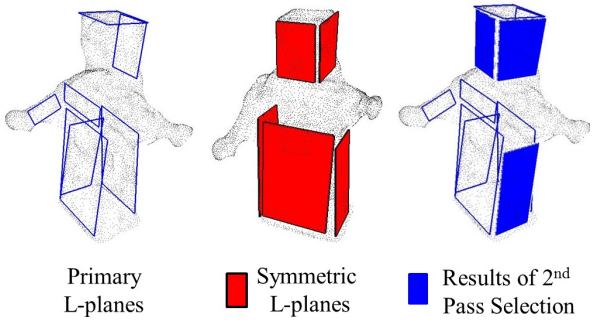


Figure 7: Symmetric L-planes and results of two-pass selection.

are still points uncovered by the selected L-planes. To fit these residual points, this stage further utilizes two fitting procedures, which are L-planes by global-axes symmetry, and RANSAC primitive fitting, and adjusts polygon endpoints.

5.1 Extension with global-axes symmetry

In 4.3, we have exploited the local symmetry for L-plane finding. After getting the fundamental frame of point clouds, we would like to find its global axes, and use them to acquire more symmetric L-planes. One common thought for retrieving the global axes is using principal component analysis (PCA). However, as the robot case in Fig. 7, the second principal axis is roughly along the arm direction. By contrast, users usually regard the normals of building façades or pedestals as the axes of an object. Therefore, we apply box fitting on selected L-planes. A box is formed by at least three L-planes. The plane normals, lengths and distances of adjacent edges are used to determine whether these L-planes are adequate to form a box. We then estimate the box center and the average normals of every two opposite planes to form possible global axes of symmetry. Fig. 8 shows an example of box fitting of selected L-planes. Extended L-planes extracted by global-axes symmetry are shown in Fig. 9.

5.2 Plane connection

Since the L-planes obtained so far may not totally cover the whole model, connective planes are generated to fill

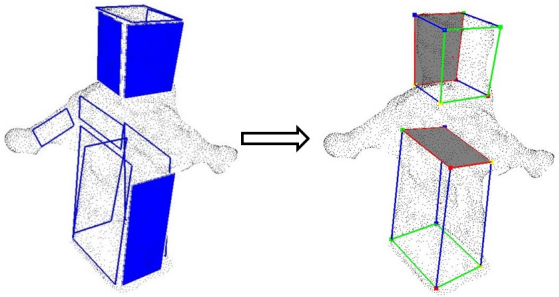


Figure 8: Box fitting of selected L-planes.

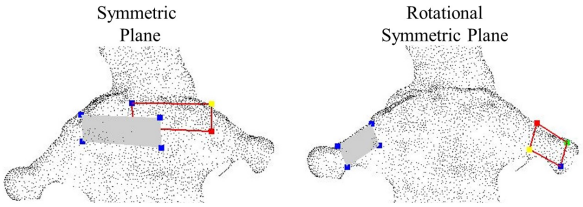


Figure 9: Examples of mirror and rotational symmetric planes by global axes.

small gaps between between two L-planes. The connection process is activated when two selected L-planes have similar plane normals, their distance is smaller than a distance T_{dis}, and their adjacent edges have similar lengths. Examples of connective planes are shown in Fig. 10.

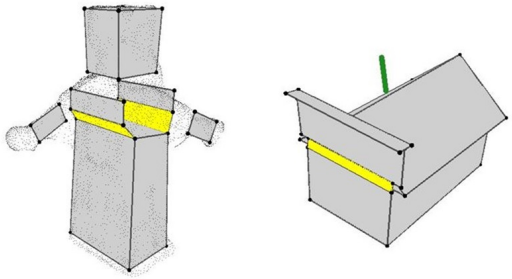


Figure 10: Examples of connective planes.

5.3 Primitive fitting by RANSAC

In this step, the state-of-the-art RANSAC-based method [Sch07a] is applied for fitting the residual point cloud. This local RANSAC-based approach can detect basic shapes, including plane, cylinder, sphere, cone and tori from unorganized point clouds. Nevertheless, our experiment found that the later three primitives were relatively unstable in fitting, and therefore, we adopted using only the plane and cylinder primitives.

Since our photogrammetry-based input point clouds are sometimes seriously disturbed by noise, the results of RANSAC detection cannot fit the target boundary accurately. Additional processes have to be applied for extracting the vertices of primitives from a subset point cloud estimated by the RANSAC method. First, we exploit the Delaunay triangulation to rearrange the mesh of subset points. Second, we apply geometric α-shape for finding the boundary points. Then, boundary points of included angles smaller than a threshold are set to be vertices. Fig. 11 shows an example of primitive boundary estimation.

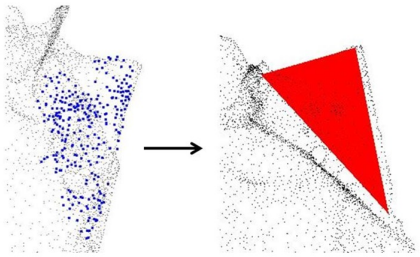


Figure 11: Primitive boundary estimation from a point cloud subset.

5.4 Endpoint consolidation

The last step in automatic modeling is to merge close endpoints of L-planes and primitive planes together. As mentioned in 5.1, we have found boxes of L-planes as candidates of global axes. In this step, the proposed system also first merges vertices around the box vertices, and then merges other points. Fig. 12 shows an example.

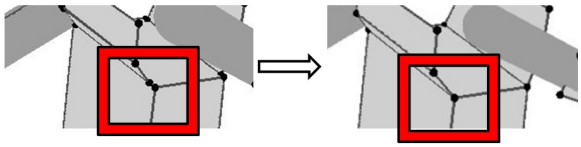


Figure 12: Examples of endpoint consolidation.

6 USER-ASSISTANCE REFINEMENT

Even though fitting the L-plane frame first improves the latter results of primitive fitting, users may still want to adjust the automatically estimated model. Hence, the proposed system also provides users with a friendly interface for rapid model editing. As shown in Fig. 13, users can turn polygons to the transparent modes and align the vertices with input photographs. As shown in Fig. 14, users can simply draw a two-dimensional stroke on the screen to select two edges and our system will predict possible operations, e.g. creating a new plane, creating a plane and its symmetry, in a row of suggestive operations for users to select. Users can also select a polygon as the working plane and insert a new primitive, such as a cylinder or sphere, aligned with the plane. The parameters of primitives, e.g. sizes, lengths, types, can also be adjusted in the control panel. Please refer to the supplementary video in which the user interface, editing processes and results are demonstrated.

 ISBN 978-80-86943-65-7

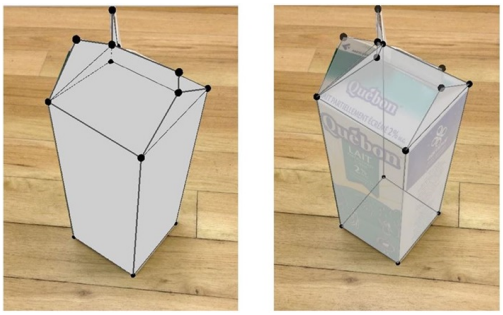


Figure 13: Overlapping the model and photograph on interface.

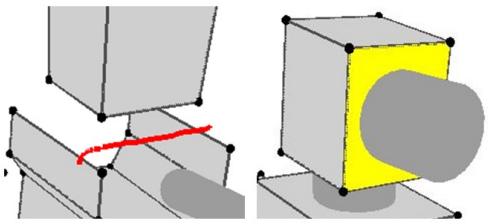


Figure 14: Generating a new plane by a stroke and a new cylinder on the selected working plane.

7 EXPERIMENTS

7.1 Result comparison

To demonstrate the effectiveness of the proposed method, we apply a state-of-the-art RANSAC method by Schnabel et al. [Sch07a] and the proposed method to several data sets from the 123D Catch gallery. As shown in Fig. 15, if we directly apply the RANSAC method, the results can be distorted due to the noise-disturbed point clouds. When the boundary of components in the target is not clear in point clouds, they cannot be accurately extracted by a RANSAC method. For instance, the signboard and frontal face of the house are considered a single plane in front of the house by direct RANSAC, and the top of the drink carton is fitted by quadrangle and triangles. By contrast, the proposed method estimated the L-plane frame in advance and the results are closer to users' expectation.

7.2 Texture mapping

In order to obtain the texture map for each polygon, the proposed system chooses the input photos in the most frontal view and transfers their texture. We also apply the perspective correction for the photos of large view angles. Examples about models with texture mapping are shown in Fig. 16.

7.3 Computation time

The content complexity of input photographs substantially influences the computation time. When a target object in photos has complicate patterns, such as

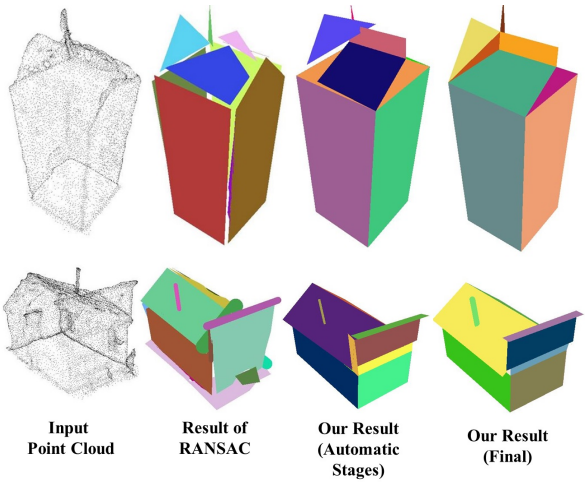


Figure 15: Comparison between the results of advanced RANSAC[Sch07a] and the proposed method.

Figure 16: Texture mapping on the result models.

wooden texture or nets, the proposed system retrieves more salient lines from images and constructs more 3D line segments for L-plane estimation, and therefore prolongs the computation time. The computation time of each result model is listed in Table 1. Fig. 17 demonstrates the results generated by the proposed automatic stages and final results with user refinement. The experiment was performed on a desktop computer with an Intel i7 3.4GHz CPU and 16GB RAM. Currently, a large portion of the system is developed in a single thread. OpenMP [Omp14a] library is applied for searching corresponding line segments in parallel; OpenGL [Ogl14a] and GLUI [Glu14a] libraries are applied for interface development.

7.4 User study

To evaluate how easily users can use our interface to refine automatically estimated models, we conducted user study with five volunteers. All participants received ten-minute demonstration and practice about our interface. Each user had to edit four models estimated by the proposed automatic stages. Table 2 lists the editing time of each user. Moreover, to compare the proposed modeling procedure with popular modeling methods, we also asked users to reconstruct these four

Model	Image num.	Point num.	Auto stage	User refine.
House	24	13297	65.81 sec	$\approx$ 4 min
Robot	15	6155	27.31 sec	$\approx$ 5 min
Mailbox	22	6045	33.44 sec	$\approx$ 4 min
Bigben	37	14526	281.1 sec	$\approx$ 12 min
Drink	23	8310	48.64 sec	$\approx$ 3 min
Lomo	21	13522	45.191 sec	$\approx$ 8 min

Table 1: Information of each result model shown in Fig. 17. The columns from left to right: model name, input image number, input point number, computation time in the automatic stage, and user refinement time

Ours	Mailbox	Bigben	Drink	House
User 1	3 min	6 min	2 min	6 min
User 2	3 min	6 min	2 min	2 min
User 3	3 min	8 min	3 min	2 min
User 4	2 min	6 min	1 min	3 min
User 5	3 min	7 min	3 min	2 min
Average	2.8 min	6.6 min	2.2 min	3 min

Table 2: Editing time of each user by using our refinement interface.

models by using Google SketchUp Make. Similarly, users received a ten-minute video demo and practice about SketchUp interface, and they were required to build the same four models. Table 3 lists the editing time by using SketchUp Make.

These participants had never used our interface and the SketchUp Make software. Since our automatic stages provided approximate models, users just needed fewer simple operations to complete the models. According to users' comments on the SketchUp Make, they can rapidly create a simple building model, but it is relatively not intuitive for modeling several specific parts, such as the roof region of the drink (carton) model.

	Mailbox	Bigben	Drink	House
User 1	9 min	9 min	8 min	9 min
User 2	16 min	9 min	13 min	9 min
User 3	7 min	15 min	8 min	13 min
User 4	8 min	9 min	10 min	7 min
User 5	8 min	11 min	13 min	10 min
Average	9.6 min	10.6 min	10.4 min	9.6 min

Table 3: Editing time of each user by using SketchUp Make.

8 CONCLUSION AND FUTURE WORK

This paper presents a novel framework to reconstruct 3D graphics models from a set of photos and point clouds. Our system first extracts salient line segments from images, and then reconstructs the fundamental 3D frame of a target based on the line segments and plane symmetry properties. Modeling based on this frame can lessen the ambiguity situations and substantially improve the accuracy of the following primitive fitting. The proposed framework can automatically estimate primitive models and it provides friendly refinement interface. User evaluation demonstrates that it can also lessen user editing time compared to a known tool.

There are several possible future directions. An interesting extension is to construct the fundamental frame according to more salient features, such as curves. It can further reduce the ambiguity in residual fitting. Besides, applying shape-from-shading techniques on surfaces [Lin10a] or using advanced fitting methods [Li11a] for residual points can further refine the estimated models.

ACKNOWLEDGMENTS

Authors would like to thank the volunteers who participated in user evaluation. This project was partially supported by Ministry of Science and Technology, Taiwan under grant no. MOST 103-2221-E-009-143 and MOST 104-2218-E-009-008.

9 REFERENCES

[Ari13a] Arikan, M., Schwärzler, M., Flöry, S., Wimmer, M., and Maierhofer, S. O-snap: Optimization-based snapping for modeling architecture. ACM Trans. Graph., vol.32, no.1, article 6, 2013.

[Aut14a] Autodesk 123D Catch Gallery, http://www.123dapp.com/Gallery/catch.

[Aze09a] Azevedo, T.C.S., Tavares, J.M.R.S., Vaz, M.A.P. 3D object reconstruction from uncalibrated images using an off-the-shelf camera. Advances in Computational Vision and Medical Image Processing Computational Methods in Applied Sciences. vol. 13, pp 117-136, 2009.

[Aze10a] Azevedo, T.C.S., Tavares, J.M.R.S., Vaz, M.A.P. Three-dimensional reconstruction and characterization of human external shapes from two-dimensional images using volumetric methods. Computer Methods in Biomechanics and Biomedical Engineering, vol.13, no.3, pp.359-369, 2010.

[Ber14a] Bernardes, P., Magalhaes, F., Ribeiro, J., Madeira, J., Martins, M. Image-based 3D modelling in archaeology: application and evaluation, Proc. Intl. Conf. Computer Graphics, Visualization and Computer Vision (WSCG), pp. 159-166, 2014.

[Cip99a] Cipolla, R., and Robertson, D. 3D models of architectural scenes from uncalibrated images and vanishing points. Proc. Intl. Conf. Image Analysis and Processing, 1999. , pp. 824-829, 1999.

[Deb96a] Debevec, P.E., Taylor, C.J., and Malik, J. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. Proc. SIGGRAPH'96, pp.11-20, 1996.

[Glu14a] GLUI User Interface Library, http://glui.sourceforge.net/

[Iza11a] Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A. Kinectfusion: real-time 3D reconstruction and interaction using a moving depth camera. Proc. ACM symp. User interface software and technology, pp.559-568, 2011.

[laf13a] Lafarge, F., and Alliez, P. Surface reconstruction through point set structuring. Comput. Graph. Forum, vol.32, no.2, pp.225-234, 2013.

[Li11a] Li, Y., Wu, X., Chrysanthou. Y., Sharf, A., Cohen-Or. D., and Mitra, N.J. Globfit: consistently fitting primitives by discovering global relations. ACM Trans. Graph., vol.30, no.4, article 52, 2011.

[Lin10a] Lin, I.-C., Chang, W.-H., Lo, Y.-S., Peng, J.-Y, Lin, C.-Y. Image-based detail reconstruction of non-Lambertian surfaces. Computer Animation and Virtual Worlds, vol.21, no.1, pp.55-68, 2010.

[Lin15a] Lin, I.-C., Lan, Y.-C., Cheng, P.-W. SI-Cut: Structural inconsistency analysis for image foreground extraction. To appear in IEEE Trans. Visualization and Computer Graphics, 2015.

[Nan10a] Nan, L., Sharf, A., Zhang, H., Cohen-Or, D., and Chen, B. Smartboxes for interactive urban reconstruction. ACM Trans. Graphics, vol. 29, no.4, article 93, 2010.

[Ngu13a] Nguyen, H.M., Wunsche, B., Delmas, P., Lutteroth, C., van der Mark, W., Zhang, E. High-definition texture reconstruction for 3D image-based modeling, Proc. Intl. Conf. Computer Graphics, Visualization and Computer Vision (WSCG), pp.39-48, 2013.

[Ogl14a] OpenGL (Open Graphics Library), https://www.opengl.org.

[Omp14a] OpenMP (Open Multi-Processing Library), http://openmp.org/wp/

[Rot04a] Rother, C., Kolmogorov, V., and Blake, A. GrabCut: Interactive foreground extraction using iterated graph cuts, ACM Trans. Graphics, vol.23, no.3, pp. 309-314, 2004.

[Sch03a] Schindler, K., and Bauer, J. A model-based method for building reconstruction. Proc. Intl. Conf. Computer Vision workshop on Higher-Level Knowledge in 3D Modeling and Motion, pp.74-82, 2003.

[Sch07a] Schnabel, R., Wahl, R., and Klein, R. Efficient ransac for point-cloud shape detection. Comput. Graph. Forum, vol.26, no.2, pp.214-226, 2007.

[Sna06a] Snavely, N., Seitz, S.M., and Szeliski, R. Photo tourism: Exploring photo collections in 3d. ACM Trans. Graph., vol.25, no.3, pp.835-846, 2006.

[Von10a] von Gioi, R.G., Jakubowicz, J., Morel, J.M., and Randall, G. LSD: A fast line segment detector with a false detection control. IEEE Trans. Pattern Analysis and Machine Intelligence, vol.32, no.4, pp.722-732, 2010.

[Wer02a] Werner, T., and Zisserman, A. New techniques for automated architectural reconstruction from photographs. Proc. European Conf. Computer Vision-Part II, pp. 541-555, 2002.

Figure 17: The leftmost column: results of automatic stages; the other columns: final results in three different views and corresponding input photos.

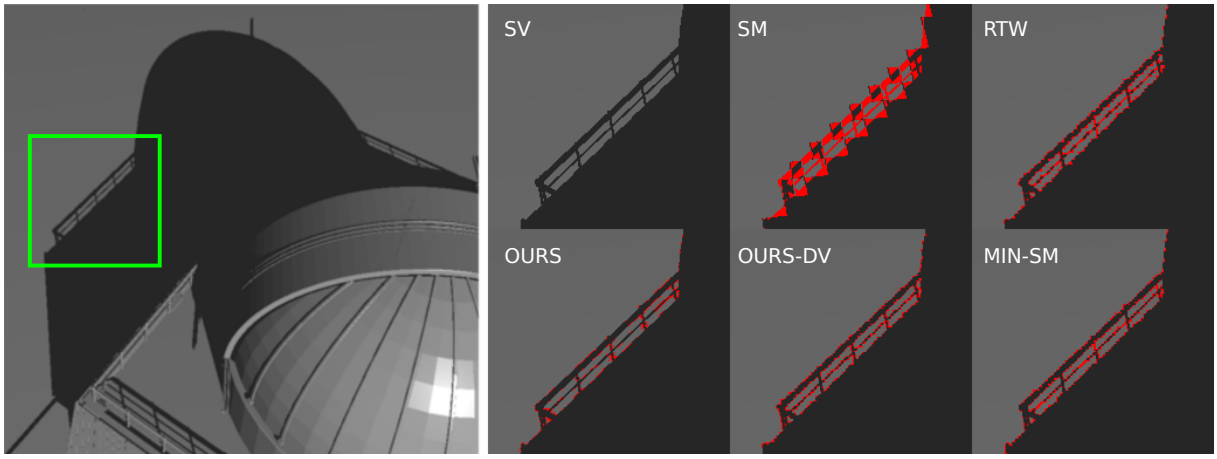
 ISBN 978-80-86943-65-7

An Improved Non-Orthogonal Texture Warping for Better Shadow Rendering

Tomáš Milet
Brno University of Technology Czech Republic
imilet@fit.vutbr.cz

Jan Navrátil
Brno University of Technology Czech Republic
inavrati@fit.vutbr.cz

Pavel Zemčík
Brno University of Technology Czech Republic
zemcik@fit.vutbr.cz



The figure shows the difference in quality. Images are zoomed on shadows cast from Observatory scene for different methods. Red pixels are wrongly evaluated. From left to right: Shadow Volumes (SV), Shadow Mapping (SM), Rectilinear Texture Warping (RTW), our solution, our solution using only desired view (DV), SM + minimal shadow frustum.

ABSTRACT

In interactive applications, shadows are traditionally rendered using the shadow mapping algorithm. The disadvantage of the algorithm is limited resolution of depth texture which may lead to aliasing artifacts on shadow edges. This paper introduces an improved depth texture warping with non-orthogonal grid that can be employed for all kinds of light sources. For instance, already known approaches for reducing aliasing artifacts are widely used in outdoor scenes with directional light sources but they are not directly applicable for point light sources. We show that the improved warping parameterization reduces the aliasing artifacts and we are able to present high quality shadows regardless of a light source or a camera position in the scene.

Keywords

shadow-mapping, alias, warping, local warping, minimal frustum, shadows

1 INTRODUCTION

Images rendered on computers are still being improved with various visual effects. Nowadays, computers can synthesize images in nearly photorealistic quality and in real-time. One of the most important visual cues, still worth improving, are shadows. The two key algorithms for shadow rendering [Wil78, Cro77] have been accelerated on GPUs.The shadow volumes approach [Cro77] suffers from the need to render large amount of data to gather necessary information for rendering shadows. On the other hand, shadow mapping approach [Wil78] is limited by the size of depth texture. In this paper, we addressed this problem with the improved depth texture parameterization that makes use of the available resources more efficient.

The resolution of the shadow map determines the number of samples that can be utilized. Recently, some

 ISBN 978-80-86943-65-7

approaches were introduced that improve sampling of the important parts of the scene. These approaches work well for outdoor scenes where we can expect that the major part of lighting comes from the directional light source [ZSXL06]. This type of light sources can be processed efficiently with the shadow mapping algorithm. On the other hand, the basic shadow mapping algorithm cannot easily address point light sources and it needs additional improvements, e.g. using cube shadow maps, or an alternative parameterization [BApS02, Ros12, JLZ13]. In this case, the improved sampling is difficult to achieve using approaches mentioned above because the algorithms would need non-trivial modification. We omit the most complex types of light sources such as area lights or volume-based emitters in this paper.

Since the point light sources can cast shadows into all directions, the regions where the need to improve scene sampling exists can be distributed throughout the depth textures. It all depends on the scene complexity and also on the mutual position of the light source and the camera. While outdoor scenes are lit by directional light source, the important parts are located in front of the camera and the importance decreases with a distance from the camera.

Various approaches have been introduced that can parameterize the shadow map and thus improve sampling on selected parts of the scene based on the scene analysis [VNHZ11, NZJP12, Ros12, JLZ13]. However, neither of these approaches is fully automatic or robust enough and they work only in few cases when the scene is lit with directional light sources or the light source is outside a camera view frustum.

Our solution computes an improved parameterization based on importance driven depth texture warping. We can identify regions in the depth texture where the sampling is not optimal and enlarge this regions in order to get higher sampling rate. We employed modern GPUs in the warping process thus these additional computing steps have no crucial impact on an overall performance.

We introduce an additional step that is performed before the traditional shadow mapping algorithm is applied. In this step, a non-orthogonal warping grid is computed and this grid is used during the shadow rendering step.

Our main contributions are:

- introduction of a novel importance function for determining sampling rate of depth texture. This function extends the set of functions introduced by Rosen et al. [Ros12],
- the non-orthogonal warping grid which leads to better control of importance-based warping without affecting the nearest regions in the texture (in the same row and/or column).

2 PREVIOUS WORK

The shadow mapping algorithm was first published in 1978 [Wil78]. Since then, many approaches addressing its aliasing issues have been published.

Stamminger and Drettakis [SD02] introduced an idea of creating depth texture after performing of perspective projection. This step emphasizes regions in front of the camera where the aliasing error are mostly observable. However, results in this approach are dependent on the mutual position of a camera and a light source. In some case, creating depth texture in post-perspective space may lead to very unpleasant results because of the perspective transformation function. Also, the overall results are influenced by object outside the camera view frustum because they introduce additional complexity to the computation.

Fernando et al. [FFBG01] introduced a hierarchical structure by subdividing shadow map into smaller shadow map pages having different resolutions due to different level of aliasing in different parts of the current camera view frustum. With camera being dynamic, this hierarchical structure needs to be updated per frame and due to limitations of graphics hardware of that time, most of the algorithm runs on CPU.

This method was further optimized by Lefohn et al. [LSK+05]. Evolution of GPU hardware allowed more of the algorithm to move on the graphic chip itself by programmable vertex and pixel shader pipeline stages.

Parallel-split shadow maps approach was introduced by Zhang et al. [ZSXL06]. The idea is to split view frustum into multiple parts according to depth, split light frustum into multiple ones and then independent shadow maps are rendered for each layer. Splitting view frustum is based on a practical splitting algorithm which averages logarithmic and uniform splitting scheme. However, this method is targeted and optimized for outdoor scenes and it would need some amount of work to adapt it for indoor scenes and namely point light sources. Also, the approach does not deal with the perspective aliasing error correctly.

Based on Zhang, Lauritzen et al. [LSL11] introduced Sample distribution shadow maps which further improves partitioning. The camera frustum is partitioned automatically based on receiver sample distribution given by depth buffer, eliminating areas with no shadow samples. This sample distribution is also used to compute tightly-bound light-space partition frusta.

The first method that addressed problem of important regions distributed in the depth texture was introduced by Rosen [Ros12]. He introduced the rectilinear warping maps that could easily control the sampling in particular parts of the depth texture. This could be controlled by importance function and the approach could be used for point light sources without complex modification. Nevertheless, the rectilinear warping schema

is not completely local.Other parts of a scene may receive unneeded resolution. This can lead to reduction in overall quality.

Similar approach was published by Jia et al. [JLZ13]. They do not limit the approach to perpendicular splitting planes; therefore, they can control the results more precisely. However, this approach needs multiple render passes of the scene to analyze the scene and decides the dividing schema. This can introduce certain issues for complex scenes.

Finally, some approaches that were focused only on point light sources have been published recently [VNHZ11, NZJP12]. Nevertheless, they discussed possibilities for improving shadow quality using Dual-Paraboloid shadow maps [BApS02]. But this technique is not sufficient due to its nonlinear transformation during generation of depth textures. This introduces additional limitation regarding model quality and especially size of polygons.

2.1 Scene Sampling and Parameterization

The shadow mapping algorithm works with two types of samples. A *view sample* is a point on a scene surface that is described by its 3D position (and other properties such as color, normal vector etc.). The view samples are generated by sampling the scene from a camera point of view. Secondly, *shadow samples* are generated by sampling the scene from a light source point of view. In both cases, the sampling is performed using an orthogonal grid with a predefined resolution.

However, multiple view samples can be projected onto one shadow sample and then aliasing can be observed in a final image as jagged edges of the shadows. This is caused by uniform rasterization of a texture produced by a graphics hardware.

Another solution is to parameterize the sampling using a *warping function* $y = f(x)$. The function enlarges important parts of a scene in order to increase shadow sampling rate. This technique increases a probability that shadows for different view samples are resolved by different shadow samples. There are two types of the warping function - *global* and *local*. The global warping function can be defined by a transformation matrix. This warping function mostly depends on a mutual position of a camera, a light source and geometry and ignores properties of view samples [SD02]. The local warping function is derived from properties of view samples and scene analysis [Ros12, JLZ13].

2.2 Rectilinear Texture Warping

Our algorithm is partially based on Rectilinear Texture Warping (RTW) approach [Ros12]. Let us make a short overview of RTW algorithm using backward analysis.

RTW approach utilizes various properties of view samples, e.g. distance to a camera, normal vector or edge detection. The warping function can be constructed using forward, backward or hybrid analysis.

The first step in the forward analysis is rendering of scene from a light source point of view. Then, the importance map is computed.One additional rendering step is necessary to compute shadows. In the backward analysis, the G-buffer with the scene's depth and color is rendered from a camera point of view. Then, the importance analysis is performed using samples projected into the light space. The hybrid analysis combines both approaches.

The backward analysis is the fastest method because it requires a scene to be rendered only two times. The first rendering pass is used to create a depth buffer from a camera. The second rendering pass creates a warped shadow map. Its complexity is linear with relation to the number of light sources.

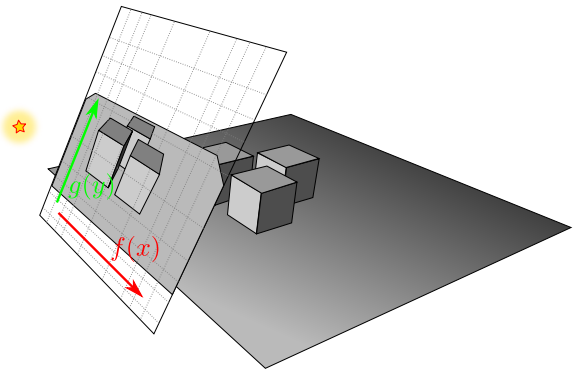


Figure 1: Two 1D warping functions enlarge parts of the scene that are important according to the importance map.

The warping function in RTW is composed of two 1D warping functions that operate in projection plane of a light source (see Figure 1). These functions are derived from an importance map. The *importance map* is constructed by projection of view samples onto the projection plane of a light source. Multiple view samples can be projected into one pixel of the importance map. In every pixel, the importance value is computed based on the view sample properties. The 1D warping functions are derived separately for column and rows according to a maximal importance value. Since the functions parameterize vertical and horizontal component of the shadow map separately they produce an orthogonal warping grid.

3 SHADOW RENDERING USING NON-ORTHOGONAL WARPING GRID

The basic idea of our algorithm is to achieve better distribution of view samples in the shadow map. Every shadow sample resolves shadow for all view samples that were projected on it. The ideal situation occurs

when one texel from the shadow map samples a surface that is projected onto one pixel in the image space. However, this is hardly achievable in most of the scenes because of the scene complexity, geometry and mutual position of the camera and the light source. Because of this fact, we can assume that the best result is observed when the number of view samples for all shadow samples is the same.

In our approach, the importance map has the same resolution as the shadow map. Every pixel in the importance map stores the number of view samples that are sampled by the given shadow sample. The importance map can be created by projection of view samples into to the light space and increase a counter by one. This step can be easily accelerated by contemporary GPUs.

The complete algorithm for computing shadow consists of the following steps:

1. Render a scene from a camera point of view to G-buffer
2. Project every view sample into the importance map
3. Compute prefix-sum for every row in the importance map
4. Construct the set of warping functions for rows according to equation 4. Use the prefix-sum from the Step 3
5. Smoothen the set of warping functions, e.g. using weighted average
6. Project every view sample onto the importance map (and increment by 1) leveraging the set of warping functions created in the previous step
7. Repeat the Steps 2-5 for all columns
8. Create shadow map using both sets of warping functions
9. Evaluate shadows in the scene using G-buffer, the set of warping functions and the warped shadow map

The first step is generation of the G-buffer. Apart from other properties, it contains positions of view samples that we need to analyze the importance for.

The most important are the steps 2-7 where we construct the set of 1D warping functions. We derive the warping functions in different manner than Rosen [Ros12]. For every row and every column, we construct one 1D warping function separately and thus we do not allocate unneeded resolution in other parts of the shadow map. The degree of freedom for warping functions is increased using this approach and we should not allow the situation illustrated on the Figure 2. The steps are described in detail in the following section.

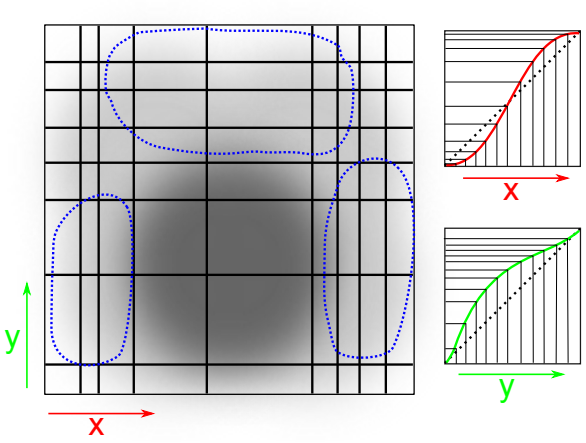


Figure 2: Importance map for RTW, Left: Combination of two 1D warping function, Right: two 1D warping function. It can be seen that blue parts are oversampled. The larger cells cover more important areas of the shadow map.

3.1 Construction of 1D Warping Functions

For one row of the importance map, let us assume a function $f(x)$ that returns the number of view samples on a normalized position x and its corresponding prefix-sum function $g(x)$:

$$n = f(x) \quad x \in \langle 0,1 \rangle \tag{1}$$

$$s = g(x) = \int_0^x f(x)dx \tag{2}$$

For evenly distributed view samples in the row, the ratio of the number of view samples on all positions before x, i.e. $g(x)$, and the total number of view samples $g(1) = N$ is equal to ratio of the position x and the row length:

$$\frac{g(x)}{g(1)} = \frac{x}{1} \tag{3}$$

Expression $g(x)/g(1) > x/1$ implies that there are more view samples than the number of samples x and thus the area needs to be enlarged to achieve uniform sampling rate. On the other hand, expression $g(x)/g(1) < x/1$ implies that there are less view samples and the area can be smaller.

Now, we can derive the warping function to be defined as an offset $o(x)$ that has to be added to the actual view sample position. The offset function is given by:

$$o(x) = \frac{g(x)}{N} - x \tag{4}$$

Let us assume that the view sample is projected onto a particular row in the shadow map. Then, a new sample position x' in the row is given by:

 ISBN 978-80-86943-65-7

$$x' = x + o(x) \tag{5}$$

Before we proceed with construction of warping functions for columns, we need to recompute the importance map. But now, we apply the newly derived set of 1D warping functions for rows. After this step, the number of view samples that have to be redistributed in a given column is nearly constant (see Figure 3). When the 1D warping functions for columns are derived, all the view samples are distributed uniformly.

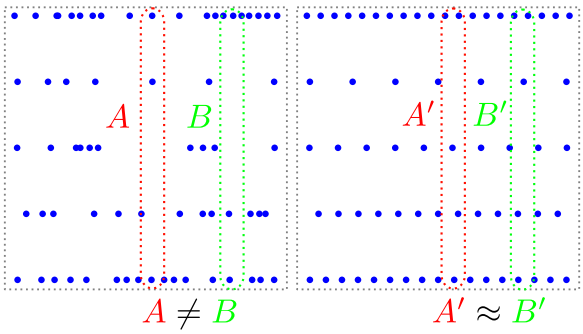


Figure 3: Left: Five rows of the importance map. Blue dots indicate view samples. Right: the importance map constructed using the set of row warping functions. Columns in the left do not contain the same number of view samples. Columns in the right contains approximately the same number of view samples.

As we mentioned in the Section 2.2, the RTW algorithm constructs two warping functions - for rows and columns respectively. We improve this approach and construct the set of warping functions for all rows and all columns. Nevertheless, we need to smoothen these functions in order to prevent them from providing too different offsets. Otherwise, the large polygons that are linearly rasterized would not be processed by the warping functions correctly. The smoothing step is included in the RTW algorithm as well. Rosen performs this step on the warping functions. However in our approach, we smoothen among all warping functions. It can be implemented, for instance, as a weighted average of the results based on the number of view samples on a row or a column respectively (see Figure 4).

The complete warping function can be expressed as:

$$\begin{aligned} warp(x,y) &= (x + o_x^{(i)}(x), y + o_y^{(j)}(y)) \\ i &= \lfloor y \cdot w \rfloor \\ j &= \lfloor (x + o_x^{(i)}(x)) \cdot w \rfloor \end{aligned} \tag{6}$$

where w is the shadow map resolution (number of pixels in one row), $o_x^{(i)}(x)$ is a warping function for i^{th} row, $o_y^{(j)}(y)$ is a warping function for j^{th} column.

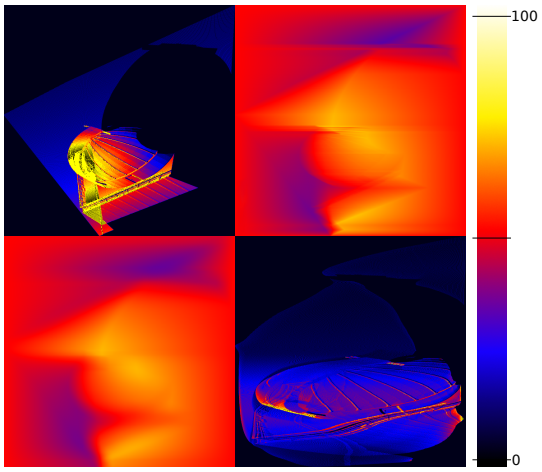


Figure 4: Top Left: Importance map, Top Right: a set of warping functions for every row of the importance map. Bottom left: smoothed warping functions, Bottom right: The importance map after application of row warping functions - importance map for columns. Yellow color in warping functions means positive offset for certain position in the row.

When we apply both sets of warping functions, the view samples projected onto the projection plane of a light source are better spread as can be seen on the Figure 5.

Once we constructed both sets of the warping functions, the shadow map can be generated (see Step 8 of the proposed algorithm). A surface point with a world-space coordinate $\mathbf{v} = (v0, v1, v2, 1)$ is projected onto the shadow map in Algorithm 1. Final shadow map can be seen in Figure 6.

Input: **v** - vertex in world space, M - light projection view matrix
Output: **p** - vertex in the shadow map clip space

1 $\mathbf{a} = M \cdot \mathbf{v}$;
2 $\mathbf{b} = ((a_1, a_2)/a_4 + 1)/2$;
3 $\mathbf{c} =$ `warp(`**b**`)`;
4 $\mathbf{d} = (\mathbf{c} \cdot 2 - 1) \cdot a_4$;
5 $\mathbf{p} = (d_1, d_2, a_3, a_4)$;

Algorithm 1: Warping function that can be used in vertex / evaluation shader. Steps 1, 2 project vertex into normalized coordinates of shadow map. Step 3 moves vertex according to warping funcions. Steps 4, 5 project vertex back into shadow map clip space.

3.2 Minimal Shadow Frustum Extension

We extended our solution with an additional improvement. We implemented an algorithm for finding a minimal shadow frustum (MSF) [SD02] and we extended it using rotating caliper. Using this technique, we project

 ISBN 978-80-86943-65-7

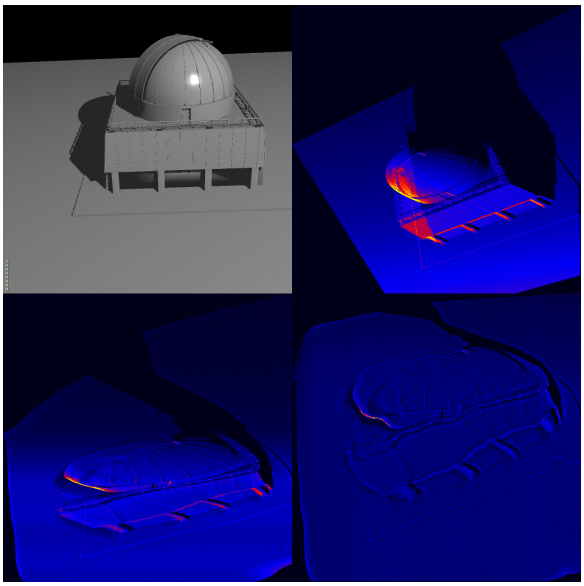


Figure 5: Top Left: Scene rendered from a camera point of view, Top Right: the importance map created from view samples. Bottom Left: reprojected view samples using only row warping functions. Bottom right: reprojected view samples using both sets for warping functions. It can be seen than importance is more spread across the importance map in the final stage. Black parts of second image are pixels with no view samples. These pixels correspond to those shadow map pixels that are useless - they resolve shadowing equation for invisible parts of the scene. In final image, these black parts almost disappear.

only parts of the scene that are visible in the camera view frustum and occluders outside the frustum that cast shadows on objects inside the frustum.

However, since the algorithm is complex, it runs on CPU and thus it may influence rendering speed. Moreover, issues caused by precision of floating point operations have to be considered during implementation.

The goal of this additional improvement is to verify whether the MSF does not provide better results with a less cost.

Rosen et al. presented a desired view (DV) function that works similarly to the MSF. However, they did not clearly show how it influences the overall quality. We support the DV in our solution as well, but it is only used as pre-process step before computing the importance map.

DV simply finds minimum and maximum view samples coordinates in the importance map. In addition, the MSF rotates the bounding box to an optimal position and adjusts near and far planes.

Rosen et al. computes DV in RTW approach from the importance map by finding first/last row and column that contains an importance value greater than zero. In our solution, DV is computed by parallel reduction over the set of view samples projected into the shadow map space. DV does not contribute to warping process, it only focuses the relevant part of shadow map. We can apply the DV function before construction of the warping functions (before the Step 2 of the Algorithm 1):

3.3 Implementations Details

We implemented the algorithm in OpenGL 4.4 using compute shaders. For creation of the importance map, we used image atomic operation *imageAtomicAdd* delivered with OpenGL.

Our solution requires additional memory as compared to the basic shadow mapping algorithm. We used deferred shading for creating the G-buffer that requires set of 2D textures. For storing the warping functions, we used two one-channel floating point 2D textures. These have the same resolution as the shadow map. Further, the algorithm requires few textures for storing temporary results - the importance map, prefix sum map and storage for not smoothed warping functions. The additional memory requirements are thus dependent on the shadow map resolution.For instance, when we use the shadow map with resolution $w = 1024$, we need to allocate additional 20 MBytes of the memory.

The memory requirements can be decreased by using e.g. another format of textures. For instance, 16bit textures for the importance map or prefix-sum map. Also, with increasing number of lights, the memory requirements increase only for storing the warping functions: $8w^2$[bytes] for one light source.

4 RESULTS AND DISCUSSION

The results were measured on a PC running Intel Core i7 4790 with 16GB of memory. We used a high-end GPU: NVidia GTX 980. Operation system was Linux Ubuntu 14.04.2.

We compared our solution with the Rectilinear Texture Warping algorithm (RTW) [Ros12], the basic shadow mapping algorithm (SM), accelerated silhouette-based shadow volumes algorithm (SV) [MKZP14] and the shadow mapping algorithm extended with the minimal shadow frustum (MIN-SM). We measured quality and speed of all approaches (see Table 1 and teaser image).

Regarding quality comparison, we selected the shadow volumes algorithm as the ground truth. It provides sample-precise shadows and moreover, it also defines a lower boundary for speed. No solution based on the shadow mapping algorithm can be slower than the silhouette-based shadow volume approach [MKZP14].

The RTW algorithm is the most similar approach to our solution. And since we suggested some improvements, the comparison to RTW is very important. We implemented RTW algorithm with backward analysis used

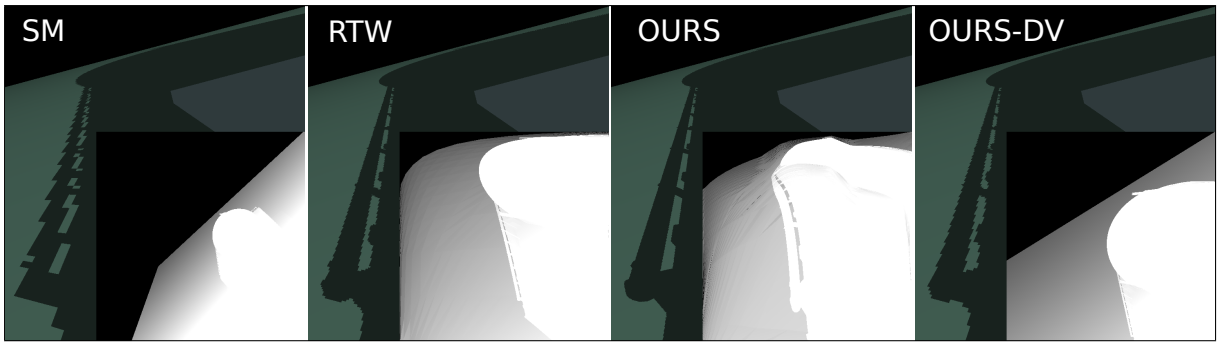


Figure 6: Images show shadow maps (grey squared images) for Observatory scene. From left to right: shadow mapping (SM), Rectilinear Texture Warping (RTW), our solution, our solution using only DV.

Method	time per frame
SM	1.596
MIN-SM	1.7
SV	8.750
RTW	3.296
Ours	**4.708**
Ours-DV	**2.521**

Table 1: Performance comparison of implemented methods. Times are in miliseconds. Measured for Observatory scene on 1024×1024 resolution with 512×512 resolution for the shadow map.

for creation of the importance map. The timings for the crucial steps of both approaches can be seen in Table 2. We used both the distance to eye importance function and the desired view function in all reference images (see teaser image).

Also, comparison with the shadow mapping algorithm extended with the minimal shadow frustum (MSF) shows some interesting results. The main reason for including this method is that we wanted to know whether the MSF is not sufficient enough to render images of the similar quality. Rosen did not described this extension and did not show any results.

We measured our algorithm on three scenes (see Figure 7). We selected various types of scenes (outdoor as well as indoor) in order to show that our solution can be adapted to different environment and types of light sources. Times for all scenes are shown in Table 3.

In Figure 7, you can see differences from a reference solution (shadow volumes algorithm). Red pixels are incorrectly computed.

As it can be seen in Table 3, our method is slightly slower than RTW but it produces better visual results (see teaser image and Figure 7). Measurements show that computing MSF is not expensive and it may be suitable in some situations.Also, it did not provide the best quality. Computing desired view (DV) function in our method is the third fastest method but visual results are worse than using MSF. Results also show that DV function is major part of decreasing alias error, but in some situation it is not sufficient.

scene	Conf. room		Sponza		Observatory	
	ours	rtw	**ours**	rtw	**ours**	rtw
desired view	**13.9**	89.1	**15.4**	82.5	**18.4**	86.8
imp. map	**68.4**		**61.0**		**69.0**	
shadow map	**14.4**	7.3	**19.9**	14.1	**8.2**	9.3
final pass	**3.3**	3.5	**3.7**	3.2	**4.4**	3.8

Table 2: Overhead of steps in our algorithm for different scenes. Values are in percent.

Scene	Conf. room	Sponza	Observatory
triangles	126665	261978	52583
gbuffer	2.16	2.229	1.84
SV	9.64	18.41	14.96
SM	0.21	0.40	0.16
RTW	3.14	3.47	3.02
Ours	**3.63**	**3.84**	**3.23**

Table 3: Performance comparison of implemented methods for different scenes. Times are in miliseconds.

Minimal Shadow Frustum

The experimental results show that performance of DV and MSF depends on current hardware setup. MSF performs better than DV when running on fast CPU and slow GPU. When we compared the impact of both approaches on quality comparing the texture warping techniques, the results are following. The MSF or DV perform better when a small part of a scene is rendered. However, in real world scenes the camera renders a bigger part of a scene and in these cases the warping techniques perform better (see Figure 7 and 6). The MSF or DV do not generate the view frustum small enough and thus artifacts on shadow edges are more apparent.

4.1 Limitations

Our implementation as well as the RTW algorithm have to deal with linear rasterization unit. In Figure 5 (bottom right), we can see the result of our warping process. It can be seen that the warping functions distorted the

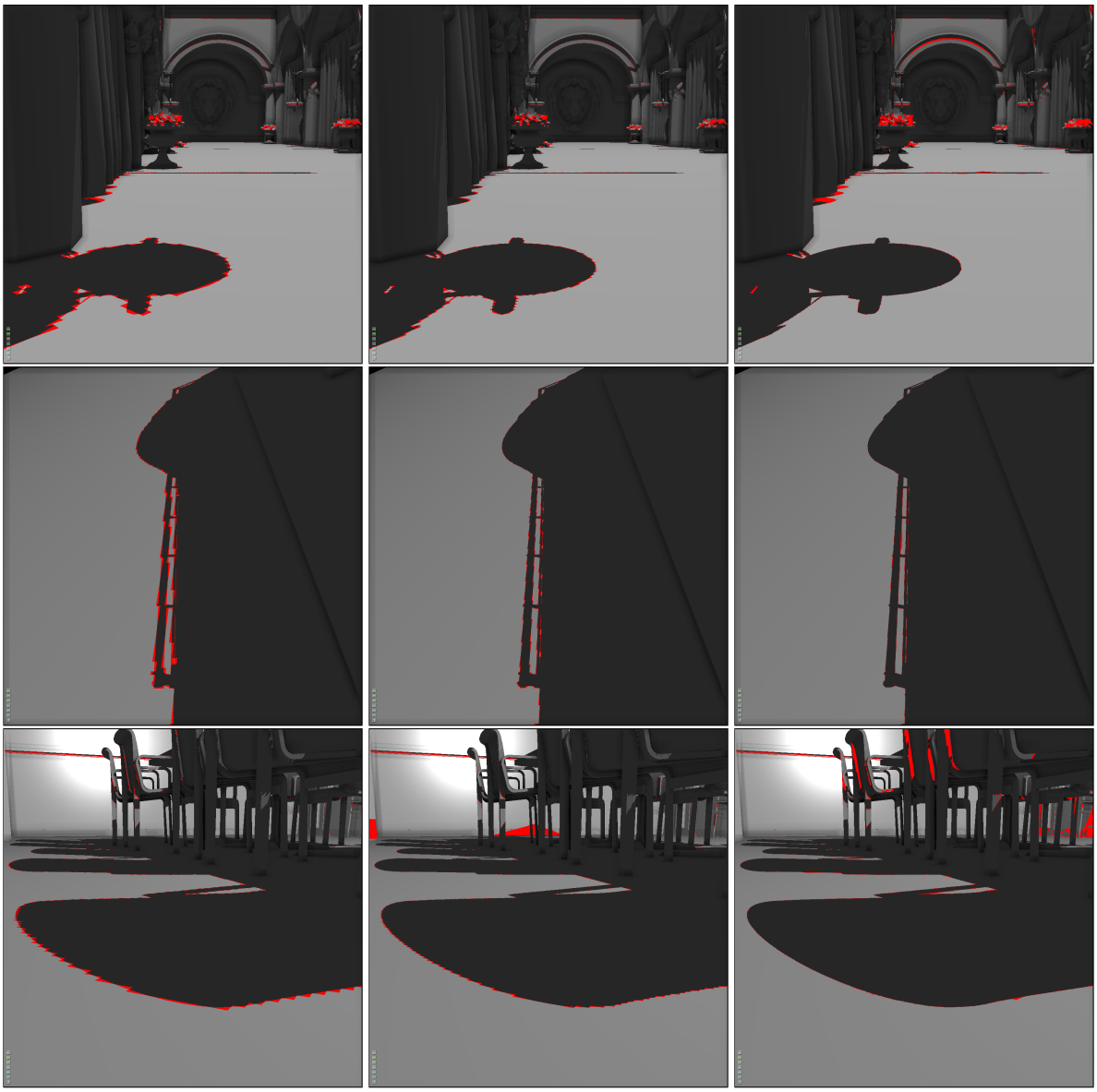


Figure 7: Images show difference between shadow mapping techniques and shadow volumes. Images in the first column show basic shadow mapping. The second column shows RTW method and third column shows our new warping method. First scene is Sponza, second scene is Observatory and last scene is Conference room. Times are shown in Table 3.

space. Nowadays, the rasterization pipeline can handle only the triangle vertices. If the warping function changes rapidly between two vertices, we can see some errors (see Figure 7 top, right for missing shadows under curtains). We used a few techniques in our solution to deal with these errors.

Firstly, we utilized the adaptive tessellation provided by OpenGL. The similar improvement was suggested by Rosen et al. Further, we modified the size of smoothing window when averaging the warping functions. The wider the window is the less different are the warping functions. In extreme case, our solution converts to the RTW algorithm. Another solution is to use weights during smoothing step. It can influence sizes of offset values.

5 CONCLUSION AND FUTURE WORK

This paper presents an extension of the Rectilinear Texture Warping algorithm achieved through the improved non-orthogonal warping grid constructed using the set of 1D warping functions. The novel importance warping function results in less artifacts at the shadow edges.

The improvement has been evaluated on various testing scenes. We showed that the method is fast and provides better results than the RTW algorithm. Also, we dis-

 ISBN 978-80-86943-65-7

cussed various improvements and extensions that can be used together with our solution.

Standard methods for alias reduction globally change sampling rate using partitioning of a scene where directional light sources are commonly used. Our method change sampling rate locally and thus it can be used with other kinds of light sources using DPSM or cube maps.

The future work includes adaptation the algorithm for other visual effects, e.g. mirrors, refraction etc. We will focus on more experiments with the shadow mapping algorithm for point light source, i.e. the Cube Shadow Maps.

6 ACKNOWLEDGMENTS

This work was supported by the Ministry of Education, Youth and Sport of the Czech Republic under the research program TE01020415 (V3C - Visual Computing Competence Center). Additional data and algorithms were provided by Cadwork.

7 REFERENCES

[BApS02] Stefan Brabec, Thomas Annen, and Hans peter Seidel. Shadow mapping for hemispherical and omnidirectional light sources. In *In Proc. of Computer Graphics International*, pages 397–408, 2002.

[Cro77] Franklin C. Crow. Shadow algorithms for computer graphics. In *Proceedings of the 4th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '77, pages 242–248, New York, NY, USA, 1977. ACM.

[FFBG01] Randima Fernando, Sebastian Fernandez, Kavita Bala, and Donald P. Greenberg. Adaptive shadow maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 387–390, New York, NY, USA, 2001. ACM.

[JLZ13] Nixiang Jia, Dening Luo, and Yanci Zhang. Distorted shadow mapping. In *Proceedings of the 19th ACM Symposium on Virtual Reality Software and Technology*, VRST '13, pages 209–214, New York, NY, USA, 2013. ACM.

[LSK+05] Aaron Lefohn, Shubhabrata Sengupta, Joe M. Kniss, Robert Strzodka, and John D. Owens. Dynamic adaptive shadow maps on graphics hardware. In *ACM SIGGRAPH 2005 Conference Abstracts and Applications*, August 2005.

[LSL11] Andrew Lauritzen, Marco Salvi, and Aaron Lefohn. Sample distribution shadow maps. In *Symposium on Interactive 3D Graphics and Games*, I3D '11, pages 97–102, New York, NY, USA, 2011. ACM.

[MKZP14] Tomáš Milet, Jozef Kobrtek, Pavel Zemčík, and Jan Pečiva. Fast and robust tessellation-based silhouette shadows. In *WSCG 2014 - Poster papers proceedings*, pages 33–38. University of West Bohemia in Pilsen, 2014.

[NZJP12] Jan Navrátil, Pavel Zemčík, Roman Juránek, and Jan Pečiva. A skewed paraboloid cut for better shadow rendering. In *Proceedings of Computer Graphics International 2012*, page 4. Springer Verlag, 2012.

[Ros12] Paul Rosen. Rectilinear texture warping for fast adaptive shadow mapping. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '12, pages 151–158, New York, NY, USA, 2012. ACM.

[SD02] Marc Stamminger and George Drettakis. Perspective shadow maps. *ACM Trans. Graph.*, 21(3):557–562, July 2002.

[VNHZ11] Juraj Vanek, Jan Navrátil, Adam Herout, and Pavel Zemčík. High-quality shadows with improved paraboloid mapping. In *Advances in Visual Computing*, Lecture Notes in Computer Science 6938, pages 421–430. Faculty of Information Technology BUT, 2011.

[Wil78] Lance Williams. Casting curved shadows on curved surfaces. *SIGGRAPH Comput. Graph.*, 12(3):270–274, August 1978.

[ZSXL06] Fan Zhang, Hanqiu Sun, Leilei Xu, and Lee Kit Lun. Parallel-split shadow maps for large-scale virtual environments. In *Proceedings of the 2006 ACM International Conference on Virtual Reality Continuum and Its Applications*, VRCIA '06, pages 311–318, New York, NY, USA, 2006. ACM.

Evaluation of Space Partitioning Data Structures for Nonlinear Mapping

Myasnikov E.V.

Samara State Aerospace University,
Image Processing Systems Institute of the Russian Academy of Sciences
Samara, Russia

mevg@geosamara.ru

ABSTRACT

Nonlinear mapping (Sammon mapping) is a nonlinear dimensionality reduction technique operating on the data structure preserving principle. Several possible space partitioning data structures (vp-trees, kd-trees and cluster trees) are applied in the paper to improve the efficiency of the nonlinear mapping algorithm. At the first step specified structures partition the input multidimensional space, at the second step space partitioning structure is used to build up the list of reference nodes used to approximate calculations. The further steps perform initialization and iterative refinement of the low-dimensional coordinates of objects in the output space using created lists of reference nodes. Analyzed space partitioning data structures are evaluated in terms of the data mapping error and runtime. The experiments are carried out on the well-known datasets.

Keywords

Dimensionality reduction, nonlinear mapping, Sammon mapping, multidimensional scaling, MDS, space partitioning, space decomposition, vp-tree, kd-tree, cluster tree

INTRODUCTION

Nonlinear mapping algorithm [Sam69] also known as a Sammon mapping is one of the most well-known explorative data analysis techniques. It belongs to a more broad class of nonlinear dimensionality reduction techniques operating on the data structure preserving principle.

Nonlinear mapping is widely used in scientific research, and in many areas of production activities.

In signal and image analysis nonlinear mapping has been applied in creation of navigation systems for image and multimedia collections. For example, for digital image collections feature information based on visual characteristics of images is extracted at first. Then the nonlinear mapping algorithm is used to map the images from multidimensional feature space to the navigation space (2 or 3 dimensional). Similar images are placed close to each other in this navigation space and a user can browse the collection due to visual characteristics of images.

Another application of nonlinear mapping is automated segmentation and thematic classification of multispectral satellite image. In this case separate pixels are first clusterized due to its feature information and then the clusters are mapped into two-dimensional space using nonlinear mapping algorithm. In this space similar clusters are placed close to each other that allows user to merge clusters belonging to same segments. Such merged clusters can be later semantically labeled to perform thematic classification of an image.

Nonlinear mapping performs projection from some input multidimensional space to output low-dimensional space for a given set of objects (are often referred to as data points) $O = \{o_1, o_2, \ldots, o_N\}$. We consider the preservation of the data structure as the preservation of the pairwise distances between the objects.

That is the distances $d^*(o_i, o_j)$ between pairs of objects o_i and o_j in the output low-dimensional space should approximate corresponding distances $d(o_i, o_j)$ in the input multidimensional space. It is obvious that such a projection cannot preserve distances exactly and the following data

representation error allows to estimate the quality of the mapping:

$$\varepsilon = \frac{1}{\sum_{\substack{o_i,o_j \in O \\ i<j}} d(o_i,o_j)} \cdot \sum_{\substack{o_i,o_j \in O \\ i<j}} \frac{\left(d(o_i,o_j) - d^*(o_i,o_j)\right)^2}{d(o_i,o_j)} \quad (1)$$

Applying gradient descent approach to minimize the data representation error we obtain the following iterative equation for coordinates $\mathbf{y}_i$ of object o_i in the output low-dimensional space:

$$\mathbf{y}_i(t+1) = \mathbf{y}_i(t) + m \cdot \sum_{o_j \in O} \varsigma(o_i,o_j) \quad (2)$$

where

$$m = \frac{2 \cdot \alpha}{\sum_{\substack{o_i,o_j \in O \\ i<j}} d(o_i,o_j)},$$

$$\varsigma(o_i,o_j) = \frac{d(o_i,o_j) - d^*(o_i,o_j)}{d(o_i,o_j) \cdot d^*(o_i,o_j)} \cdot \left(\mathbf{y}_i - \mathbf{y}_j\right) \quad (3)$$

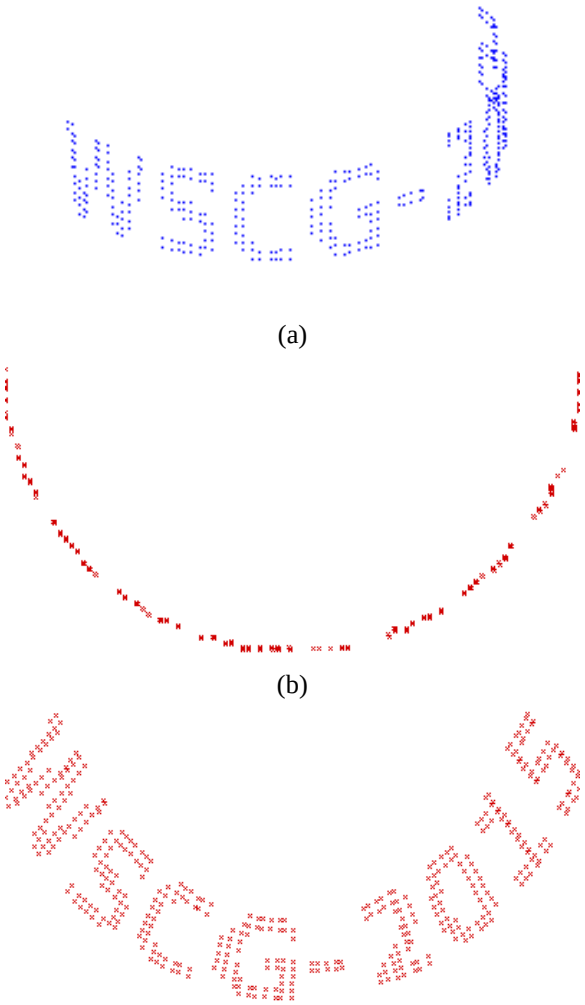
Here t is the number of iteration, α is some coefficient that affects the convergence of the algorithm. The notations in the above equations are different from the ones usually used in literature to simplify the further description.

An example of a nonlinear mapping for a synthetic dataset is shown on fig. 1. The dataset consists of a set of points obtained from the text depicted on a cylinder in 3D space (fig. 1.a). Fig 1.b shows the results obtained using the principal component analysis (PCA). Dataset is mapped on the first two principal components. The results of the described above nonlinear mapping algorithm is shown on the fig. 1.c. As it can be seen the local data structure appears.

Fig. 2 shows a mapping of a set of multi-spectral values (4 spectral bands) of pixels in 3x3 neighborhoods (total 36 attributes) in a LANDSAT satellite image [LAN]. Different classes are shown in different color.

To obtain a solution one should initialize the coordinates $\mathbf{y}_i(0)$ and iteratively refine coordinates of all the objects in accordance to (2) until the coordinates become stable.

Unfortunately, this simple iterative procedure given above has a significant drawback. If the set O contains N objects then the computational complexity is $O(N^2)$ per iteration (the computational complexity for the data representation error is roughly the same).



(a)

(b)

(c)

Figure 1. Mapping for synthetic dataset:

(a) synthetic 3D data; (b) mapping on the first two principal components; (c) nonlinear mapping (data representation error is ε = 0.0046)

The algorithm becomes time consuming for relatively small sets of objects containing hundreds and thousands of objects. The problem becomes more significant for sets containing some tens thousands of objects as the memory needed to store precomputed distances between the objects in the input multidimensional space is *N*(N-1)/2* (having account that the distance matrix is symmetrical). For example, for a set of 20 000 objects we need about 1,5 Gb of operational memory to store the distances as a 8 byte floating point values. Recomputing the distances makes the optimization process even more time consuming and dependent on the dimensionality of the input space. As an example, the time needed to compute the error in accordance to (1) for a MNIST dataset containing 60 000 objects used in this paper for an experimental study takes more than one hour.

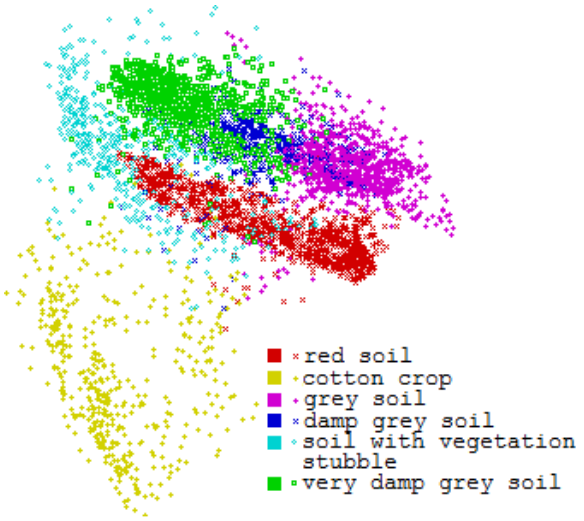


Figure 2. Nonlinear mapping for the Landsat Satellite dataset (data representation error is ε = 0.0177)

One of the most effective and promising approaches to reduce the computational complexity is to perform hierarchical partitioning of space. In the case of the considered problem such partitioning can be performed using different space partitioning structures. This study is devoted to the comparison of several space partitioning structures in terms of the quality of mapping, as well as in terms of operating time.

Related works

To address the problem of a computational complexity of the nonlinear mapping the number of techniques has been proposed. For this purpose the triangulation [Lee77], and linear transformation [Pek99] are used.

Later the performance of the nonlinear mapping has been improved by extending data representation error using left [Sun11a] and right [Sun11b] Bregman divergence.

Taking into account that nonlinear mapping belongs to the multidimensional scaling techniques it is worth mentioning the methods based on the stochastic optimization [Cha96].

The idea of a hierarchical space partitioning to reduce the computational complexity has originated from physics where it is widely used in modelling systems consisting of a huge number of objects (n-body simulations).

The partitioning of the space has been implemented in graph drawing, for example, in [Fru91] (regular decomposition) and [Qui01] (hierarchical decomposition) to approximate the forces acting on the vertices of the graph.

In dimensionality reduction the hierarchical partitioning of the input multidimensional space similar to Barnes-Hut [Bar86] algorithm has been implemented in [Mya12] to improve nonlinear mapping. The hierarchical partitioning of the output low-dimensional space using Barnes-Hut algorithm has been implemented in [Van13, Yan13] to accelerate stochastic neighbor embedding algorithm (t-SNE). In [Vla14] another widely used fast multipole method [Gre87] has been applied to speed up elastic embedding algorithm.

Approximate computations

The main idea of speeding up computations using the space partitioning techniques is to divide the whole set of objects O to some subsets $s_1, s_2, \ldots$ so that all objects $o_i \in s_k$ from some subset s_k should possess similar characteristics. That is all objects from some subset s_k are situated close to each other in the given space. In this case objects in a subset s_k can be analyzed not individually, but as a single object under some circumstances. Assume that the subset s_k is situated far from the object o_i. Then the exact summation

$$\sum_{o_j \in s_k} \frac{d(o_i, o_j) - d^*(o_i, o_j)}{d(o_i, o_j) \cdot d^*(o_i, o_j)} \cdot \left(\mathbf{y}_i - \mathbf{y}_j\right)\Big)$$

can be approximated as follows

$$|s_k| \frac{d(o_i, s_k) - d^*(o_i, s_k)}{d(o_i, s_k) \cdot d^*(o_i, s_k)} \cdot \left(\mathbf{y}_i - \mathbf{y}_{s_k}\right)$$

where $|s_k|$ is the number of objects in the considered subset, $d(o_i, s_k)$ is the distance from the object o_i to the center of the subset s_k in the input space, $d^*(o_i, s_k)$ is the distance from the object o_i to the center of the subset in the output space, $\mathbf{y}_{s_k}$ is the coordinates of the center of the group in the output space.

Now if all the objects of the original set O are divided into subsets $s_k \in S$, then (2) takes the form

$$\mathbf{y}_i(t+1) = \mathbf{y}_i(t) + m \cdot \sum_{s_k \in S} \tilde{\xi}(o_i, s_k), \qquad (4)$$

$$\tilde{\xi}(o_i, s_k) = |s_k| \cdot \frac{d(o_i, s_k) - d^*(o_i, s_k)}{d(o_i, s_k) \cdot d^*(o_i, s_k)} \cdot \left(\mathbf{y}_i - \mathbf{y}_{s_k}\right), \quad (5)$$

It is obvious that the equation (4) allows to approximate (2) with a certain accuracy that depends on how well objects are divided into subsets.

For this reason, there is a question about how to perform such a decomposition.

Description of the Methods

The base scheme for the method described below is taken from [Mya12] with some modifications. In general the considered method consists of the four following steps:

1. Partitioning of the input space
2. Construction of reference nodes lists
3. Initialization in the output space
4. Iterative optimization

The above stages are discussed below in more detail.

Partitioning of the input space

The first step of the original method assumed the hierarchical clustering to partition the input space. This lead to the $O(N^2)$ complexity in the case of effective agglomerative clustering. In the case of divisive method based on the neural network (WTA) that was implemented in [Mya12] the runtime of hierarchical clustering is highly dependent on the parameters of the algorithm.

At the same time there are a number of space partitioning trees that can be built in less time and that used widely in multimedia databases, geographic information systems, information retrieval, computer graphics and so on. The review of such data structures is beyond the scope of this paper. A comprehensive work on space partitioning trees can be found, for example, in [Sam06].

In this work several binary space partitioning trees were used and compared for input space partitioning: kd-tree [Ben75], metric tree (vp-tree) [Uhl91, Yia93], and binary tree based on the simplification of minmax distance clustering approach [Tou74] (further "cluster tree").

Such choice is motivated to study structures that are different in their properties. Kd-tree is one of the old and well-known space partitioning structures based on the recursive splitting a space with hyperplanes orthogonal to the coordinate axes. Vp-tree is another well-known space partitioning structure based on the hypersherical partitioning of a space. Cluster tree is the distance based structure that partitions a space with hyperplanes orthogonal to a pair of chosen distant points.

All the mentioned structures can be built in $O(N \log N)$ operations (if the tree is balanced) that is more suitable for the considered task. As the construction algorithms for the first two structures are well known only the algorithm for latter structure is given below by a pseudo code.

```
Function CreateTree( SetOfObjects O ) returns Node
begin
    Create new node S
    if Number of objects in O is less than threshold then
    begin
        S.Children = O
        return S
    end
    Find mean vector value in O
    Find o1 object farthest from mean
    Find o2 object farthest from o1
    Create new SetOfObjects s1
    Create new SetOfObjects s2
    for each object o in O begin
        if ( d(o1, o) < d( o2, o ) )
            add o to s1
        else
            add o to s2
    end
    Node n1 = CreateTree( s1 )
    Node n2 = CreateTree( s2 )
    add n1 to S.Children
    add n2 to S.Children
    return S
end
```

Let us assume that using some partitioning method (e.g. using function given above) we get the tree-like data structure (fig. 3), containing the objects of the initial set O as leaves (terminal nodes).

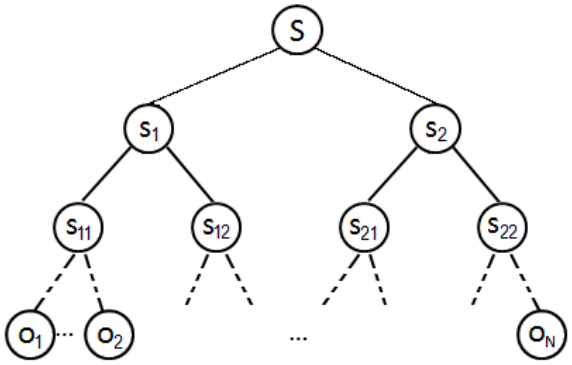


Figure 3. The structure of a binary space partitioning tree

The tree created at the first step of an algorithm can be used in optimization process immediately in the following way. We iteratively update low-dimensional coordinates for each object $o_i \in O$ in accordance to (4), using top-level nodes of the tree if such nodes are far enough from the object o_i, and low-level nodes of the tree or the objects of the initial set O in the other case.

However, the constructed tree is not used directly in the optimization process in this paper. Instead of it the special data structure is used in optimization, which is described in the next subsection. The details of the optimization process are given in "Iterative optimization" subsections.

Construction of reference nodes lists

It is worth noting that as the partitioning is performed in the input multidimensional space then the

partitioning tree is constant in the optimization process.

That is why in [Mya12] it was proposed to calculate and to store the set of nodes and the objects which are involved in the optimization process (called the list of reference nodes) for each object $o_i \in O$ of the initial set. The use of the lists of reference nodes requires additional memory but allows to avoid recalculating the partitioning criterion during the optimization process and allows to store precomputed distances to reference nodes that makes the optimization process independent on the input dimensionality.

The straightforward algorithm for creation of reference nodes is recursive and described below by pseudo code.

```
Procedure CreateRefList( Object o, Node s, RefNodeList rl)
begin
    if o far from s then
        add s to rl
    else if s contains objects then
        for each object a in s
            add a to rl
    else    /* s contains subtrees */
        for each node d contained in s
            CreateRefList( o, d, rl )
end
```

To avoid recursive calls the iterative implementation using special pointers in tree nodes was used in the experiments.

The described above algorithm is slightly different compared to [Mya12] by removing the notion of incomplete node. The use of incomplete nodes allow to slightly accelerate calculations by aggregating some portion of objects in decomposed nodes of the tree but it requires extra memory to store information about aggregated objects.

The first condition "*o* is far from *s*" defines the decomposition criterion of the given node *s* with respect to the object *o*. Different decomposition criterions were described in physics literature (e.g. [Sal94]). And here we adapt (as in [Mya12]) simple decomposition criterion based on the radius *R* of the given node *s*. That is we decompose the node of the space partitioning tree under the following condition:

$$d(s,o)/R(s) \le T$$

where T is a predefined threshold parameter. Otherwise we add the node *s* of the tree to the list of the reference nodes.

Fig. 4 illustrates this process. The node s_1 will be divided into two nodes if $d(s_1,o)/R(s_1) \le T$. The node s_2 will be considered as a single node if $d(s_2,o)/R(s_2) > T$.

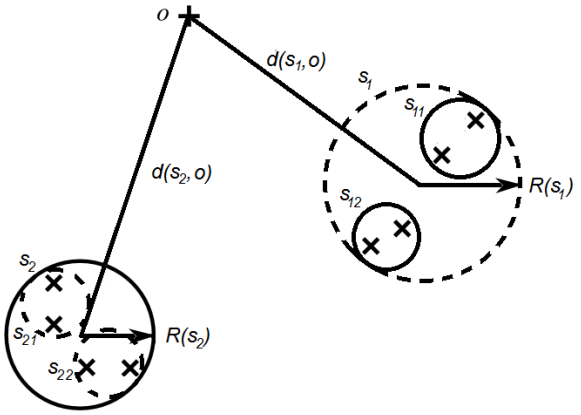


Figure 4. Illustration to the construction of reference nodes list

It is worth noting that the number of the nodes in the list of the reference nodes is dependent on the distribution of the data. Assuming that the list of reference nodes contains on average L nodes the memory needed to store the lists is O[LN].

Initialization in the output space

The third step of the original method performs an initialization of the low-dimensional coordinates of the objects. Only two first principal components are computed to initialize objects in the output space and then input multidimensional coordinates are projected to the plane formed by these components. This approach allows to reduce the computation time compared to random initialization as the next step of iterative optimization process starts with the better initial conditions.

Although principal components finding by the covariance matrix is dependent on the dimensionality of the input space D and can be estimated as O(D^2) per iteration the overall process can be time consuming as covariance matrix finding is O(ND^2) and depends not only on the dimensionality but on the number N of objects also. To reduce the computational time we use only a small subset of randomly selected objects of the initial set O making the overall complexity independent on the number of objects. There are other solutions, e.g. use of neural network approach based on generalized Hebbian (Sanger) learning rule [San89] used in [Mya12].

Iterative optimization

After initializing low-dimensional coordinates of objects the optimization procedure is performed to find sub optimal coordinates of objects in output low-dimensional space. It consists of iterative refinement of output coordinates of all objects in accordance with

$$\mathbf{y}_i(t+1) = \mathbf{y}_i(t) + m \cdot \sum_{s_{ik} \in S_i} \tilde{\boldsymbol{\varsigma}}(o_i, s_{ik})$$

 ISBN 978-80-86943-65-7

where set S_i is the list of reference nodes for object o_i, and subsets s_{ik} are reference nodes for object o_i.

The computational complexity of the optimization stage of the method can be estimated as O(LN) on average where L is the average length of lists of reference nodes.

In practice it may be reasonable to control the data mapping error along the process of optimization as it can indicate the divergence of the process or it may be used in a stop criterion. The computation complexity of the data mapping error (1) is roughly the same as the complexity of one step of the base version of the optimization process (O[N^2]). Using the lists of reference nodes data mapping error can be estimated in O(LN) on the average:

$$\tilde{\varepsilon} = \frac{1}{\sum_{\substack{o_i \in O \\ s_{ik} \in S_i}} d(o_i, s_{ik})} \cdot \sum_{\substack{o_i \in O \\ s_{ik} \in S_i}} \frac{\left(d(o_i, s_{ik}) - d^*(o_i, s_{ik})\right)^2}{d(o_i, s_{ik})}$$

It is worth noting that the above equation may give greatly underestimated values of the data mapping error especially in case when approximation based on the input multidimensional information becomes too coarse. For example, this can take place due to poor configuration of objects in the low-dimensional space.

In the present work this equation was used to control the optimization process. When the estimated error value was increasing then the α coefficient was decreasing until the process became convergent. Also this equation was used to stop the optimization process when the relative decrease in estimated error for a given number of iterations did not exceed the predefined value.

Experimental study

Two well-known datasets were used in the presented study. The first one is the MNIST database of handwritten digits [MNI]. The second one is the Corel Image Features Data Set [COR].

The first database contains digital grayscale images of handwritten digits. The database is divided in two sets: a training set containing 60 000 instances, and test set containing 10 000 instances. Images of the training set with size 28x28 pixels are treated as vectors in 784-dimensional space in the experiments.

The second dataset contains features, calculated from the digital images of the Corel image collection (http://corel.digitalriver.com/). The Corel Image Features Data Set contains 68 040 instances.

The following features have been used in the experiments:

- color histograms [Swa91] constructed in the HSV color space. Color space was divided into 8 ranges of H and 4 ranges of S. The dimensionality of the feature space is 32.

- color moments [Sti95]. Three features were calculated for each color component: mean, standard deviation, and skewness. The dimensionality of the feature space is 9.

- texture features based on co-occurrence matrices [Har73]. Four co-occurrence features (second angular moment, contrast, inverse difference moment, and entropy) were computed in four directions (horizontal, vertical, and two diagonal). The dimensionality of the feature space is 16.

To evaluate the effectiveness of the methods a number of characteristics has been measured and calculated:

- building time of the binary space partitioning tree,

- building time of the list of reference nodes,

- length of the list of reference nodes,

- initialization time of the low-dimensional coordinates,

- per iteration execution time of the optimization procedure,

- multidimensional data representation error.

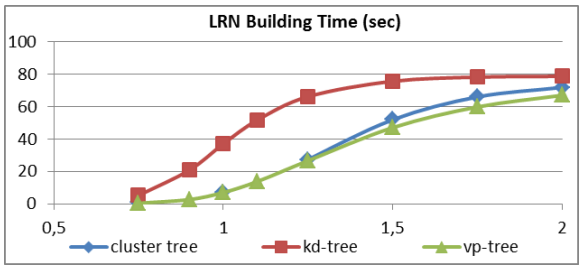
All the described methods were implemented in C++. The studies with the MNIST dataset have been carried out on PC based on Intel Core i5-3470 CPU 3.2 GHz. The studies with the COREL dataset have been carried out on laptop based on Intel Core i3 M370 CPU 2.4 GHz.

The work of the methods stopped when the relative decrease in estimated error for ten iterations did not exceed 0.01. In all cases, the dimension of the target space has been set equal to two (two-dimensional data mapping).

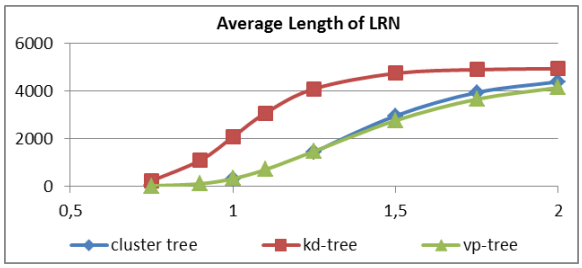
Some results are shown in fig. 5-7.

Fig. 5 shows the dependence of the qualitative and temporal characteristics on the threshold parameter T at which the algorithm moves to the child nodes of the corresponding partitioning structure (metric (vp-) tree, kd-tree or cluster tree).

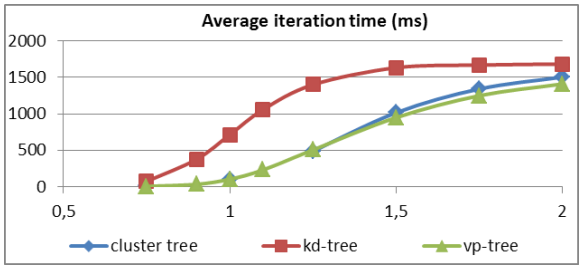
As it can be seen from these results, the small values of T (especially T<1) leads to the expected deterioration of the mapping quality due to a coarser approximation, which is reflected in the higher values of the multidimensional data representation error ε (see fig. 5d). The time it takes to perform a single iteration, increases with increasing T (see fig. 5c), due to the large number of the processed reference nodes of the corresponding hierarchical structure (see fig. 5b).



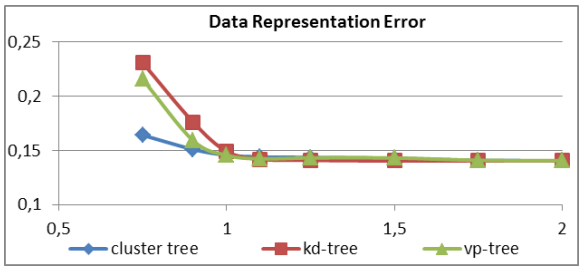
(a)



(b)



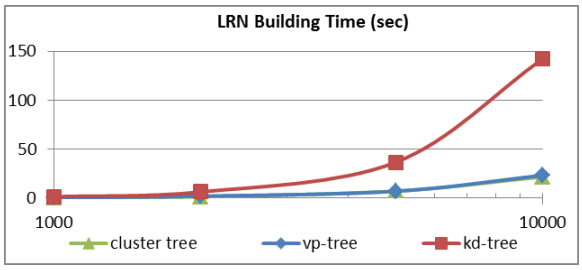
(c)



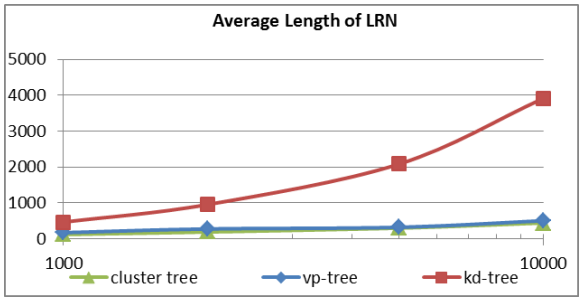
(d)

Figure 5. Dependencies on the threshold parameter *T* (MNIST dataset): (a) average building time of the list of reference nodes (LRN); (b) average length of the list of reference nodes; (c) average time per one iteration of the optimization process; (d) multidimensional data representation error ε

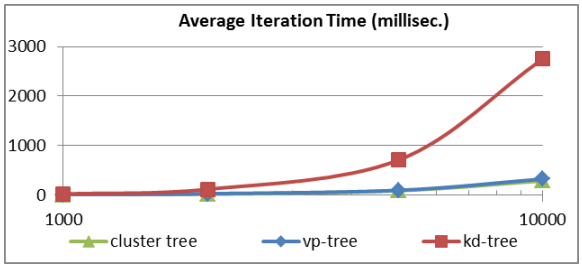
Dependencies of quality and temporal characteristics on the number of objects is shown in the fig. 6 and 7. The quality of mapping, measured by the multidimensional data representation error ε (see fig. 6d, 7d) is weakly dependent on the type of space partitioning structure. At the same time the average length of lists of reference nodes is significantly larger when we use kd-tree (see fig. 6b, 7b). This confirms that kd-trees are poorly suited for multidimensional data processing.



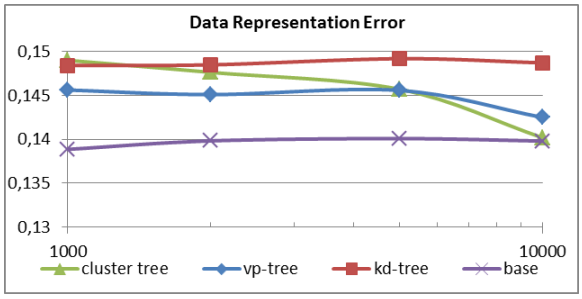
(a)



(b)



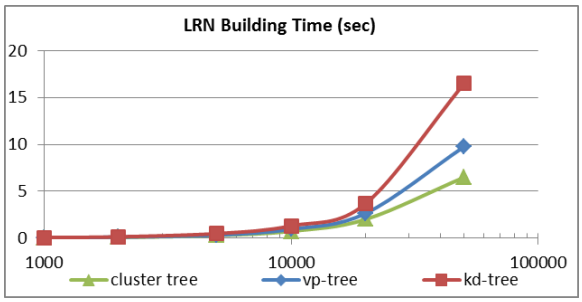
(c)



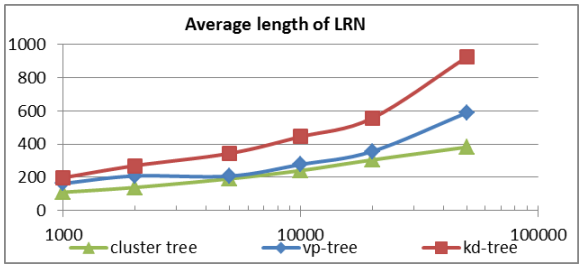
(d)

Figure 6. Dependencies on the sample size (MNIST dataset): (a) average building time of the list of reference nodes (LRN); (b) average length of the list of reference nodes; (c) average time per one iteration of the optimization process; (d) multidimensional data representation error ε

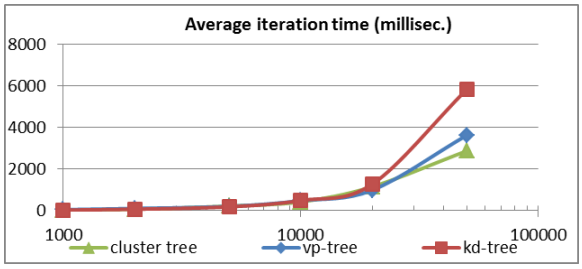
Some results obtained for the base nonlinear mapping algorithm is shown on fig. 8 (logarithmic scale). Timings for the case of precomputed distances are not shown for large sample sizes due to memory limitations. The multidimensional data representation error is shown on fig. 7 for comparison.



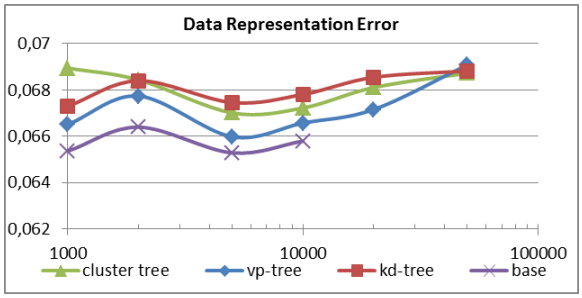
(a)



(b)



(c)



(d)

Figure 7. Dependencies on the sample size (COREL dataset): (a) average building time of the list of reference nodes (LRN); (b) average length of the list of reference nodes; (c) average time per one iteration of the optimization process; (d) multidimensional data representation error ε

As we can see the error values for the base method is only slightly better than error values obtained using the studied methods.

Note that the experiments performed on other datasets described above, show similar results.

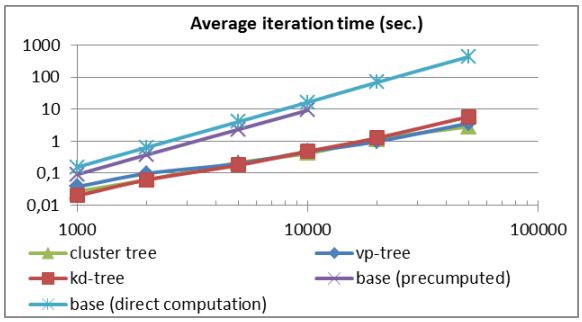


Figure 8. Dependency of the average time per one iteration on the sample size for the base algorithm (COREL dataset)

Some examples of the nonlinear mapping obtained using space partitioning structure and the base algorithm for a subset containing 5000 instances of the MNIST dataset is shown in fig. 9.

An example of the nonlinear mapping using the described approach for 60 000 objects of the MNIST database is shown in fig. 10. The database was processed in less than 30 minutes including error estimation at each iteration (19 minutes without error estimation). The multidimensional data representation error was equal to 0.14.

Conclusion

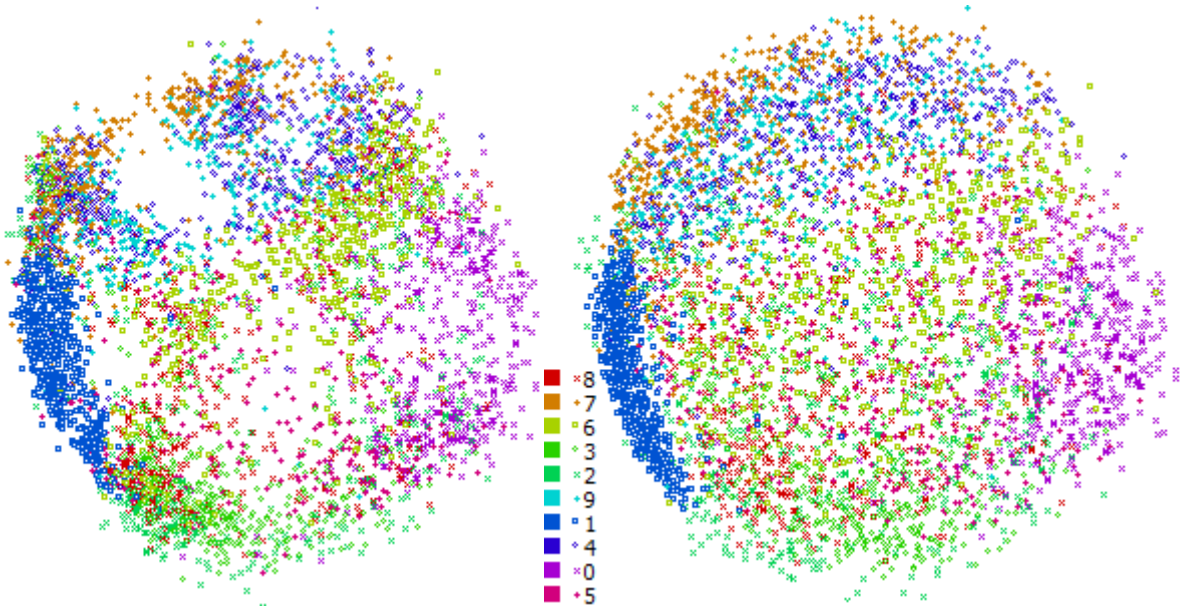
In this paper, we conducted a study of several space partitioning structures namely metric trees (vp-trees), kd-trees, and cluster trees to speed up the nonlinear mapping. The study showed that the quality of mapping is weakly dependent on the type of the structure but the average iteration time was different for the considered structures. For kd-tree the number of reference nodes was significantly larger than for the other structures, hence the average iteration time was larger.

Thus for the presented data sets metric trees (vp-trees) and cluster trees can be efficiently applied to partition the input space for the considered problem. Using of the considered data structures made it possible to generate low-dimensional embeddings for relatively large datasets in a comfortable time.

At the same time the data representation error for the base algorithm had slightly lower values. To improve the quality of the mapping one can use the considered structures increasing the threshold parameter T or use the obtained low-dimensional configuration as an initial configuration for the base method.

ACKNOWLEDGMENTS

This work was financially supported by the Russian Foundation for Basic Research, project № 15-07-01164-a.



(a) **(b)**

Figure 9. Nonlinear mapping for 5000 instances of MNIST database:
(a) cluster tree, 207 iterations, threshold T = 1, multidimensional data representation error ε = 0.14255;
(b) base method, 194 iterations, multidimensional data representation error ε = 0.14077

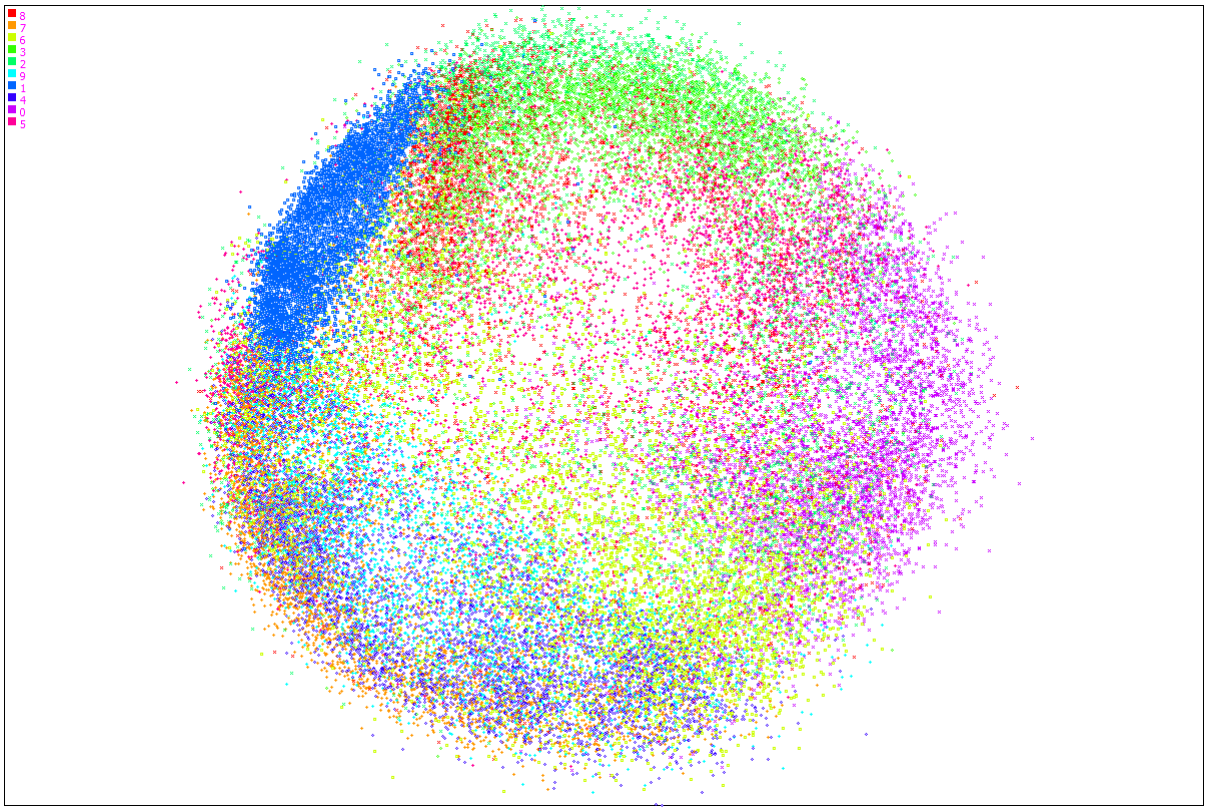


Figure 10. Nonlinear mapping for training set of MNIST database (60,000 examples)

 ISBN 978-80-86943-65-7

REFERENCES

[Bar86] Barnes J., Hut P. A hierarchical O(N log N) force-calculation algorithm // Nature, 324 (4). – 1986. – pp. 446–449.

[Ben75] Bentley J. L. Multidimensional binary search trees used for associative searching // Communications of the ACM, 18 (9). – 1975. – 509

[Cha96] M. Chalmers. A linear iteration time layout algorithm for visualising high-dimensional data // In Proceedings of IEEE Visualization. – 1996. – pp. 127–132.

[COR] https://archive.ics.uci.edu/ml/datasets/Corel+Image+Features

[Fin74] Finkel R., Bentley J.L. Quad Trees: A Data Structure for Retrieval on Composite Keys. – 1974. - Acta Informatica, 4 (1). – pp. 1–9.

[Fru91] Fruchterman T., Reingold E. Graph Drawing by Force-directed Placement. //Software – Practice and Experience. 1991. vol. 21, no. 11. pp. 1129-1164.

[Gre87] Greengard L., Rokhlin V. A fast algorithm for particle simulations. Journal of Computational Physics, 73. – 1987. – pp. 325–348.

[Har73] Haralick R.M., Shanmugam K., Dinstein I. Texture features for image classification. IEEE Trans. on Sys. Man. and Cyb. SMC-3(6), 1973

[LAN] https://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite)

[Lee77] Lee R.C.T., Slagle J.R., Blum H. A Triangulation Method for the Sequential Mapping of Points from N-Space to Two-Space //IEEE Transactions on Computers. – 1977. - V. 26, №3. - pp. 288-292.

[MNI] http://yann.lecun.com/exdb/mnist/

[Mya12] E.V. Myasnikov A Nonlinear Method for Dimensionality Reduction of Data Using Reference Nodes // Pattern Recognition and Image Analysis, 2012, Vol. 22, No. 2, pp. 337–345.

[Pek99] Pekalska E., de Ridder D., Duin R.P.W., Kraaijveld M.A. A new method of generalizing Sammon mapping with application to algorithm speed-up. //Proc. ASCI'99, 5th Annual Conf. of the Advanced School for Computing and Imaging - Heijen, The Netherlands. – 1999. - June 15-17. - P. 221-228.

[Qui01] Quigley A., Eades P. FADE: Graph Drawing, Clustering, and Visual Abstraction". //Proceedings of the 8th International Symposium on Graph Drawing. 2001. pp. 197–210.

[Sal94] Salmon J.K., Warren M.S. Skeletons from the Treecode Closet // J. Comp. Phys. V.111 – 1994. – pp. 136-155.

[Sam06] Samet, H. Foundations of multidimensional and metric data structures // Morgan Kaufmann. – 2006. – 1024 p.

[Sam69] Sammon J.W., Jr. A nonlinear mapping for data structure analysis. //IEEE Transactions on Computers. – 1969. - V. C-18, No.5. - P.401-409.

[San89] Sanger T.D. Optimal unsupervised learning in a single-layer linear feedforward neural network // Neural Networks, 2 (6). – 1989. – pp. 459–473.

[Sti95] Stricker M., Orengo M. Similarity of color images // In Proc. SPIE Conf. on Vis. Commun. and Image Proc., 1995

[Swa91] Swain M., Ballard D. Color indexing. International Journal of Computer Vision. 7(1), 1991.

[Tou74] Tou J.T., Gonzalez R.C. Pattern Recognition Principles // Addison-Wesley Publishing Company. – 1974.

[Uhl91] Uhlmann J. Satisfying General Proximity/Similarity Queries with Metric Trees // Information Processing Letters, 40 (4). - 1991.

[Van13] van der Maaten L.J.P. Barnes-Hut-SNE // In Proceedings of the International Conference on Learning Representations. - 2013.

[Vla14] Vladymyrov M., Carreira-Perpiñán, M.Á. Linear-time training of nonlinear low-dimensional embeddings // 17th International Conference on Artificial Intelligence and Statistics (AISTATS 2014) – 2014. - pp. 968-977.

[Yan13] Z. Yang, J. Peltonen, and S. Kaski. Scalable optimization of neighbor embedding for visualization // In Proc. of the Int. Conf. on Machine Learning. - 2013.

[Yia93] Yianilos Data structures and algorithms for nearest neighbor search in general metric spaces // Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms. - 1993. - pp. 311–321.

A GPU-Accelerated Augmented Lagrangian Based L^1-mean Curvature Image Denoising Algorithm Implementation

Mirko Myllykoski[1]
University of Jyväskylä
mirko.myllykoski@jyu.fi

Roland Glowinski[2]
University of Houston
roland@math.uh.edu

Tommi Kärkkäinen[1]
University of Jyväskylä
tommi.karkkainen@jyu.fi

Tuomo Rossi[1]
University of Jyväskylä
tuomo.j.rossi@jyu.fi

Abstract

This paper presents a graphics processing unit (GPU) implementation of a recently published augmented Lagrangian based L^1-mean curvature image denoising algorithm. The algorithm uses a particular alternating direction method of multipliers to reduce the related saddle-point problem to an iterative sequence of four simpler minimization problems. Two of these subproblems do not contain the derivatives of the unknown variables and can therefore be solved point-wise without inter-process communication. In particular, this facilitates the efficient solution of the subproblem that deals with the non-convex term in the original objective function by modern GPUs. The two remaining subproblems are solved using the conjugate gradient method and a partial solution variant of the cyclic reduction method, both of which can be implemented relatively efficiently on GPUs. The numerical results indicate up to 33-fold speedups when compared against a single-threaded CPU implementation. The pointwise treated subproblem that takes care of the non-convex term in the original objective function was solved up to 76 times faster.

Keywords

augmented Lagrangian method, GPU computing, image denoising, image processing, mean curvature, OpenCL

1 INTRODUCTION

Image denoising, or more generally noise reduction, is a process in which a given noisy signal, such as a digital image, is cleared from excess noise. This process has numerous applications since all recording devices have some traits that make them susceptible to interference. For example, thermal noise effecting digital image sensors is a typical source of interference in digital photography. The noise must be removed, or at least significantly reduced, before essential information can be successfully extracted from an image.

Image denoising methods are divided into multiple subcategories. For example, wavelet methods are based around the idea of decomposing the image into the wavelet basis and shrinking (or otherwise modifying) the wavelet coefficients in order to denoise the image. A somewhat similar approach is to process the image in the frequency domain using the fast Fourier transformation (FFT) method. Statistical methods, on the other hand, utilize local statistical information, such as median and mean, from the neighboring pixels that fall within an appropriately selected window.

The most relevant sub-category to the topic of this paper is referred to as variational-based methods. These methods treat the noisy image as a discretely differentiable function and denoise the image using derivate information. In more formal terms, let Ω be a rectangular domain of $\mathbb{R}^2$ and the function $f : \Omega \to \mathbb{R}$ represent the given noisy image. We want to find a function $u : \Omega \to \mathbb{R}$ that is a representative of the desired denoised image. A variety of variational-based techniques have been developed and a significant proportion of them (see, e.g., [Mum94, Rud92]) are based on solving an unconstrained minimization problem of the form

$$\begin{cases} u \in V, \\ \mathscr{J}(u) \leq \mathscr{J}(v), \forall v \in V, \end{cases} \tag{1}$$

where

$$\mathscr{J}(v) = \varepsilon \mathscr{J}_r(v) + \mathscr{J}_f(v), \tag{2}$$

V is a suitable function space, and $\varepsilon > 0$. Here $\mathscr{J}_r$ is so-called regularization term and $\mathscr{J}_f$ is so-called fidelity

[1] University of Jyväskylä, Department of Mathematical Information Technology, P.O. Box 35, FI-40014 University of Jyväskylä, Finland.
[2] University of Houston, Department of Mathematics, Houston, TX 77204, USA.

term whose role is to fit the obtained solution u to the noisy data f. The work in this field of research is aimed primarily at finding a suitable regularization term that is able to detect noise but preserves as much relevant information as possible.

During the last two decades, the variational-based image denoising scene has been dominated to a large extent by Rudin–Osher–Fatemi (ROF) method [Rud92] which uses the following objective function:

$$\mathscr{J}(v) = \varepsilon \int_\Omega |\nabla v|\, dx + \frac{1}{2}\int_\Omega |f-v|^2\, dx. \qquad (3)$$

Here, $\int_\Omega |\nabla v|\, dx$ is so-called total variation norm (TV norm) and the function space V made out of function whose total variation is bounded (a.k.a BV space). This very popular method has, however, some well-known drawbacks, such as the loss of image contrast, the smearing of corners, and the staircase effect.

Some attempts to remedy these drawbacks have led to higher-order variational-models which seek to take advantage of the higher-order derivatives. One approach suggested by Zhu and Chan [Zhu12] is to treat an image $v : \Omega \to \mathbb{R}$ as a surface in $\Omega \times \mathbb{R}$ and utilize the surface mean curvature information in the regularization term. More specifically, the surface in question is defined by the equation $F_v(x,y,z) = v(x,y) - z = 0$ and the mean curvature of the function F_v is given by

$$\kappa(F_v) = -\nabla\cdot \frac{\nabla F_v}{|\nabla F_v|} = -\nabla\cdot \frac{\nabla v}{\sqrt{1+|\nabla v|^2}}. \qquad (4)$$

All in all, the objective function used in the model suggested by Zhu and Chan is of the form:

$$\mathscr{J}(v) = \varepsilon \int_\Omega |\kappa(F_v)|\, dx + \frac{1}{2}\int_\Omega |f-v|^2\, dx. \qquad (5)$$

This model is commonly known these days as the L^1-mean curvature denoising model. As noted in [Zhu12], this model has the ability to remove noise without the undesirable drawbacks associated with the ROF model. However, the non-convex and non-smooth nature of the objective function (5) makes the problem very difficult to solve.

1.1 Related work

Although the model suggested by Zhu and Chan is very difficult to solve as noted above, effective solution algorithms for this particular formulation have been proposed, for example, in [Zhu13] and [Myl15]. The solution algorithm presented in [Zhu13] uses an augmented Lagrangian based approach and solves the arising saddle-point problem using a particular alternating direction method of multipliers. This leads to an iterative sequence of five simpler minimization problems. These subproblems can be solved using explicit formulas and the FFT method. The solution algorithm presented in [Myl15] uses the same alternating direction approach as [Zhu13] but relies on a different type of augmented Lagrangian functional. Two of the four arising subproblems can be solved pointwise using the Newton's method, a bisection search algorithm, and explicit formulas. The two remaining subproblems can be solved using the conjugate gradient method and a partial solution variant of the cyclic reduction (PSCR) method [Kuz85, Kuz96, Vas84, Val85].

It should be noted that the model suggested by Zhu and Chan is closely related to the model depicted in [Lys04]. The model uses the following regularization term:

$$\mathscr{J}_r(v) = \int_\Omega \left|\nabla\cdot \frac{\nabla v}{|\nabla v|}\right| dx. \qquad (6)$$

In [Lys04], the authors explained that their goal was to minimize the "TV norm" of the unit normal vectors of the level curves of the image. An alternative interpretation is that the regularization term (6) measures the total mean curvature at every level curve of the image. In contrast, the regularization term in the model suggested by Zhu and Chan measures the total mean curvature at the surface defined by the image (graph).

From a practical point of view the most relevant connection comes from the fact that the resulting model is often regularized in such a way that $|\nabla v|$ in (6) is replaced by $|\nabla v|_\beta = \sqrt{|\nabla v|^2 + \beta}$, $\beta > 0$. Thus, the solution algorithms developed for this denoising model and its variants (see, e.g., [Bri10, Sun14, Yan14]) could be in principle generalized for the model suggested by Zhu and Chan by taking $\beta = 1$.

The solution algorithm presented in [Bri10] solves the related Euler-Lagrange (EL) equation using a stabilized fixed point method and a geometric multigrid (MG) algorithm. The authors in [Sun14] aimed to improve upon that by introducing an additional operator splitting step. They then moved on to solving the EL equations associated with the related constrained minimization problem using a linearized fixed point method and a nonlinear MG method. In [Yan14], the problem is tacked with a relaxed fixed point method and a homotopy algorithm. The papers [Bri10, Sun14, Yan14] included comparisons where the value of the parameter β was varied (including the case $\beta = 1$). In other aspects these three recent papers were mostly interested in the case where the regularization term (6) is replaced by

$$\mathscr{J}_r(v) = \int_\Omega \left(\nabla\cdot \frac{\nabla v}{|\nabla v|_\beta}\right)^2 dx. \qquad (7)$$

and $\beta \ll 1$.

1.2 Motivation and structure

Now that the overall context and main related works have been dealt with, we can move on to the main topic of this paper. We present a graphics processing unit (GPU) implementation of the solution algorithm depicted in [Myl15] and compare the GPU implementation against a single-threaded CPU implementation.

Our main motivation is that the most demanding step in the solution algorithm is a pointwise treated subproblem that handles the non-convex term in the original objective function. This makes the solution algorithm very suitable for GPU computation since the solution of this subproblem does not require inter-process communication. Thus, it is very likely that GPU-acceleration would bring significant performance benefits.

The rest of this paper is organized as follows: Section 2 describes the augmented Lagrangian based image denoising algorithm closely following the presentation in [Myl15]. Section 3 gives a brief introduction to GPU computing and describes the GPU implementation. Section 4 presents the numerical results, comparisons, and discussion. The final conclusions are given in Section 5.

2 SOLUTION ALGORITHM

2.1 Augmented Lagrangian formulation

Augmented Lagrangian techniques are a well-established framework for analyzing (constrained) optimization problems and deriving solution algorithms for such problems. When applied to convex minimization problems, the basic idea is to decompose the problem with the help of auxiliary variables. This so-called operator splitting operation effectively splits the problem into subproblems which can be treated separately using methods that are best suited for each subproblem. This greatly improves the effectiveness of the resulting solution algorithm.

The addition of these new auxiliary variables leads to a new constrained minimization problem that has the same minimizer as the original minimization problem. This constrained minimization is then associated with a suitable augmented Lagrangian functional whose saddle-point correspond to the minimizer of the constrained minimization problem. The saddle-points can be solved by, for example, using an alternating direction type approach. See, for example, [For83] for further information.

Although the objective function in the model suggested by Zhu and Chan is not convex, we will now describe a formal augmented Lagrangian formulation for the minimization problem similarly to [Myl15]. To begin with, let us define

$$\begin{aligned}\Upsilon = [V \times \mathbf{E}_{12} \times H(\Omega;\mathrm{div}) \times L^2(\Omega)] \\ \times [(L^2(\Omega))^2 \times (L^2(\Omega))^2 \times L^2(\Omega)],\end{aligned} \tag{8}$$

where

$$H(\Omega;\mathrm{div}) = \left\{\mathbf{q} \in (L^2(\Omega))^2 : \nabla\cdot\mathbf{q} \in L^2(\Omega)\right\} \tag{9}$$

and

$$\begin{aligned}\mathbf{E}_{12} = \Bigg\{ &(\mathbf{q}_1,\mathbf{q}_2) \in \left(L^2(\Omega)\right)^{2\times 2} \\ &: \mathbf{q}_2 = \frac{\mathbf{q}_1}{\sqrt{1+|\mathbf{q}_1|^2}} \Bigg\}.\end{aligned} \tag{10}$$

Following the remarks made in [Myl15], we take $V = H^2(\Omega)$. The minimization problem (1) with $\mathscr{J}$ defined by (5) is associated with the following augmented Lagrangian functional $\mathscr{L} : \Upsilon \to \mathbb{R}$:

$$\begin{aligned}&\mathscr{L}(v,\mathbf{q}_1,\mathbf{q}_2,\mathbf{q}_3,\varphi;\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\mu_3) \\ &\quad= \varepsilon\int_\Omega |\varphi| dx + \frac{1}{2}\int_\Omega |f-v|^2\,dx \\ &\quad+ \frac{r_1}{2}\int_\Omega |\nabla v - \mathbf{q}_1|^2\,dx + \int_\Omega \boldsymbol{\mu}_1\cdot(\nabla v - \mathbf{q}_1)\,dx \\ &\quad+ \frac{r_2}{2}\int_\Omega |\mathbf{q}_2 - \mathbf{q}_3|^2\,dx + \int_\Omega \boldsymbol{\mu}_2\cdot(\mathbf{q}_2 - \mathbf{q}_3)\,dx \\ &\quad+ \frac{r_3}{2}\int_\Omega |\nabla\cdot\mathbf{q}_3 - \varphi|^2\,dx \\ &\quad+ \int_\Omega \mu_3(\nabla\cdot\mathbf{q}_3 - \varphi)\,dx,\end{aligned} \tag{11}$$

where $(\mathbf{q}_1,\mathbf{q}_2) \in \mathbf{E}_{12}$ and $r_i > 0$, $i = 1,2,3$. Above, $\mathbf{q}_1$, $\mathbf{q}_2$, $\mathbf{q}_3$, and φ are the previously mentioned auxiliary variables, and $\boldsymbol{\mu}_1$,$\boldsymbol{\mu}_2$, and μ_3 are called Lagrange multipliers. Note that the non-convex term $\mathbf{q}_2 = \frac{\mathbf{q}_1}{\sqrt{1+|\mathbf{q}_1|^2}}$ is treated by projection in (10) and thus does not appear in the augmented Lagrangian functional $\mathscr{L}$.

Now, if we can find a saddle-point

$$\omega = (u,\mathbf{p}_1,\mathbf{p}_2,\mathbf{p}_3,\psi;\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2,\lambda_3) \in \Upsilon \tag{12}$$

for the augmented Lagrangian $\mathscr{L}$, that is

$$\begin{aligned}&\mathscr{L}(u,\mathbf{p}_1,\mathbf{p}_2,\mathbf{p}_3,\psi;\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\mu_3) \\ &\le \mathscr{L}(u,\mathbf{p}_1,\mathbf{p}_2,\mathbf{p}_3,\psi;\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2,\lambda_3) \\ &\le \mathscr{L}(v,\mathbf{q}_1,\mathbf{q}_2,\mathbf{q}_3,\varphi;\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2,\lambda_3),\end{aligned} \tag{13}$$

for all $(v,\mathbf{q}_1,\mathbf{q}_2,\mathbf{q}_3,\varphi;\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\mu_3) \in \Upsilon$, then

$$\begin{aligned}\mathbf{p}_1 = \nabla u,\ \mathbf{p}_2 = \frac{\mathbf{p}_1}{\sqrt{1+|\mathbf{p}_1|^2}}, \\ \mathbf{p}_3 = \mathbf{p}_2,\ \psi = \nabla\cdot\mathbf{p}_3,\end{aligned} \tag{14}$$

and, more importantly, u is a local minimizer of the minimization problem (1) with $\mathscr{J}$ defined by (5).

2.2 Subproblems

In [Myl15], the saddle-point problem (12) – (13) is solved using a particular alternating direction method

of multipliers called ALG-2 [For83, Glo89]. The idea is to minimize the augmented Lagrangian functional (11) one variable at a time until the method converges. The Lagrange multipliers are update accordingly after each iteration.

Since we have five variables and two of the auxiliary variables ($\mathbf{q}_1$ and $\mathbf{q}_2$) are coupled together, this leads to an iterative sequential solution of four subproblems. More precisely, the task of finding a saddle-point for the augmented Lagrangian functional (11) is transformed into one smooth but nonlinear and non-convex minimization problem in $\mathbb{R}^2$, one purely explicit pointwise treated minimization problem, and two linear minimization problems with positive definite and symmetric coefficient matrices.

Each outer iteration is defined as follows: Let $(u^n, \mathbf{p}_1^n, \mathbf{p}_2^n, \mathbf{p}_3^n, \psi^n; \boldsymbol{\lambda}_1^n, \boldsymbol{\lambda}_2^n, \lambda_3^n) \in \Upsilon$ be the output of the previous iteration. *The first subproblem* minimizes the augmented Lagrangian functional (11) with respect to the pair $(\mathbf{q}_1, \mathbf{q}_2) \in \mathbf{E}_{12}$. Using the nonlinear relation in (10), the function $\mathbf{p}_1^{n+1}$ can be solved pointwise from the following non-convex minimization problem:

$$\mathbf{x} = \arg\min_{\mathbf{y}\in\mathbb{R}^2} \left[\frac{|\mathbf{y}|^2}{2} \left(r_1 + \frac{r_2}{1+|\mathbf{y}|^2} \right) - \left(\mathbf{b}_1 + \frac{\mathbf{b}_2}{\sqrt{1+|\mathbf{y}|^2}} \right) \cdot \mathbf{y} \right], \tag{15}$$

where $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^2$ depend on the other variables. Or, alternatively, by noticing that the following nonlinear relation must hold

$$\mathbf{x} = \alpha \left(\mathbf{b}_1 + \frac{\mathbf{b}_2}{\sqrt{1+|\mathbf{x}|^2}} \right), \tag{16}$$

where $\alpha \geq 0$, we can write the two-dimensional minimization problem (15) as

$$\mathbf{x} = \frac{\rho}{\left| \mathbf{b}_1 + \frac{\mathbf{b}_2}{\sqrt{1+\rho^2}} \right|} \left(\mathbf{b}_1 + \frac{\mathbf{b}_2}{\sqrt{1+\rho^2}} \right), \tag{17}$$

where

$$\rho = \arg\min_{\sigma\in[0,\infty)} \left[\frac{\sigma^2}{2} \left(r_1 + \frac{r_2}{1+\sigma^2} \right) - \sigma \left| \mathbf{b}_1 + \frac{\mathbf{b}_2}{\sqrt{1+\sigma^2}} \right| \right]. \tag{18}$$

In *the second subproblem* we minimize the augmented Lagrangian functional (11) with respect to the variable $\mathbf{q}_3$. More specifically, we solve the following linear vector-valued minimization problems with positive definite and symmetric coefficient matrix:

$$\begin{aligned} \mathbf{p}_3^{n+1} &\in H(\Omega; \mathrm{div}), \\ r_2 \int_\Omega \mathbf{p}_3^{n+1} \cdot \mathbf{q}\, dx + r_3 \int_\Omega \nabla\cdot \mathbf{p}_3^{n+1} \nabla\cdot \mathbf{q}\, dx &= \int_\Omega \left(r_2 \mathbf{p}_2^{n+1} + \boldsymbol{\lambda}_2^n \right) \cdot \mathbf{q}\, dx \\ &+ \int_\Omega (r_3 \psi^n - \lambda_3^n) \nabla\cdot \mathbf{q}\, dx, \\ \forall \mathbf{q} &\in H(\Omega; \mathrm{div}). \end{aligned} \tag{19}$$

The third subproblem minimizes the augmented Lagrangian functional (11) with respect to the variable φ and is of the form:

$$\psi^{n+1} = \arg\min_{\varphi\in L^2(\Omega)} \left[\varepsilon \int_\Omega |\varphi|\, dx + \frac{r_3}{2} \int_\Omega |\varphi|^2\, dx - \int_\Omega \left(r_3 \nabla\cdot \mathbf{p}_3^{n+1} + \lambda_3^n \right) \varphi\, dx \right]. \tag{20}$$

The minimization problem (20) has a closed-form solution

$$\psi^{n+1}(x) = \frac{1}{r_3} \operatorname{sgn}(\xi(x)) \max(0, |\xi(x)| - \varepsilon), \tag{21}$$

with $\xi(x) = (r_3 \nabla\cdot \mathbf{p}_3^{n+1} + \lambda_3^n)(x)$.

The fourth subproblem minimizes the augmented Lagrangian functional (11) with respect to the variable v. The subproblem can be written as the following linear scalar-valued minimization problems with positive definite, symmetric, and separable coefficient matrix:

$$\begin{aligned} u^{n+1} &\in V, \\ r_1 \int_\Omega \nabla u^{n+1} \cdot \nabla v\, dx + \int_\Omega (u^{n+1} - f)\, v\, dx &= \int_\Omega \left(r_1 \mathbf{p}_1^{n+1} - \boldsymbol{\lambda}_1^n \right) \cdot \nabla v\, dx, \forall v \in V. \end{aligned} \tag{22}$$

Finally, *the Lagrange multipliers* are updated as follows:

$$\begin{aligned} \boldsymbol{\lambda}_1^{n+1} &= \boldsymbol{\lambda}_1^n + r_1(\nabla u^{n+1} - \mathbf{p}_1^{n+1}), \\ \boldsymbol{\lambda}_2^{n+1} &= \boldsymbol{\lambda}_2^n + r_2(\mathbf{p}_2^{n+1} - \mathbf{p}_3^{n+1}), \\ \lambda_3^{n+1} &= \lambda_3^n + r_3(\nabla\cdot \mathbf{p}_3^{n+1} - \psi^{n+1}). \end{aligned} \tag{23}$$

2.3 Finite element realization

The domain Ω is triangulated using a uniform finite element triangulation $\mathscr{T}_h$. The function space V is approximated by a piecewise linear finite element space

$$V_h = \left\{ v \in C^0(\bar{\Omega}) : v|_T \in P_1,\ \forall\, T \in \mathscr{T}_h \right\}, \tag{24}$$

where P_1 is the space of the polynomials of two variables of degree ≤ 1. The spaces $(L^2(\Omega))^2$ and

$H(\Omega;\mathrm{div})$ are approximated by the following piecewise constant finite element space:

$$\mathbf{Q}_h = \left\{ \mathbf{q} \in (L^\infty)^2 : \mathbf{q}|_T \in (P_0)^2, \forall\, T \in \mathscr{T}_h \right\}, \qquad (25)$$

where P_0 is the space of the constant functions. Clearly, we have $\nabla V_h \subset \mathbf{Q}_h$.

Let $\{X_j\}_{j=1}^{N_h}$ be the set of vertices of $\mathscr{T}_h$ and $\mathbf{q} \in \mathbf{Q}_h$. The divergence operator is approximated by using an appropriate discrete Green's formula and the trapezoidal rule as follows:

$$(\mathrm{div}_h \mathbf{q})(X_j) = -\frac{3}{|\Omega_j|} \int_{\Omega_j} \mathbf{q} \cdot \nabla w_j \, dx, \qquad (26)$$

where X_j is a vertex that does not belong to $\partial\Omega$, Ω_j is the polygon that is the union of those triangles of $\mathscr{T}_h$ that have X_j as a common vertex, $|\Omega_j|$ is the measure of Ω_j, and the shape function $w_j \in V_h$ is uniquely defined as

$$\begin{cases} w_j(X_j) = 1, \\ w_j(X_k) = 0,\ k \neq j. \end{cases} \qquad (27)$$

3 GPU IMPLEMENTATION

3.1 GPU computing and OpenCL

The GPU implementation presented in this paper is written using the OpenCL framework. This section introduces the reader to general OpenCL concepts and terminology. Some additional information related to Nvidia's current hardware is provided since that information is essential for the understanding of the implementation and obtained numerical results.

A contemporary high-end GPU contains thousands of processing elements (cores) which are grouped into multiple computing units. The processing elements inside the same computing unit share a fast (on-chip) memory space called local memory which can be used for sharing data among the processing elements. The local memory is divided into 32-bit (or 64-bit) memory banks organized in such a way that successive 32-bit (or 64-bit) words map to successive memory banks. Multiple processing elements also share the same scheduler, which means that the processing elements are executing the program code in a synchronous manner. In addition to the local memory, all processing elements can access a much larger but slower (off-chip) memory space called global memory. The global memory can serve memory requests at the optimal rate when processing elements are synchronously accessing data that is located inside a same memory block.

GPU-side program code execution is based on the concept of a special kind of subroutine called (OpenCL) kernel. All work-items (threads) start from the beginning of the kernel but each work-item is given a unique index number which allows the execution paths of different work-items to branch off. The work-items are grouped into work groups which are also given unique index numbers and a work group can share a portion of the local memory. Nvidia uses the term warp when referring to a set of work-items that are executed together in a synchronized manner. Diverging execution paths, also known as warp divergences, lead to a suboptimal performance as all the necessary paths have to be evaluated by the whole warp. In contemporary Nvidia GPUs the warp size is 32 work-items.

3.2 General notes

The GPU implementation is principally identical with the CPU implementation described in [Myl15] but the low level details vary considerably. The less simplified two-dimensional form of the critical non-convex subproblem (15) is initially solved using the Newton's method, whose solution candidate is then tested against the explicit relation (16). If the solution candidate does not fulfill the explicit relation, the implementation proceeds to the one-dimensional form (18) which is solved using the bisection search algorithm as described in [Myl15].

The linear vector-valued subproblem (19) is solved using the conjugate gradient algorithm without preconditioning. While more generalized GPU implementations have been presented in the past (see, for example, [Ame10, Bol03, Hel12]), the conjugate gradient solver used in the GPU implementation described in this paper was tailored for this specific subproblem and the matrix-vector multiplication operation was hard coded into the kernels. The explicit subproblem (20) is solved using the closed form solution (21) and the linear scalar-valued subproblem (22) is solved using the PSCR method.

All computational operations are carried out in the GPU side. The floating point division operation was accelerated using a Newton-Raphson division algorithm [Fly70] and an initial approximation that leads to full double precision accuracy with only four iterations [Par92].

3.3 Element numbering

The elements of the finite element space V_h are numbered in a row-wide fashion. This means that the coefficient matrix in the linear scalar-valued subproblem (22) is block tridiagonal and presentable in a separable form using so-called Kronecker matrix tensor product. This is required by the PSCR method.

The numbering of the elements of the finite element space $\mathbf{Q}_h$ can be chosen more freely. Figure 1 shows two possible numbering schemes. If the numbering scheme shown on the left (referred to hereinafter as

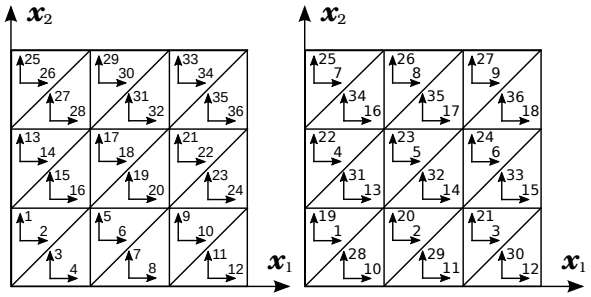


Figure 1: Two possible numbering schemes for the elements of the finite element space $\mathbf{Q}_h$ (3×3 grid): the dense numbering scheme (on the left) and the sparse numbering scheme (on the right).

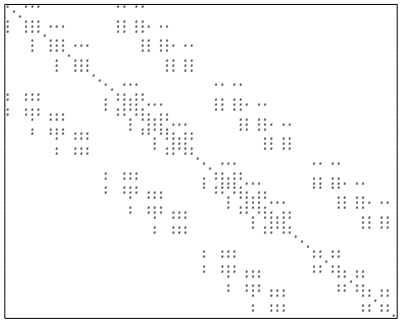


Figure 2: The non-zero elements of the coefficient matrix in the linear vector-valued subproblem (19) when the dense numbering scheme is used (4×4 grid).

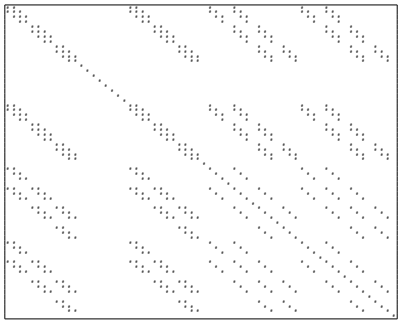


Figure 3: The non-zero elements of the coefficient matrix in the linear vector-valued subproblem (19) when the sparse numbering scheme is used (4×4 grid).

the dense numbering scheme) is chosen, then the coefficient matrix in the linear vector-valued subproblem (19) is of the form shown in Figure 2. On the one hand, if the numbering scheme shown on the right (referred to hereinafter as the sparse numbering scheme) is chosen, then the coefficient matrix is of the form shown in Figure 3. Each numbering scheme has its own advantages and disadvantages.

The dense numbering scheme leads to a more optimal global memory access pattern during the solution of the linear vector-valued subproblem (19) as the non-zero elements of the coefficient matrix are packed tightly to three bands and elements of each band can be shared among work-items using the local memory. This is particularly important because many contemporary high end GPUs have an extremely high peak floating-point performance but a relatively low peak global memory bandwidth. Thus, the use of the global memory should be kept at minimum. The most significant downside of this numbering scheme is that a straightforward implementation would lead to warp divergences throughout the implementation. Most of these warp divergences could be avoided by re-arranging the computational tasks appropriately with the help of the local memory. However, this re-arranging would complicate the implementation considerably and introduce memory bank conflicts in many places.

The sparse numbering scheme leads to a simpler implementation but the pattern of non-zero elements in the coefficient matrix is much more fragmented. This means that less data can be shared between the work-items using the local memory and, thus, the global memory usage increases significantly. Despite this, the sparse numbering scheme was chosen for the GPU implementation described in this paper because it was not clear whether this choice would lead to an actual global memory bottleneck that would limit the performance of the whole GPU implementation. In addition, if the dense numbering scheme is chosen, then the increased complexity in the other parts of the implementation might negate the potential benefits. The reference CPU implementation uses the dense numbering scheme because it allows more effective utilization of the CPU caches.

3.4 PSCR implementation

The PSCR method [Kuz85, Kuz96, Vas84, Val85] is a block cyclic reduction type direct solver which can be applied to certain separable block tridiagonal linear systems. To put it briefly, the PSCR method solves the linear scalar-valued subproblem (22) by recursively eliminating block-rows from the corresponding linear system and then solves the generated sub-systems in the reverse order during so-called back substitution stage. Each reduction and back substitution step produces a large set of tridiagonal linear system.

The GPU implementation of the PSCR method used in this paper is based on the radix-4 variant described in [Ros99] and it is in many respects similar to the simplified radix-4 block cyclic reduction GPU implementation presented in [Myl13]. However, the GPU implementation used in this paper is much more generalized as the problem size can be arbitrary.

The arising tridiagonal subproblems are solved using the cyclic reduction (CR) [Hoc65], the parallel cyclic reduction (PCR) [Hoc81] and the Thomas [Con80] methods. If a tridiagonal system does not fit into the allocated local memory buffer,then the system size is first reduced using the CR method and the global

memory data permutations depicted in [Myl13]. The tridiagonal systems that do fit into the allocated local memory buffer are solved using a CR-PCR-Thomas hybrid method. The CR stage of the hybrid solver uses the local memory data permutations depicted in [Myl13]. The PCR stage further splits the reduced tridiagonal systems into smaller subsystems which are eventually solved using the Thomas method in a manner similar to [Dav11, Kim11]. Somewhat similar tridiagonal solver techniques have been used, for example, in [Göd11, Lam12, Zha10].

4 NUMERICAL RESULTS

4.1 Test setting

GPU tests were carried out on a few years old consumer level Nvidia GeForce GTX580 GPU and a high-end computing orientated Nvidia Tesla K40c GPU. The CPU implementation is the same as was used in [Myl15]. It is written using C++ and Fortran. It utilizes a single-threaded variant of the radix-4 PSCR method presented in [Ros99]. CPU tests were carried out using an Intel Xeon E5-2630 v2 (2.60GHz) CPU. All the tests were performed using double precision floating point arithmetic and ECC memory (excluding the GTX580 GPU which does not support ECC).

Four test images (shown in Figure 4) were used in the numerical tests: Test9, Lena, Boat, and Mercado. In addition, four different sized versions of the test image Mercado were included: 256×256, 512×512, 1024×1024, and 2048×2048. The dimensions of the other test images are 512×512. The original test images were scaled to the range $[0,1]$. Two sets of noisy input images were generated: uniformly distributed zero-mean noise with the standard deviation $\sigma = 0.025$ and uniformly distributed zero-mean noise with the standard deviation $\sigma = 0.1$.

Based on the remarks made in [Myl15], the following initialization was used:

$$u^0 = 0,\ \mathbf{q}_1^0 = \mathbf{q}_2^0 = \mathbf{q}_3^0 = 0,\ \psi^0 = 0,\quad \boldsymbol{\lambda}_1^0 = \boldsymbol{\lambda}_2^0 = 0,\ \lambda_3^0 = 0. \tag{28}$$

In the same way, the stopping criterion read as

$$\frac{|\mathscr{L}_h(\upsilon^n) - \mathscr{L}_h(\upsilon^{n+1})|}{|\mathscr{L}_h(\upsilon^n)|} < 10^{-4}, \tag{29}$$

where $\mathscr{L}_h$ is the discrete counterpart of the augmented Lagrangian functional (11) and $\upsilon^n = (u^n, \mathbf{p}_1^n, \mathbf{p}_2^n, \mathbf{p}_3^n, \psi^n; \boldsymbol{\lambda}_1^n, \boldsymbol{\lambda}_2^n, \lambda_3^n)$. The parameter ε and the Lagrangian multipliers were coupled as follows: $\varepsilon = r_0 h$, $r_1 = 10 r_0 h$, $r_2 = 5 r_0$, and $r_3 = 5 r_0 h^2$, where $r_0 > 0$. We took $h = 0.005$ for the spatial discretization step.

4.2 Comparisons

Figure 4 shows the original test images, the generated input images ($\sigma = 0.1$), and the obtained output images. Table 1 shows the used parameter values, iteration counts and execution times for the Intel Xeon CPU. The input images and parameter values were the same for the three platforms. In addition, as the GPUs used in the numerical experiments are fully IEEE 754 compliant, the iteration counts, objective function values, and output images were also identical.

The "Whole" column shows the average total per iteration execution times; the "Sub. #1", "Sub. #2", "Sub. #3", and "Sub. #4" columns show the average per iteration execution times for each subproblem; and the "Misc." column shows the combined average per iteration execution times for augmentation term update kernels and an objective function value computation kernel. The CPU results show that a significant portion of the total execution time goes to solving the critical non-convex subproblem (15).

Tables 2 and 3 show the average per iteration execution times and the obtained speedups for the GTX580 and K40c GPUs, respectively. The GTX580 was on average 15.6 times faster than the Xeon CPU. The highest speedups were obtained in the case of the synthetic test image Test9 in which case the GTX580 GPU was up to 21.5 times faster. The K40c GPU was on average 26.0 times faster than the Intel Xeon CPU and the Test9 test image was processed up to 33.7 times faster. Both GPUs achieved the highest speedups in the case of the critical non-convex subproblem (15). The K40c GPU was at its best 76.0 times faster than the Intel Xeon CPU at solving the subproblem.

4.3 Discussion

A significant portion of the total execution time still goes to solving the critical non-convex subproblem (15) but the gap between it and the linear vector-valued subproblems (19) has narrowed considerably. However, even if we managed to overcome the potential global memory bottleneck associated with the linear vector-valued subproblems (19), the critical non-convex subproblem (15) would still dominate the total execution time in such a degree that it probably would not be of a significant improvement. Finally, the speedups obtained with the Mercado test images show that GPU's computational resources can be utilized best when the image size is relatively large.

Although the highest speedups were obtained in the case of the critical non-convex subproblem (15), the K40c GPU did not perform quite as well as expected. One culprit might be the Newton-bisection hybrid method which was used to solve the subproblem. For example, in the case of the Lena ($\sigma = 0.1$) input image, the Newton's method had an average success rate of

Figure 4: From top to bottom: the original images, the noisy input images ($\sigma = 0.1$), and the obtained output images. From left to right: Test9, Lena, Boat, and Mercado512[3].

Image	r_0	Iter.	Whole	Sub. #1	Sub. #2	Sub. #3	Sub. #4	Misc.
Test9, $\sigma = 0.025$	0.005	103	0.6170	0.4489	0.0985	0.0031	0.0412	0.0206
Lena, $\sigma = 0.025$	0.002	77	0.7463	0.5676	0.1097	0.0031	0.0405	0.0206
Boat, $\sigma = 0.025$	0.001	75	0.7967	0.6154	0.1115	0.0031	0.0412	0.0206
Mercado256, $\sigma = 0.025$	0.002	125	0.1719	0.1335	0.0233	0.0008	0.0081	0.0050
Mercado512, $\sigma = 0.025$	0.002	143	0.6813	0.5147	0.0977	0.0031	0.0406	0.0206
Mercado1024, $\sigma = 0.025$	0.002	172	2.6356	1.9665	0.3869	0.0125	0.1670	0.0847
Mercado2048, $\sigma = 0.025$	0.002	169	10.691	7.7660	1.6705	0.0499	0.7866	0.3460
Test9, $\sigma = 0.1$	0.015	163	0.5985	0.4348	0.0942	0.0031	0.0412	0.0206
Lena, $\sigma = 0.1$	0.005	158	0.6881	0.5189	0.0997	0.0031	0.0412	0.0206
Boat, $\sigma = 0.1$	0.005	163	0.6889	0.5181	0.1013	0.0031	0.0412	0.0206
Mercado256, $\sigma = 0.1$	0.005	213	0.1692	0.1309	0.0232	0.0008	0.0082	0.0050
Mercado512, $\sigma = 0.1$	0.005	205	0.6778	0.5104	0.0979	0.0031	0.0412	0.0206
Mercado1024, $\sigma = 0.1$	0.005	208	2.6719	1.9906	0.3958	0.0125	0.1694	0.0857
Mercado2048, $\sigma = 0.1$	0.005	205	10.661	7.7299	1.6658	0.0500	0.7982	0.3458

Table 1: Parameter values, iteration counts, and average per iteration execution times (in seconds) for the Intel Xeon CPU.

99.40%. This is perfectly fine for the CPU since the cost of processing the remaining triangles using the bisection search algorithm is neglectable. However, on the basis of the same data, there is on average 16.59% probability that an individual warp contains a work-item that has to process a triangle using the bisection search algorithm. This has a significant impact on the performance since the cost of processing a single triangle in this way is the same as processing similarly all the 32 triangles as the longest execution path of work-items within the warp determines the cost of completing the computational task assigned to this warp.

The above does not, however, explain why the considerably more powerful K40c GPU did not outperform the GTX580 GPU in such a large extent as would have been expected. The results could be partly explained by the fact that, based on our measurements, the K40c GPU is only 2-3 times faster than the GTX580 GPU at performing special operations such as computing reciprocals and square roots. The Newton-Raphson division algorithm improved performance less than 10%. In addition, we noticed that the K40c GPU was unusually sensitive to how the work group size was chosen. The critical non-convex subproblem (15) required us to set

[3] The Mercado test image is based on the works of Diego Delso and licensed under Wikimedia Commons license CC-BY-SA 3.0 (http://creativecommons.org/licenses/by-sa/3.0/legalcode).

Image	Whole		Sub. #1		Sub. #2		Sub. #3		Sub. #4		Misc.	
Test9, $\sigma = 0.025$	0.0287	**21.5**	0.0147	30.5	0.0085	11.6	0.0002	17.7	0.0041	10.0	0.0012	17.3
Lena, $\sigma = 0.025$	0.0507	**14.7**	0.0357	15.9	0.0095	11.5	0.0002	17.7	0.0041	9.9	0.0012	17.3
Boat, $\sigma = 0.025$	0.0654	**12.2**	0.0501	12.3	0.0097	11.5	0.0002	17.7	0.0041	10.0	0.0012	17.4
Mercado256, $\sigma = 0.025$	0.0136	**12.6**	0.0095	14.0	0.0027	8.6	0.0001	14.1	0.0010	8.5	0.0003	17.1
Mercado512, $\sigma = 0.025$	0.0472	**14.4**	0.0335	15.4	0.0082	11.8	0.0002	17.9	0.0041	10.0	0.0012	17.3
Mercado1024, $\sigma = 0.025$	0.1681	**15.7**	0.1124	17.5	0.0296	13.1	0.0006	19.6	0.0208	8.0	0.0046	18.5
Mercado2048, $\sigma = 0.025$	0.6457	**16.6**	0.4095	19.0	0.1178	14.2	0.0025	20.0	0.0975	8.1	0.0182	19.0
Test9, $\sigma = 0.1$	0.0278	**21.5**	0.0142	30.6	0.0080	11.7	0.0002	17.8	0.0042	9.9	0.0012	17.2
Lena, $\sigma = 0.1$	0.0454	**15.1**	0.0312	16.6	0.0087	11.4	0.0002	17.8	0.0041	10.0	0.0012	17.3
Boat, $\sigma = 0.1$	0.0452	**15.2**	0.0310	16.7	0.0087	11.6	0.0002	17.8	0.0041	10.0	0.0012	17.3
Mercado256, $\sigma = 0.1$	0.0133	**12.7**	0.0093	14.1	0.0027	8.5	0.0001	14.2	0.0009	8.7	0.0003	17.1
Mercado512, $\sigma = 0.1$	0.0464	**14.6**	0.0325	15.7	0.0084	11.7	0.0002	17.9	0.0041	10.1	0.0012	17.3
Mercado1024, $\sigma = 0.1$	0.1726	**15.5**	0.1167	17.1	0.0303	13.1	0.0006	19.6	0.0203	8.3	0.0046	18.7
Mercado2048, $\sigma = 0.1$	0.6440	**16.6**	0.4075	19.0	0.1178	14.1	0.0025	20.0	0.0977	8.2	0.0182	19.0
Average speedup		**15.6**		18.2		11.8		17.8		9.3		17.7

Table 2: Average per iteration execution times (in seconds) and obtained speedups for the Nvidia GeForce GTX580 GPU.

Image	Whole		Sub. #1		Sub. #2		Sub. #3		Sub. #4		Misc.	
Test9, $\sigma = 0.025$	0.0183	**33.7**	0.0059	76.0	0.0078	12.7	0.0001	29.7	0.0036	11.3	0.0009	23.3
Lena, $\sigma = 0.025$	0.0297	**25.1**	0.0163	34.8	0.0087	12.5	0.0001	29.5	0.0036	11.2	0.0009	23.4
Boat, $\sigma = 0.025$	0.0361	**22.1**	0.0225	27.3	0.0089	12.6	0.0001	29.5	0.0036	11.3	0.0009	23.4
Mercado256, $\sigma = 0.025$	0.0091	**18.8**	0.0045	29.4	0.0031	7.6	0.0000	16.6	0.0012	7.0	0.0003	17.1
Mercado512, $\sigma = 0.025$	0.0277	**24.6**	0.0153	33.6	0.0077	12.6	0.0001	29.9	0.0037	11.1	0.0009	23.2
Mercado1024, $\sigma = 0.025$	0.0970	**27.2**	0.0509	38.6	0.0262	14.8	0.0003	37.3	0.0163	10.2	0.0033	26.0
Mercado2048, $\sigma = 0.025$	0.3736	**28.6**	0.1841	42.2	0.1033	16.2	0.0012	40.6	0.0721	10.9	0.0127	27.2
Test9, $\sigma = 0.1$	0.0178	**33.6**	0.0057	75.7	0.0074	12.7	0.0001	29.5	0.0037	11.2	0.0009	23.3
Lena, $\sigma = 0.1$	0.0269	**25.5**	0.0143	36.3	0.0080	12.5	0.0001	29.6	0.0036	11.4	0.0009	23.4
Boat, $\sigma = 0.1$	0.0272	**25.4**	0.0144	35.9	0.0080	12.6	0.0001	29.7	0.0037	11.2	0.0009	23.2
Mercado256, $\sigma = 0.1$	0.0088	**19.1**	0.0045	29.3	0.0031	7.6	0.0000	16.0	0.0010	8.4	0.0003	18.2
Mercado512, $\sigma = 0.1$	0.0274	**24.8**	0.0149	34.2	0.0077	12.6	0.0001	29.8	0.0037	11.2	0.0009	23.2
Mercado1024, $\sigma = 0.1$	0.0996	**26.8**	0.0530	37.5	0.0267	14.8	0.0003	37.5	0.0162	10.4	0.0032	26.4
Mercado2048, $\sigma = 0.1$	0.3721	**28.7**	0.1826	42.3	0.1033	16.1	0.0012	40.7	0.0721	11.1	0.0127	27.2
Average speedup		**26.0**		40.9		12.7		30.4		10.6		23.5

Table 3: Average per iteration execution times (in seconds) and obtained speedups for the Nvidia Tesla K40c GPU.

the work group size as low as 64 work-items. In turn, the GTX580 GPU performed just fine when the work group size was set as high as 512 work-items. This suggest that Nvidia's OpenCL compiler might have problems with resource management. In general, the compiler seems to generate less optimal code for the K40c GPU in many situations. It also appears that the K40c GPU does not perform well in situations where the solution of a subproblem requires multiple kernel launches.

5 CONCLUSIONS

This paper presented a GPU implementation of an augmented Lagrangian based L^1-mean curvature image denoising algorithm and numerical results obtained while comparing the GPU implementation against a single-threaded CPU implementation. Up to 33-fold speedups were obtained, the average speedup being 26-fold. The pointwise handled non-convex subproblem predictably benefited most from the GPU-acceleration. The numerical results indicate that GPUs provide demonstrable benefits in the context of the higher-order variational-based image denoising algorithms and alternating direction type augmented Lagrangian methods.

6 ACKNOWLEDGMENTS

The authors thank anonymous reviewers for their valuable feedback. The presentation of our paper was significantly improved thanks to their comments and suggestions. The research of the first author was supported by the Academy of Finland (grant #252549), the Jyväskylä Doctoral Program in Computing and Mathematical Sciences (COMAS), and the Foundation of Nokia Corporation.

7 REFERENCES

[Ame10] Ament, M., Knittel, G., Weiskopf, D., and Strasser, W. A parallel preconditioned conjugate gradient solver for the Poisson problem on a multi-GPU platform. In: 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), pp. 583–592, IEEE, 2010.

[Bri10] Brito-Loeza, C., Chen, K. Multigrid algorithm for high order denoising. SIAM J. Imaging Sci., 3(3), pp. 363–389, 2010.

[Bol03] Bolz, J., Farmer, I., Grinspun, E., and Schröoder, P. Sparse matrix solvers on the GPU: conjugate gradients and multigrid. ACM Trans. Graph., 22(3), pp. 917–924, 2003.

[Con80] Conte, S. D., and de Boor, C. Elementary numerical analysis: an algorithmic approach. Mcgraw-Hill College, 1980.

[Dav11] Davidson, A., Zhang, Y., and Owens, J. D. An auto-tuned method for solving large tridiagonal systems on the GPU. In: Proceedings of the 25th IEEE International Parallel and Distributed Processing Symposium, pp. 956–965, IEEE, 2011.

[Fly70] Flynn, M. On division by functional iteration. IEEE T. Comput., C19(8), pp. 702–706, 1970.

[For83] Fortin, M., and Glowinski, R. Augmented Lagrangian methods: applications to the numerical solution of boundary value problems. North-Holland, Amsterdam, 1983.

[Glo89] Glowinski, R., and Le Tallec, P.. Augmented Lagrangian and operator-splitting methods in nonlinear mechanics. SIAM, Philadelphia, 1989.

[Göd11] Göddeke, D., and Strzodka, R. Cyclic reduction tridiagonal solvers on GPUs applied to mixed precision multigrid. IEEE T. Parall. Distr., Special Issue: High Performance Computing with Accelerators, 22(1), pp. 22–32, 2011.

[Hel12] Helfenstein, R., and Koko, J. Parallel preconditioned conjugate gradient algorithm on GPU. J. Comput. Appl. Math., 236(15), pp. 3584–3590, 2012.

[Hoc65] Hockney, R. W. A fast direct solution of Poisson's equation using Fourier analysis. J. Assoc. Comput. Mach., 12(1), pp. 95–113, 1965.

[Hoc81] Hockney, R. W., and Jesshope, C. R. Parallel computers: architecture, programming and algorithms. Bristol, UK, 1981.

[Kim11] Kim, H.-S., Wu, S., Chang. L., and Hwu W. W. A scalable tridiagonal solver for GPUs. In: 42nd International Conference on Parallel Processing, pp. 444–453, IEEE Computer Society, Los Alamitos, CA, USA, 2011.

[Kuz85] Kuznetsov, Y. A. Numerical methods in subspaces. Vychislitel'-nye Processy i Sistemy II. 37, pp. 265–350, 1985.

[Kuz96] Kuznetsov, Yu. A., and Rossi, T. Fast direct method for solving algebraic systems with separable symmetric band matrices. East-West J. Numer. Math., 4, pp. 53–68, 1996.

[Lam12] Lamas-Rodriguez, J., Arguello, F., Heras, D.B., and Boo, M. Memory hierarchy optimization for large tridiagonal system solvers on GPU. In: IEEE 10th International Symposium on Parallel and Distributed Processing with Applications (ISPA), pp. 87–94, IEEE Press, Piscataway, NJ, USA, 2012.

[Lys04] Lysaker, M., Osher, S., Tai, X.-C. Noise removal using smoothed normals and surface fitting. IEEE T. Image Process., 13(10), pp. 1345 – 1357, 2004.

[Mum94] Mumford, D. Elastica and computer vision. Algebraic geometry and its applications, pp. 491–506, Springer-Verlag, New York, 1994.

[Myl13] Myllykoski, M., Rossi, T., and Toivanen, J. Fast Poisson solvers for graphics processing units. In: Applied Parallel and Scientific Computing, Manninen, P., and Öster, P. (eds.), Lecture Notes in Computer Science, 7782, pp. 265–279, 2013.

[Myl15] Myllykoski, M., Glowinski, R., Kärkkäinen, T., and Rossi, T. A new augmented Lagrangian approach for L^1-mean curvature image denoising. SIAM J. Imaging Sci., 8(1), pp. 95–125, 2015.

[Par92] Parker, A., and Hamblen, J.O. Optimal value for the Newton-Raphson division algorithm. Inform. Process. Lett., 42(3), pp. 141–144, 1992.

[Ros99] Rossi, T., and Toivanen T. A parallel fast direct solver for block tridiagonal systems with separable matrices of arbitrary dimension. SIAM J. Sci. Comput. 20(5), pp. 1778–1796, 1999.

[Rud92] Rudin, L., Osher, S., and Fatemi, E. Nonlinear total variation based noise removal algorithms. Phys. D, 60(1–4), pp. 259–268, 1992.

[Sun14] Sun, L., and Chen, K. A new iterative algorithm for mean curvature-based variational image denoising. BIT, 54(2), pp. 523–553, 2014.

[Vas84] Vassilevski, P. Fast algorithm for solving a linear algebraic problem with separable variables. C.R. Acad. Bulgare Sci., 37, pp. 305–308, 1984.

[Val85] Vassilevski, P. Fast algorithm for solving discrete Poisson equation in a rectangle. C.R. Acad. Bulgare Sci., 38, pp. 1311–1314, 1985.

[Yan14] Yangab, F., Chenc, K., Yub, B., Fang, D. A relaxed fixed point method for a mean curvature-based denoising model. Optim. Method Softw, 29(2), pp. 274 – 285, 2014.

[Zha10] Zhang, Y. and Cohen, J., and Owens, J. D. Fast tridiagonal solvers on the GPU. In: Proceedings of the 15th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming PPoPP 10, pp. 127–136, ACM Press, New York, NY, USA, 2010.

[Zhu12] Zhu, W., and Chan, T. Image denoising using mean curvature of image surface. SIAM J. Imaging Sci., 5(1), pp. 1–32, 2012.

[Zhu13] Zhu, W., Tai, X.-C., and Chan, T. Augmented Lagrangian method for a mean curvature based image denoising model. Inverse Probl., 7(4), pp. 1409–1432, 2013.

Simultaneous Absorption and Environment Light Reconstruction in Optical Tomography Problem

Afanasyev V.
Keldysh Institite of Applied Math RAS
4, Miusskaya sq.
Russia, 125047, Moscow
vafanasjev@graphics.cs.msu.ru

Ignatenko A.
CMC MSU
Leninskie Gory, MSU
Russia, 119991, Moscow
ignatenko@graphics.cs.msu.ru

Voloboy A.
Keldysh Institite of Applied Math RAS
4, Miusskaya sq.
Russia, 125047, Moscow
voloboy@gin.keldysh.ru

ABSTRACT

Classic tomography algorithms applied in optical tomography require the light source pre-calibration and do not allow refining the light map in tomography algorithm. This article shows an approach to environment light reconstruction during the ART algorithm execution. It makes the optical tomography scanning process more fast and simple, allowing to exclude the light calibration stage.

Keywords

ART, tomography, lighting reconstruction

1. INTRODUCTION

Tomography concept

Computed tomography is a class of problems of object internal structure reconstruction using a set of its projections. The closest application area of computed tomography methods is X-ray tomography.

The X-ray tomography device consists of emitter, detector and a place for observable object between them. Emitter irradiates the X-rays with fixed intensity, they are absorbed inside the object and the detector registers residual ray intensity. During the object rotation on some trajectory a set of projections is created and using them the internal object structure is reconstructed.

Different configurations of ray beam are possible: it can be flat or volumetric, parallel or cone. Depending on this, flat or volumetric tomographic reconstruction is used.

The radiation absorption inside the material obeys Beer's law:

$$I = I_0 e^{-\int_a^b k(x)dx} \quad (1)$$

where I is residual intensity received by detector, I_0 is initial emitter intensity irradiated in this direction, $k(x)$ is distribution of absorption index along the ray.

If we take a set of parallel lines perpendicular to flat detector and integrate the absorption index along every line, and consider all directions of these lines, we get the Radon transform for examined volume as it is shown on fig. 1.

$$R(P,n) = \int_{-\infty}^{\infty} f(P + \vec{n}t)dt \quad (2)$$

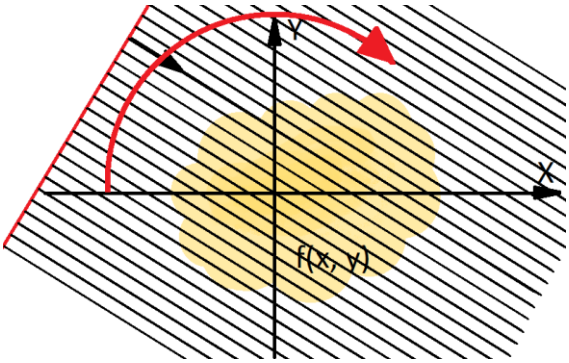


Figure 1. Radon transform

If we put some restrictions on function to reconstruct, the inverse Radon transform exists **[Hel99a]**. But if there is a noise or some other inaccuracy in initial data, it is unstable. Also counting reverse Radon transform is computationally inefficient.

There is a number of approximate tomography algorithms, for example, algorithm based on inverse Fourier transform, convolution and back projection, algebraic reconstruction techniques (ART). ART group of methods are the most flexible, so one of them is used in this study.

ART is based on sequential correction of resulting function stored in voxel map using its projections one-by-one. Every projection value which is an integral of initial function along the corresponding ray affects the resulting function along the same ray according to some law. That makes observable

integral along the ray closer to the real measured value of this integral every time. Then, the iterative process stops on some trigger.

Optical tomography

Optical tomography uses the same principle as X-ray tomography, but there are light rays instead of X-rays. This makes significant difference and adds some physical effects and difficulties in handling them.

- Light is reflected and refracted when it meets transparent object
- Complete absorption of light is a usual case
- Refraction changes the light direction, and usually for some particular area inside transparent object only part of all directions are available for observation
- Defocusing on convex refracting surfaces
- Rays that were parallel in the air, after refraction cross the target volume from multiple directions

Since here, we will consider only objects with flat surfaces to avoid defocusing problem which is not the point of this study. Also in order to simplify the equations, working with single wavelength will be considered because processing polychromatic spectrum data does not refer directly to the subject of this article.

The optical tomography scanner (fig. 2) consists of areal diffuse light source, camera and axis to place and rotate the observable object. The installation is covered with opaque housing to prevent object illumination from outside.

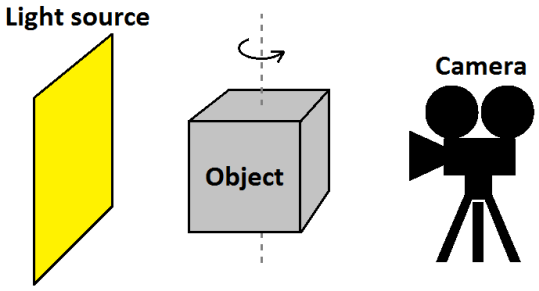


Figure 2. Optical tomography scanner

In this article the approximate ART (Algebraic Reconstruction Technique) algorithm will be used for volumetric absorption reconstruction. Unlike classic ART **[Gor70a]** this algorithm uses approximate correction of absorption along the ray inside target volume. It is based on difference between observable pixel brightness and its expected value, which was calculated using current absorption distribution inside the target volume, without direct calculation of observable integral absorption value.

2. ENVIRONMENT LIGHT PROBLEM. EXISTING APPROACHES

Tomographic reconstruction requires exact knowing of integral absorption along all rays crossing the volume of interest. In ideal case, light should leave the light source, then, after two refractions and absorption inside the object it should be captured by camera. Also light intensity distribution on the light source should be known. Then using Beer's law and Fresnel equations the residual intensity can be evaluated:

$$I = I_0 T^2 e^{-K} \qquad (3)$$

Where I is light source intensity in the ray hit point, I_0 is pixel value measured with camera, T is Fresnel transmittance coefficient for particular angle of incidence, $K = -\int_a^b k(x)dx$ is integral absorption along the ray, this value will be used in tomography.

$$K = -\ln\frac{I}{I_0 T^2} \qquad (4)$$

The light source can be more complex than it is shown on the scheme. For example, it can consist of primary light source like LED and reflective diffusor. The scanner has several internal parts and though the installation is closed from external light, camera can receive some light that was reflected multiple times inside the scanner. Observable object itself can reflect light back to diffusor that increases its brightness. Finally, this excess light is reflected from observable object to camera and it contributes into the pixel value. So, in order to find a precise value K of absorption inside the object, all illumination around the object and its reflections should be considered.

Simple light acquisition

The simplest method to get some approximation of real light is to make a photo of background behind observable object, when this object is removed. Generally, it's impossible to make one photo for the scanner and to use it for every scanned object. Scanning every new object may require its own light settings that can be tuned only after taking some photos of observable object. So, there are 2 ways to carry out scanning and tomography. Note that tomography requires having data about lighting before it started.

1. Separate sequential scanning and tomography:
   - Placing object to scanner
   - Tuning light using sample photos of object
   - Scanning object (taking photos of it, rotating 360° around vertical axis)
   - Removing object
   - Taking photo of background

- Carrying out tomography

2. Simultaneous scanning and tomography:
   - Placing object to scanner
   - Tuning light using sample photos of object
   - Removing object
   - Taking photo of background
   - Placing object again
   - Scanning and tomography (every taken photo is immediately applied in algorithm, scanner and computer work simultaneously)

As we can see, in first case there is no parallelism, in second case taking photo of background requires extra placing and removing object in case of tuning light. Every variant takes some additional time or human actions.

Another problem is the size of light source. Normally, light source is bigger than camera frame. Otherwise, if the light source is smaller or of the same size, there will be dark areas on photos: on the object or around it and this will decrease the amount of information for tomography. Taking photo of the background gives us only part of light source, though, most part of observable light comes from this area.

Simple calibration

Possible solution to the problem is light source calibration using parametric model **[Afa14a]**. Light distribution on the light source inside and outside the frame is calculated using a set of parameters. Then, these parameters are optimized by comparing rendered scene with real photos. The geometric model of light source, some other data like LED radiation pattern can help to reduce the number of parameters. This approach can be used without removing object from scanner, the only requirement is to use frame(s) with some piece of background area or knowing average absorption of the object.

A disadvantage of parametric light model is inexact matching of computed light intensity and real background in visible light source area. Solution to this problem, which is used in practice, is combining last two methods: we use real photo in visible area and tuned parametric model in invisible area. It takes additional time, but gives better result than any of two "pure" methods.

If for some reason the parametric model is too complex to be tuned fine, or side reflections (fig. 3) have significant brightness, the environment light reconstruction can help to make more accurate light model.

3. SEPARATE LIGHT RECONSTRUCTION

Existing approaches

There are some studies that solved the problem of environment light reconstruction using artificial objects like reflective balls **[Hey05a]** or more generally, using reflections from any scene objects **[Gib01a]**. Usually more general algorithms represent light as panorama or a set of point light sources. They use an iterative technique to reconstruct lighting. A similar technique is used in the following algorithm which also takes light transmission into account, and it is a base for more complex algorithm discussed later.

Implemented algorithm

An algorithm using both transmitted and reflected light to reconstruct light panorama was implemented.

The observable object is required to be reflective and can be also transparent. The environment can be defined as 3D scene model or as simple spherical panorama. In the tests of this study a sphere with finite radius was used, because the available real scanner has orthographic camera and does not have any precise draft of internal geometry. For example, light can be positioned manually.

The algorithm takes the following data:

- A set of greyscale photos of observable object
- Camera calibration for all photos
- 3D model of environment (not necessary)

The algorithm output is a panorama light map which can be applied to initial scene as a texture.

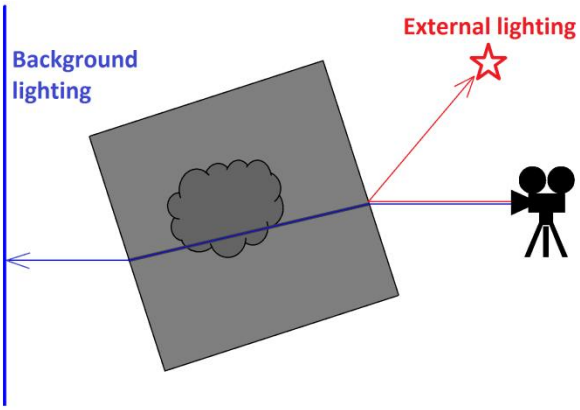


Figure 3. Rays path from camera to environment

It works iteratively, using a sequence of given images in some defined order. For example, it can be a trivial order with continuous camera movement. The initial panorama is black. For each frame there are the following steps:

1. **Rendering step.** A ray is traced from camera into scene as it usually is done in ray-tracing. It hits some surfaces, reflects, transmits, splits into

multiple rays and finally every of these rays cross the environment in some point corresponding to a panorama point. Radiance from all rays is gathered (summed, multiplied by Fresnel coefficient and transmittance coefficient in case of absorption) and assigned to current pixel brightness I.

2. **Counting correction.** Existing brightness value in pixel is compared with observable brightness I_0 taken from photo. A difference $\Delta I = I - I_0$ is counted. Then, a correction value C is calculated based on difference. For example, $C = t\Delta I$, where $0 < t \leq 1$.
3. **Correction step.** The second time a ray is traced, having the same route. Each ray has its own correction value and there are the following rules for correction propagation:
   - If the ray hits environment, its correction is added to panorama value in this point
   - After hitting a surface of transparent object the ray splits into 2 rays. Corrections of new rays are proportional to their impact (to Fresnel coefficients) and sum of them is correction of initial ray.
   - If total internal reflection occurs, the ray correction is preserved.

It is important that during correction step rays repeat the trajectory of the rays on rendering step and hit the same panorama points. It can be achieved by just saving hit points and ray impacts on the first step without tracing one more time, if architecture allows this. Also hitting one panorama pixel by 2 rays on one frame should be considered: double correction will lead to wrong result.

Panorama light reconstruction on scanner with areal background light showed interesting results. The object to reconstruct can be an immersion glass cube with a gemstone inside it (typical object for tomography reconstruction: in this case the inclusion models inside the stone are the main target). On the figure 4 you can see an example of analogic object: a cube of epoxide glue with some wires inside it.

Figure 4. Object example

On the first iterations algorithm makes some phantom light source behind the camera and the real light source behind. That should be expected, because the route from camera to this panorama area contains only one reflection from front cube face.

The figures 5 and 6 show sphere maps of light intensity. Camera looks to the point in center of real light labeled on figure 5. If we move half an image leftwards, we will get into the point behind camera. Top and bottom of picture are sphere poles.

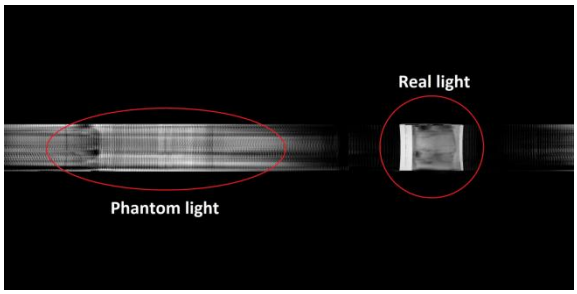


Figure 5. First iterations of light reconstruction.

But then the process converges to correct result, eliminating brightness on the place of phantom light.

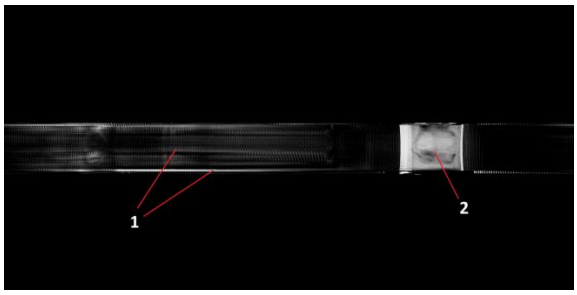


Figure 6. Final result of reconstruction

The final map is not ideal, because not all real effects were considered in the model. We can see different types of artefacts:

1. Some residual light on the place of phantom light source behind the camera. It remains because the used model of cube has inaccuracies and its edges produce big difference between photos and renders every time. Bright stripe behind camera has the same explanation.
2. A grey image on the light source that resembles original object: the semi-transparent cube and some dirt inside it. The absorption wasn't included into the model specially and some effect of really present absorption should have appeared: proportional decreasing of light brightness behind the cube is such effect.

It should be mentioned that the process has no convergence in a strict sense due to model inconsistency that is result of its incompleteness. The light map cannot represent absorption effect

correctly, so, after a number of iterations light map will just make small but continuous oscillations.

Comparison of the light map reconstruction algorithm to parametric model **[Afa14a]** shows that the new algorithm gives better result in visible areas of light source (expectedly). It has almost the same result as the photo in visible light areas (difference is less than 1 grade of 255). But in the areas behind the object parametric model are still much better due to absorption map inside the object not being taken into account.

The advantage is that the new algorithm reconstructed invisible parts of diffusor using reflections from side cube faces. The data that was previously only extrapolation with parametric model, now is image-based. The new algorithm generally does not need the model of light source, its projection to sphere is reconstructed automatically.

4. SIMULTANEOUS LIGHT AND ABSORPTION RECONSTRUCTION

The next step is combining light reconstruction and tomography in a single process.

Algorithm on the frame and pixel level is the same as described in light reconstruction. The difference is addition of absorption impact. It should be taken in account in rendering step and in correction step.

Rendering step is obvious: brightness is calculated and summed as usual, but also the integral absorption inside target volume is counted and final brightness of the ray is multiplied by this value.

Correction step requires separation of absorption and brightness correction. For the ray which crossed target volume and leaves the medium, we should choose, how to share correction between these two opposite effects: absorption that decreases intensity of light brought with the ray and brightness of panorama pixels situated somewhere farther along this ray. The problem and the main difference from separation between transmitted and reflected rays is multiplication of brightness and transmittance coefficient, as we can see in equation 1.

$$I = I_0 A \qquad (5)$$

If we want to make some correction ΔI we cannot just separate it like the following:

$$(I + \Delta I) = (I_0 + \Delta I_0)(A + \Delta A) \qquad (6)$$

This problem is not solved yet, correction separation here is regulated with manually set coefficients for now and is inaccurate. The source of this issue is differential nature of correction. Carrying the desired value of brightness with all rays seems solve it, because in correction formula (6) we get rid of multiplication two sums. But this replacement will cause problems with storing brightness corrections in panorama.

5. IMPLEMENTATION AND RESULTS

Figure 7 shows an example of one horizontal layer of absorption map built from photos of glass cube. Bright areas represent dirt inside the glass.

Figure 7. Absorption voxel map slice

Figure 8 shows rendering of reconstructed absorption map in reconstructed lighting compared to real photo.

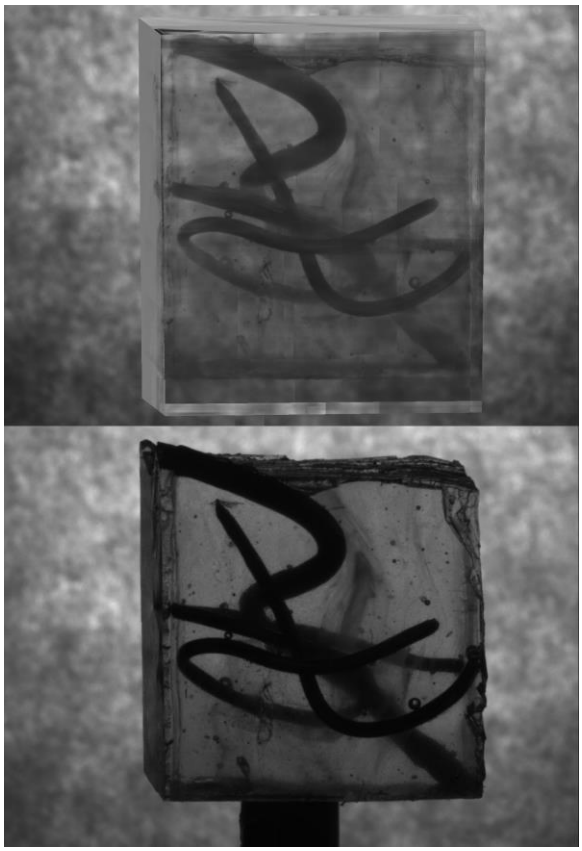


Figure 8. Rendering (top) and photo (bottom)

 ISBN 978-80-86943-65-7

The algorithm is implemented on the base of ray tracing algorithm. Tomography procedure is built in ray tracing mechanism. NVidia OptiX GPU ray tracing engine was used **[Par10a]**.

Absorption index is stored as voxel map of float32 values. Its size is 512^3 voxels. Lighting panorama has resolution 4096×2048 and float32 type is used also. These arrays are stored in GPU memory.

The algorithm reaches acceptable absorption map quality after 1000 iterations (the same as algorithm using pre-calibrated light). The algorithm execution takes less than 10 minutes on GTX 980 for a cube occupying nearly all the visible area. The test sample with wires inside epoxide glue took 2.5 minutes on GTX Titan X. However, the light map quality is not enough after this due to the effect of phantom light described before, and about 10000 iterations are necessary to get acceptable lighting map.

6. CONCLUSION AND DISCUSSION

So, the algorithm was created which allows to build an absorption map without having previously calibrated light. It allows to exclude the physical manipulations with the scanner and observable object before scanning.

This study contains several unresolved problems: phantom light remaining in panorama, inexact separation of correction between brightness and absorption, slow convergence of light reconstruction. Though these issues do not disturb fast building of absorption map, which was the main purpose, solving them will help to make algorithm more accurate and fast.

Also this study considers only transparent materials. Although, real objects may contain some other effects that influence light travelling inside the material. Scattering, more complex absorption, reflections from internal structures among them. These effects can be taken into account by changing the physical model and light transport model: for example, a modified Beer's law can be used. The further research will be directed to reconstruction of areas which have complex interaction with light, like cracks inside gemstones. Currently such objects are not reconstructed with acceptable quality.

7. REFERENCES

[Hel99a] Helgason S. Radon Transform Second Edition. Cambridge, pp.15-20, 1999.

[Gor70a] Gordon R., Bender R., Herman GT. Algebraic reconstruction techniques (ART) for three-dimensional electron microscopy and x-ray photography. Journal of Theoretical Biology.29. No.3, pp. 471–81. 1970.

[Afa14a] Afanasiev V., Ignatenko A., Tisevich I. Secondary light source calibration using photos. Information techno;ogies in design and production. No. 2, pp. 41-45. 2014.

[Hey05a] Heymann, Sebastian, et al. Illumination reconstruction from real-time video for interactive augmented reality. Proc. of International Workshop on Image Analysis for Multimedia Interactive Services. 2005.

[Gib01a] Gibson S., Howard T.J., Hubbold R.J. Flexible Image-Based Photometric Reconstruction using Virtual Light Sources. Eurographics 20, no.3, Manchester. 2001.

[Par10a] Parker S., Bigler J., Dietrich A. OptiX: a general purpose ray tracing engine. ACM SIGGRAPH 2010 papers, Article No. 66. 2010.

A Simple and Efficient Feature Descriptor for Fast Matching

Anders Hast
Department of
Information Technology
Uppsala University,
Uppsala, Sweden
anders.hast@it.uu.se

Victoria A. Sablina
Department of
Electronic Computers
RSREU,
Ryazan, Russia
sablina.v.a@evm.rsreu.ru

Gustaf Kylberg
Vironova AB
Stockholm
Sweden
gustaf.kylberg@vironova.com

Ida-Maria Sintorn
Vironova AB
Stockholm, Sweden &
Department of
Information Technology
Uppsala University,
Uppsala, Sweden
ida.sintorn@it.uu.se

ABSTRACT

A very simple but efficient feature descriptor is proposed for image matching/registration applications where invariance is not important. The descriptor length is only three times the height of the local region in which the descriptor is calculated, and experiments were conducted to compare it to the SURF descriptor. In addition, it is shown, how the sampling can be modified in order to obtain a rotation invariant descriptor, while still keeping it simple and efficient. Examples from stitching in microscopy and stereo processing of pairs of photographs are given to prove the concept.

Keywords

Feature Descriptor, Nearest Neighbour Matching, Rotation Invariance, SURF, Interest Point Detector.

1 INTRODUCTION

Feature detectors are commonly used in applications such as image registration, image mosaicing [BL07, Sze06], object detection and classification. Other areas where computationally fast feature detectors are used are tracking [ST94], motion estimation, camera calibration, stereo vision and image superimposition [SNNL13, SNNL14].

Many modern computer vision applications require computations in real time, e.g., Simultaneous Localisation and Mapping (SLAM) [DWB06, BDW06] and therefore one of the most important properties for time-critical applications is efficiency, i.e. low computational cost. By increasing the level of invariance, some feature properties are affected such as accuracy and efficiency. It is important to use an adequate detector, no more and no less, in order to not loose these properties.

1.1 Interest Points and Descriptors

In many of the aforementioned applications, features are extracted by first finding interest points, a local neighbourhood is extracted and a descriptor is formed from this neighbourhood. The descriptors can be distinguished, not only by the type of feature, but also by the level of invariance to rotation, translation, scale and perspective distortions. SIFT [Low04, BL07] and SURF [BETVG08] (which is less complex than SIFT and therefore also faster) are scale and rotation invariant detectors. The descriptor for SIFT of length 128, is formed from a histogram of local gradients. The descriptor for SURF is computed by first finding the dominant orientation. It uses wavelet responses, which are weighted with a Gaussian and the dominant orientation is estimated as the sum of all responses within a sliding window. The descriptor contains the wavelet responses in 4×4 subregions together with the sum of the absolute values of the responses. The final vector has length 64 and is therefore both shorter than SIFT and faster when performing the matching. The sampling area is a multiple of the scale factor.

1.2 Feature Matching

In the next step, a process called feature matching is required to establish the correspondences between the features in the images using the extracted feature descriptors. The particular matching method can be chosen depending on the type of the extracted descriptors [ML12]. Usually, a distance measure is used, such as Sum of Squared Distances (SSD), normalised cross correlation [NT13] or the Chi-squared distance [Has14], to determine how similar feature vectors are. The pair of features having the smallest distance is consequently considered to be nearest neighbours. An exhaustive search can be applied if there are few correspondences or when the

feature vectors themselves are short. Otherwise, some kind of partitioning method, such as kd-trees [FBF77], k-means clustering [FN75], or combinations of both [ML09] can be used to speed up this part of the process.

1.3 Outlier Removal and Computing the Transformation

Finally, outliers are removed and the geometric correspondence between the images is established using some version of the RANdom SAmple Consensus (RANSAC) algorithm [FB81, HZ04]. RANSAC starts by selecting the minimal number of points required to determine the model parameters, i.e. the homography [HZ03, BL07], which is the projective transformation between the images. Using this transformation, the number of inliers that falls below a certain tolerance ϵ, are counted, i.e. points being close enough to its corresponding match are regarded as inliers. When the probability of finding a better model becomes lower than some threshold, the algorithm terminates, otherwise it starts all over selecting a new point set. Generally, N iterations are needed in order to find an outlier free set with the probability p (often set to 99% or more) as:

$$N = \frac{\log(1-p)}{\log(1-\gamma^{\,s})}, \tag{1}$$

where γ is the inlier ratio, i.e. number of inliers divided by number of points in the cluster and s is the number of samples drawn each time. The algorithm starts all over and samples the set once again if N is larger than the number of iterations of the main loop. An alternative termination criterion is used for the OptimalRANSAC [HNM13], which terminates when two identical sets are being found after applying local optimisation [CMK03], re-estimation and pruning.

1.4 Contributions

To summarise there are four steps in the process of matching/registering images from interest point detection to computing the transformation between a pair of images:

1. Detection of interest points.
2. Extraction of feature descriptors for those interest points.
3. Matching of the descriptors to find tentative correspondences.
4. Removing outliers with RANSAC or similar, and obtaining the transformation in the same process.

In this paper we focus on the second point. We propose a very efficient feature descriptor for fast matching that can be used in applications where speed is crucial and scale invariance is not important. Two such cases are examined herein: image stitching in microscopy, and stereo pair extraction from two aligned cameras. The descriptor can easily be extended to be rotation invariant. We compare and evaluate our proposed descriptors to the SURF descriptor [BETVG08], which is a popular and fast descriptor invariant to scale and rotation.

2 RELATED WORK

There are many different interest point detectors proposed in literature such the Shi-Tomasi corner detector [ST94], which is based on the Harris detector [HS88] and FAST [RD06], which is based on SUSAN [SB97], just to mention a few. Some find corners other blobs or edges [Can86]. Several overviews of different types of detectors have been published [TM08, SMB00, ZKM04].

Many different feature descriptors have been developed [MS05, GHT11]. The main requirements of a feature descriptor are low computational cost (efficiency) and high robustness, i.e. invariance to illumination and image transformations like scaling and rotation. Histogram of Oriented Gradients (HoG) [DT05] is a descriptor, which is distribution based just as SURF. However, HoG is not invariant to rotation. To reduce the dimensionality it was proposed to build binary descriptors like the BRISK descriptor [RD06] and FREAK [LCS11, AOV12]. Binary descriptors are faster but less precise. Another approach is to use the frequency domain to construct a feature descriptor [HM13, Has14]. Such descriptors are robust but the use of the Fourier transform leads to increased computational cost.

For both detectors and descriptors, invariance is often required, either to illumination, rotation or scale. However, not all applications require all of these and the importance of these different feature properties depends on the application.

3 A SIMPLE NON INVARIANT FEATURE DESCRIPTOR

The idea proposed in this paper is to construct the descriptor in as simple way as possible for applications where speed is crucial and the extra overhead for handling scale and rotation are not needed. The pixel values in the area around the interest points are sampled in the following way: if the size of the area around the point is $n \times n$ then a descriptor of length $k = 3n$ is obtained by sampling each line of pixels and computing the following three measures: mean μ, $(min - \mu)^2$ and

$(max - \mu)^2$, where min and max are the min and max of the pixels in that row. Several experiments where conducted and it turns out that computing the square of the differences, rather than just the difference, improves the result noticeably, which has also been shown for other matching methods [Bor84]. The experiments reported herein also contain the results using the mean only in order to show that adding these squared difference values makes the descriptor more robust than when just using the mean.

3.1 Data Sets

Two data sets were used for testing and evaluating the performance of the descriptor. The first data set comes from the MiniTEM, which is a benchtop low-voltage transmission electron microscope designed for easy TEM imaging and quantitative analysis of biological as well as inorganic samples. The images are all grayscale in 12 bit and 2048x2048 pixels in size. The first three (A, B and C) are images of tissue section of human kidney. The two last (D and E) are images of mimivirus particles inside of an amoebae. The images in this data set are found to the left in figure 1.

The second data set, found to the left in figure 2, contains stereo images from video sequences of objects of different heights.

3.2 Experiments

The images were first blurred using a Gaussian of size 9 with $\sigma = 1.0$ in order to remove noise. The top 1200 interest points using the *detectSURFFeatures* function in Matlab® were detected. In order to investigate how the number of points used in the matching affect the result, the matching was performed using the top 400, 800 and 1200 points.

The SURF features were extracted using the Matlab function *extractFeatures* using the following parameters: 'Method','SURF'. The proposed descriptors were extracted using a MEX function. The matching was performed using the *matchFeatures* function in Matlab using the SSD. Finally a version of RANSAC was used that is supposed to obtain the optimal set of inliers in each run [HNM13].

The results of matching the image pairs in the first data set are shown in figure 1. The image pairs are depicted to the left and as an illustration of the matching, green '*' indicate inliers and red outliers. For all images the results from using the proposed descriptor is shown, using the top 1200 points and a descriptor length of 63, i.e. a bit shorter than the SURF descriptor. The diagrams show the ratio of inliers compared to the total number of points used, i.e. 400 (blue), 800 (yellow) and 1200 (red), and the inlier ratio (points remaining after matching). The same colour scheme is used for both diagrams, and the circles connected with lines corresponds to the proposed descriptor, while the squares connected with dotted lines are the results of the SURF descriptor. The non-connected triangles are the proposed descriptor using the mean only and the descriptors used are hence just a third as long. The y- axis shows the percentage (i.e. the ratios) and the $x-$ axis the size n of the sampling area, which is always odd, since the key point pixel must lie in the middle of the area.

The results of matching the image pairs in the second data set are shown in figure 2. It should be noted that the RANSAC used for the experiments finds the largest set of inliers, i.e. the optimal set, for *one* model. However, in the images there are sometimes more than one model since objects are placed on different heights. Nevertheless, this problem applies to both the novel descriptor and SURF and hence the comparison is done only for the optimal set for practical reasons.

Normally, SURF gives the same number of inliers since the descriptor is always 64 long and always samples the same neighbourhood size. However, the region of interest (ROI) of the whole image must be set for the *detectSURFFeatures* so that sampling is not done outside the image. As exactly the same features was going to be used for both methods being compared, the number of inliers could vary slightly since the ROI varies depending on the size of the sampling area. In this way the maximum area possible was always sampled instead of setting a fixed ROI for all images. Nevertheless, it can be noted that SURF does better than the novel method when small neighbourhoods are sampled and that the novel method starts to do better than SURF for larger sizes.

When the size of the area sampled is 22, the descriptor will consequently be of size 66 and it will therefore be slightly larger than the SURF descriptor. Hence, if the connected circles are found above the connected squares to the left of 22 in the diagrams, then the proposed method gives a better result for shorter descriptors.

4 ROTATION INVARIANCE

Rotation invariance can be added in a very simple way, by just sampling in circles around the centre point instead of sampling along lines. The idea is shown in figure 3. The mean μ for each circle is stored in the beginning of the vector. If a radius of r around the centre pixel is used for sampling, the number of circles n should be less than r. The $(min - \mu)^2$ and $(max - \mu)^2$ are stored after the

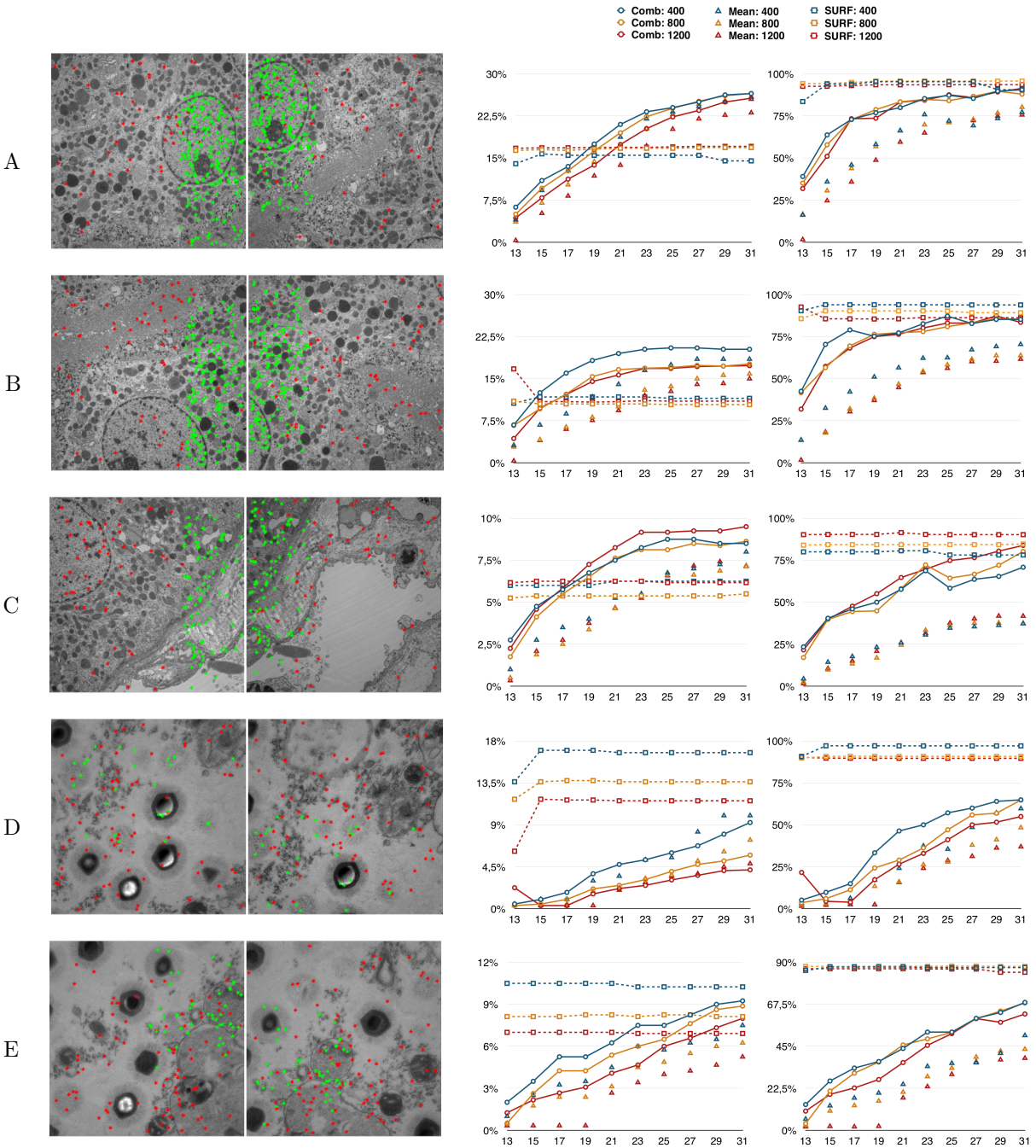


Figure 1: Results from the MiniTEM data set. The images to the left are matched using the proposed method (both the descriptor based on the mean only and also the one using the combination of mean, max and min) and SURF. The matching was performed using the top 400 (blue), 800 (yellow) and 1200 (red) points. The diagrams shows the result and the circles connected with lines corresponds to the proposed descriptor, while the squares connected with dotted lines are the results of the SURF descriptor. The non-connected triangles are the proposed descriptor using the mean only. The diagram to the left shows the ratio of inliers compared to the total number of points used and the diagram to the right shows the inlier ratio after matching and RANSAC. The $y-$ axis shows the percentage and the $x-$ axis the size n of the sampling area.

 ISBN 978-80-86943-65-7

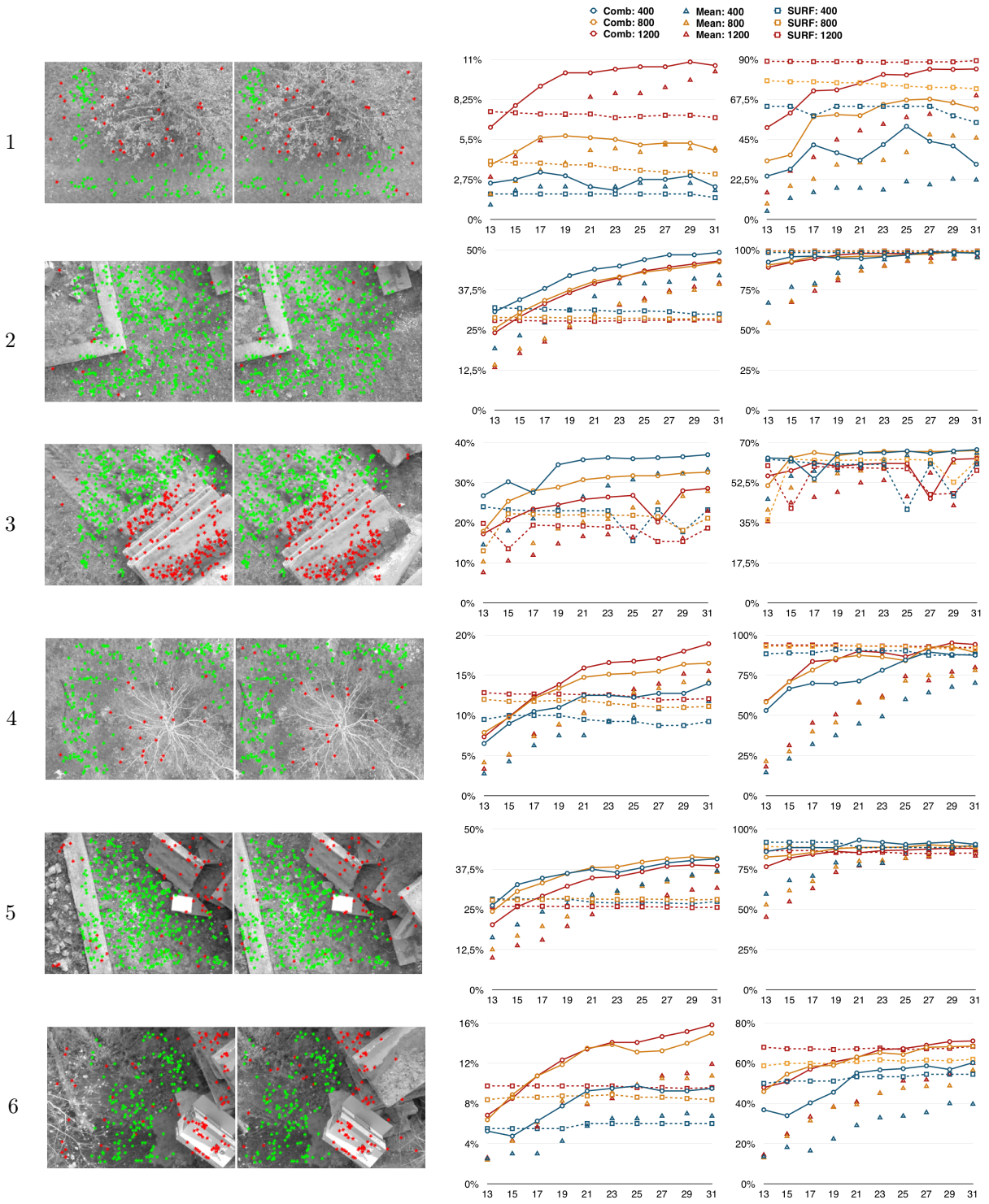


Figure 2: Results from the stereo data set. The images to the left are matched using the proposed method (both the descriptor based on the mean only and also the one using the combination of mean, max and min) and SURF. The matching was performed using the top 400 (blue), 800 (yellow) and 1200 (red) points. The diagrams shows the result and the circles connected with lines corresponds to the proposed descriptor, while the squares connected with dotted lines are the results of the SURF descriptor. The non-connected triangles are the proposed descriptor using the mean only. The diagram to the left shows the ratio of inliers compared to the total number of points used and the diagram to the right shows the inlier ratio after matching and RANSAC. The y- axis shows the percentage and the $x-$ axis the size n of the sampling area.

 ISBN 978-80-86943-65-7

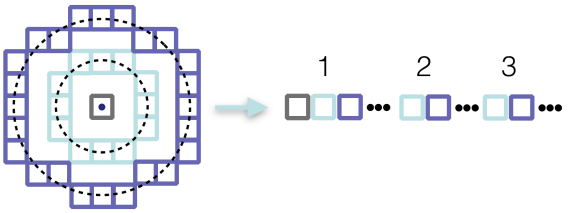


Figure 3: A rotation invariant descriptor is achieved by sampling in circles around the interest point.

mean for each circle. The total length of the descriptor will be $3n - 2$, since the centre pixel has no min or max.

4.1 Results of Experiments

The rotation invariant descriptor was compared to SURF for all images in the second data set (figure 2). The right image was rotated 45° using bicubic interpolation. A radius $r = 14$ was chosen in order to sample a rather small area together with $n = 13$, giving a descriptor length of 37, as indicated in the diagram in figure 4, which is close to half the length of the SURF descriptor (length 64). Still the proposed descriptor does better or almost as good when it comes to finding inliers from the set of points. This time the top 1200 points were used, and it was noted that the results were almost identical for using 400 or 800. The $y-$ axis show the percentage and the $x-$ axis corresponds to the row number in figure 2, so that the image pairs could be easily identified.

One can note that the number of found points are just as good or even better with the new descriptor, but also that adding the $(min - \mu)^2$ and $(max - \mu)^2$ really has a great impact on the result. The green bars in the diagrams correspond to using just the mean giving a length of 13 as indicated. Even if the new detector often finds more inliers compared to the size of the data set, the inlier ratio is still less than with SURF. This is due to the fact that there are more outliers after the matching process for the new descriptor. However, the good news is that also more inliers are found.

5 DISCUSSION

The novel descriptor proposed in this article was tested on two rather different data sets, where scale invariance is not necessary and just brings extra overhead. Instead it is important that the descriptor is fast to compute so that the transformation can be obtained in real-time. For the MiniTEM images it is important since image matching is used in the alignment process, and it can also be used to create a larger digital field of view than the microscope directly can provide. Since video sequences obtained from the stereo camera can be rather long, the time consuming task of stereo matching could be decreased with a short but efficient descriptor. Moreover, if they are fast enough for real-time matching, it would be possible to obtain instant stereo images while filming.

The proposed descriptor is very fast to compute since it contains just the mean of each row in the square area around the interest points together with the squared differences between the mean and the min and max of each row. In order to obtain rotation invariance it is just changed so that the values are computed for circles around the centre point instead of lines. The extra work needed is minimal as sampling in a circular manner can be achieved without the sine or cosine in the inner loop by using the Chebyshev recurrence relation [BF01, BHB04]. The results show, not surprisingly, that the larger the descriptor the better the result. However, even for rather small sizes the proposed descriptor performs better than SURF when it comes to finding more inliers. Nonetheless, it can be noted that SURF sometimes achieves a higher inlier ratio, especially when the novel descriptor is shorter. To some extent, this can be due to the fact that the interest point detector used for both SURF and the novel descriptor is tailored for SURF. It should be noted that here is nothing that prevents from using any other interest point detector together with the novel descriptor, and in fact one should use some approach that is faster, like Harris. However, to make a fair comparison between the novel descriptor and SURF, the same interest point detector was used in order to make sure that the very same number of points were used for obtaining the descriptors.

The rotation invariant descriptor also works very well, even for shorter descriptors. It was noted in the experiments that a vector length of only 37 gave just as good or better results than SURF. This can be a powerful descriptor in cases where rotation invariance is required but not scale invariance. As an example, for panoramic stitching one could rotate the camera when taking pictures and the matching would still be able to handle this. Of course, zooming would not be possible as the detector is not scale invariant.

Finally, it should be mentioned that SURF was used instead of SIFT since the latter is more complex, and therefore slower. There are also other alternatives but SURF was chosen as it has be-

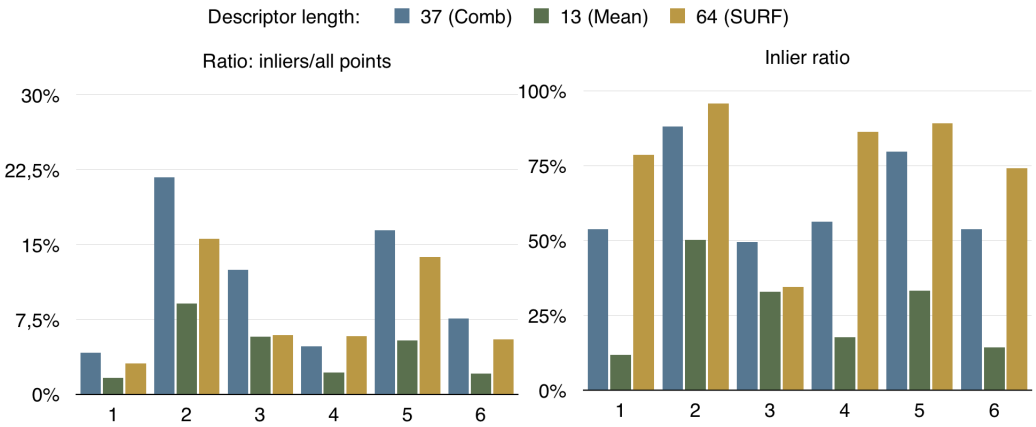


Figure 4: Comparison of image matching where one image is rotated 45°. Methods used are the proposed method (blue), the same but using the mean only (green) and SURF (yellow). The x-axis label corresponds to the image pairs in figure 2.

come very popular and is implemented in both Matlab and OpenCV.

6 CONCLUSION

Fast feature descriptor extraction and matching is crucial for many applications where invariance is less important. Two examples were used, one from microscopy stitching and the other from stereo cameras. A very fast to extract but efficient descriptor can be achieved by simply sampling each row in the area surrounding the interest points and for each line computing the mean and squared differences of the mean and the max and min. Hence, the resulting descriptor will be three times larger than the height of the area being sampled. In comparison with SURF, this novel detector often did better, even for shorter descriptor lengths than 64, which is the length of SURF.

An efficient rotation invariant descriptor was also proposed by simply sampling in a circular manner around the centre points. This can be useful for situations where speed is crucial but where rotations might occur.

7 ACKNOWLEDGEMENT

We would like to thank Kjell Hultenby at Karolinska Institutet, Huddinge, Sweden as well as Michael Laue and Lars Möller at the Robert Koch Institute, Berlin, Germany for providing the samples used in this article for the first data set.

8 REFERENCES

[AOV12] Alahi A., Ortiz R., Vandergheynst P.: Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (June 2012), pp. 510–517.

[BDW06] Bailey T., Durrant-Whyte H.: Simultaneous localization and mapping (slam): Part ii. *Robotics Automation Magazine, IEEE 13*, 3 (Sept 2006), 108–117.

[BETVG08] Bay H., Ess A., Tuytelaars T., Van Gool L.: Speeded-up robust features (surf). *Comput. Vis. Image Underst. 110*, 3 (June 2008), 346–359.

[BF01] Burden R. L., Faires J. D.: *Numerical Analysis Brooks.* Cole, Thomson Learning, 2001.

[BHB04] Barrera T., Hast A., Bengtsson E.: Incremental spherical linear interpolation. In *Sigrad 2004* (2004), pp. 7–10.

[BL07] Brown M., Lowe D. G.: Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision 74*, 1 (2007), 59–73.

[Bor84] Borgefors G.: An improved version of the chamfer matching algorithm. In *7th International Conference of Pattern Recognition* (Montreal, 1984), pp. 1175–1177.

[Can86] Canny J.: A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell. 8*, 6 (June 1986), 679–698.

[CMK03] Chum O., Matas J., Kittler J.: Locally optimized ransac. In *the Annual Pattern Recognition Symposium of the German Association for Pattern Recognition (DAGM)* (2003), pp. 236–243.

[DT05] Dalal N., Triggs B.: Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01* (Washington, DC, USA, 2005), CVPR '05, IEEE Computer Society, pp. 886–893.

[DWB06] Durrant-Whyte H., Bailey T.: Simultaneous localization and mapping (slam): Part i. *Robotics Automation Magazine, IEEE 13*, 2 (June 2006), 99–110.

[FB81] Fischler M. A., Bolles R. C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24*, 6 (June 1981), 381–395.

[FBF77] Friedman J. H., Bentley J. L., Finkel R. A.: An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw. 3*, 3 (Sept. 1977), 209–226.

[FN75] Fukunage K., Narendra P. M.: A branch and bound algorithm for computing k-nearest neighbors. *IEEE Trans. Comput. 24*, 7 (July 1975), 750–753.

[GHT11] Gauglitz S., Höllerer T., Turk M.: Evaluation of interest point detectors and feature descriptors for visual tracking. *Int. J. Comput. Vision 94*, 3 (Sept. 2011), 335–360.

[Has14] Hast A.: Robust and invariant phase based local feature matching. In *Proceedings of the 22nd International Conference of Pattern Recognition (ICPR)* (2014), pp. 809–814. Poster with Paper.

[HM13] Hast A., Marchetti A.: Rotation invariant feature matching - based on gaussian filtered log polar transform and phase correlation. In *Proceedings of the 8th International Symposium on Image and Signal Processing and Analysis (ISPA)* (2013), pp. 100–105.

[HNM13] Hast A., Nysjö J., Marchetti A.: Optimal ransac - towards a repeatable algorithm for finding the optimal set. *Journal of WSCG no.1* (2013), 21–30.

[HS88] Harris C., Stephens M.: A combined corner and edge detector. In *Proceedings of The Fourth Alvey Vision Conference* (1988), pp. 147–151.

[HZ03] Hartley R. I., Zisserman A.: *Multiple View Geometry â 2nd edition*. Cambridge University Press, 2003.

[HZ04] Hartley R. I., Zisserman A.: *Multiple View Geometry in Computer Vision*, second ed. Cambridge University Press, ISBN: 0521540518, 2004.

[LCS11] Leutenegger S., Chli M., Siegwart R. Y.: Brisk: Binary robust invariant scalable keypoints. In *Proceedings of the 2011 International Conference on Computer Vision* (Washington, DC, USA, 2011), ICCV '11, IEEE Computer Society, pp. 2548–2555.

[Low04] Lowe D. G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (2004), 91–110.

[ML09] Muja M., Lowe D. G.: Fast approximate nearest neighbors with automatic algorithm configuration. In *In VISAPP International Conference on Computer Vision Theory and Applications* (2009), pp. 331–340.

[ML12] Muja M., Lowe D.: Fast matching of binary features. In *Computer and Robot Vision (CRV), 2012 Ninth Conference on* (May 2012), pp. 404–410.

[MS05] Mikolajczyk K., Schmid C.: A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 10 (Oct. 2005), 1615–1630.

[NT13] Nakhmani A., Tannenbaum A.: A new distance measure based on generalized image normalized cross-correlation for robust video tracking and image recognition. *Pattern Recogn. Lett. 34*, 3 (Feb. 2013), 315–321.

[RD06] Rosten E., Drummond T.: Machine learning for high-speed corner detection. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part I* (Berlin, Heidelberg, 2006), ECCV'06, Springer-Verlag, pp. 430–443.

[SB97] Smith S. M., Brady J. M.: Susan - a new approach to low level image processing. *Int. J. Comput. Vision 23*, 1 (May 1997), 45–78.

[SMB00] Schmid C., Mohr R., Bauckhage C.: Evaluation of interest point detectors. *Int. J. Comput. Vision 37*, 2 (June 2000), 151–172.

[SNNL13] Sablina V. A., Novikov A. I., Nikiforov M. B., Loginov A. A.: An approach to the image superimposition problem in multispectral computer vision systems. In *2nd Mediterranean Conference on Embedded Computing (MECO), 2013* (June 2013), pp. 117–120.

[SNNL14] Sablina V. A., Novikov A. I., Nikiforov M. B., Loginov A. A.: Navigation parameters correction technique using multiple view geometry methods. In *Communication Paper Proceedings of the 22nd International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)* (Plzen, Czech Republic, 2014), pp. 103–110. Short Paper.

[ST94] Shi J., Tomasi C.: Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR, 1994 IEEE Computer Society Conference on* (June 1994), IEEE, pp. 593–600.

[Sze06] Szeliski R.: Image alignment and stitching: a tutorial. *Found. Trends. Comput. Graph. Vis. 2*, 1 (January 2006), 1–104.

[TM08] Tuytelaars T., Mikolajczyk K.: Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision 3*, 3 (July 2008), 177–280.

[ZKM04] Zuliani M., Kenney C., Manjunath B. S.: A mathematical comparison of point detectors. In *Second IEEE Image and Video Registration Workshop (IVR)* (Jun 2004), pp. 172–178.

Heterogeneous Dataset Acquisition for a Continuously Expandable Benchmark (CEB)

Bernd Krolla
DFKI - German Research
Center for Artificial Intelligence
bernd.krolla@dfki.de

Didier Stricker
DFKI - German Research
Center for Artificial Intelligence
didier.stricker@dfki.de

ABSTRACT

Ongoing research within the field of computer vision yielded a wide range of image based 3D reconstruction approaches. Starting years ago with low resolution RGB images as input, we face today a wide and fast growing range of available imaging devices to perform this task.
To allow for a good comparability of resulting reconstructions, many different benchmarks and datasets have been made available. At the same time, we observe, that these benchmarks commonly address only a single capturing approach omitting the chance to compare against results of other acquisition methods.
In contrast to such homogeneous benchmarks, we present in this work a heterogeneous benchmark, considering different acquisition devices to obtain our datasets. Besides these datasets, we furthermore provide reference data for download.
To lastly keep track of the rapidly increasing number of different acquisition sensors, we opt to provide occasional updates of this benchmark within the future.

Keywords

Computer Vision, 3D Reconstruction, Benchmark, Heterogeneous Dataset Acquisition

Within the field of computer vision, image-based 3D reconstruction of objects and environments has been subject to intense research since many years. Gradually, the estimation of essential and fundamental matrices [7], camera calibration [20] and multiple view reconstruction [7] was understood and improved [11, 12, 16, 17].
Having calibrated camera parameter as well as sparse pointclouds of a scene at hand, many different reconstruction algorithms have been developed, to generate notable image based reconstruction results such as [2, 3, 5, 15].
While most of the former approaches for image-based 3D reconstruction rely on the processing of perspective RGB-images, the computer vision community can nowadays access a rapidly expanding variety of new sensors:
Recent developments introduced technical devices such as *high definition* and *4K* video-cameras, *high dynamic range (HDR)* imaging devices, *RGB-depth (RGBD)* cameras, consumer cameras capturing at frame rates of *90Hz* and more, *stereo* cameras, *light field* cameras, *Time of Flight (ToF)* cameras and many more. Various of those devices are capable to offer new approaches for 3D reconstruction, which are commonly addressed in the context of ongoing research. To access and quantify the potential of such newly developed algorithms, a wide range of benchmarks has been made available [4, 6, 8, 9, 14, 18, 19].
When taking these above listed benchmarks for 3D reconstruction into consideration, we claim that they do not yet allow for a comparison of reconstruction algorithms, which rely on different acquisition approaches: A benchmark for RGB-image based 3D reconstruction allows for a comparison of different algorithms which rely on RGB-data. But the very same benchmark excludes any performance assessment with respect to approaches which apply RGB-D, lightfield or video data. This lack of comparability of reconstruction approaches is therefore the underlying motivation for the publication of our presented benchmark.

Contribution We introduce in this work a benchmark consisting of datasets captured from a small set of objects by applying a heterogeneous variety of acquisition sensors. We furthermore aim at a continuous expansion of the dataset by making acquired data from new devices available in the future.
The core contribution within this work is therefore summarized as follows: We selected a set of objects, which provide different challenges for 3D reconstruction We

captured datasets using imaging devices of different kind and quality and make those publicly available. We provide reference data using a structured light approach [1].

Nomenclature Within the further course of this work, all acquisition devices used for the actual capturing are referred as *devices*.
Any physical subject, which has been acquired by a *device* is referred as *object*. It is noteworthy, that an object can therefore also be assembled from multiple, jointly mounted items.
A *dataset* furthermore refers to the digitalized data resulting from an acquisition process of an *object* by using a *device*.
An *environment* is finally considered in this work to contain all surroundings of the acquisition setup, such as background or illumination situation.
All datasets are finally combined into one benchmark, presented in this work. Since the authors intent to add future datasets, whenever new acquisition devices become available, the benchmark may be referred as *CEB (Continuously Expandable Benchmark*).

1 RELATED WORK

In [18], Seitz provided the well-known Middlebury benchmark, consisting of a main set of 2 different objects, acquired from 317 different camera positions while supplying according camera parameter. Reconstruction results obtained from the provided images can be submitted and benchmarked against ground truth data.
Furukawa provided a similar scenario as benchmark in [4]. While providing images with a significantly higher resolution and calibrated camera parameter, ground truth is not provided for all datasets.
Jensen introduced a benchmark in [8], which provides in contrast to the aforementioned benchmarks a wider range of acquired objects, while using a 6-axis industrial robot for the image acquisition. Moreels provided in [14] a benchmark, containing 3D objects on a turntable under varying illumination conditions. The dataset however, remains without ground truth data.

Ground truth vs. Reference Within a complete and meaningful benchmark, the careful generation of an accurate ground truth is however always an important point. While benchmarks, which rely on synthetically generated data are capable of providing ideal ground truth data, any measurement based ground truth acquisition is always subject to error prone measurements.
The different implications in this context are discussed in detail by Kondermann in [10].
To minimize the occurring errors to a minimal ratio, Seitz [18] combined more than 200 laser scans of a single object and applied super resolution algorithms for an improved overall result. At the same time, the actual images of the dataset were provided at a relatively low resolution of 640x480 pixel.
Strecha provided in [19] a laser scan as ground truth for their reconstruction challenges. In their work, they estimated the expected precision of the acquired scans and supplied the ground truth together with an estimated variance of the obtained 3D points.

2 OUTLINE

The remainder of this paper is organized as follows: In Section 3, we detail the choice of objects, which were used within the benchmark. Subsequently we elucidate the acquisition process of the individual datasets in Section 4 and present a set of reference measurements. We discuss and conclude this work in Section 5. For further material the reference may be made to the supplementary material, submitted in conjunction with this work. The benchmark itself is available at `http://ceb.dfki.uni-kl.de`.

3 DATASET COMPOSITION

The presented benchmark consists of a total set of 11 different objects, containing various reconstruction scenarios and challenges.
Parts of these objects are composed from groups of items, other datasets consist of single objects.
An important prerequisite to all selected objects is the expected longterm usability for reconstruction purposes to comply with the previously introduced option for future expansion of the *CEB* by adding further datasets.
To satisfy this requirement, exclusively rigid objects were selected to be part of the benchmark. The different objects themselves unify furthermore various geometric challenges including repetitive structures, self occlusions, smooth, irregular, convex and concave surfaces. The benchmark is furthermore characterized by various different surfaces subsumed by the different objects, including wood, plaster, painted plaster, plastics, metal, Styrofoam and others.
In summary, Table 2 provides an overview over the different objects and their main characteristics, while Table 1 gives an overview over the naming conventions for the accompanying camera parameter.

4 ACQUISITION PROCESS

4.1 Preparation

Preceding to the first data acquisition, all objects were mounted on top of quadratic base plates with an edge length varying between 10*cm* and 30*cm*, acknowledging the varying overall size of the objects as listed in Table 2.
Each plate contains a set of drilled holes to allow for a precise mounting on different underground and environments. To assure a stress-free mounting, the objects

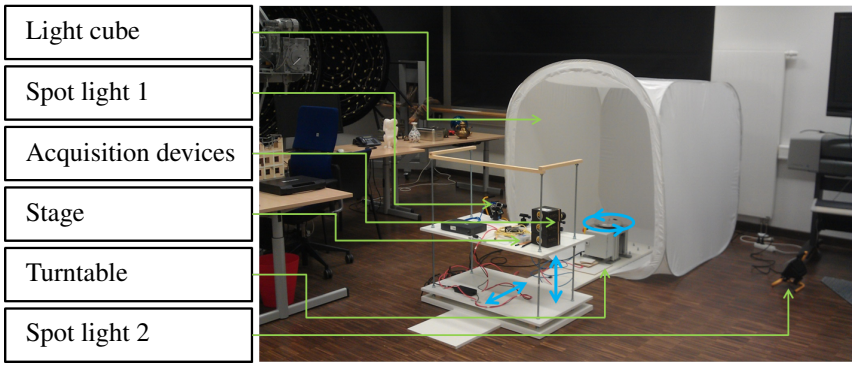


Figure 1: Main acquisition environment: A turntable mounted inside a light tent, being surrounded and illuminated by a set of point light sources. The whole setup is software controlled, placed within a windowless room and allows for a positioning of objects with a precision of 0.012°.

were not skewed onto the mounting plates. Instead, all objects were fixed using a low temperature glue, which avoided thermal dependent tensions during the cooling process.
Taking these measures into account, the authors consider the objects to be ready to meet the requirements for long term availability.

4.2 Dataset acquisition

The wide majority of the provided datasets was acquired within an indoor environment with constant and controlled acquisition conditions. We aimed hereby towards a good reproducibility of external parameters and comparability between different camera types in respect to various acquisition characteristics, such as point of view, number of acquired images and illumination conditions.
To assure the comparability of view points onto the objects for different camera types, all mounting plates with their attached objects were setup on top of a turntable as depicted in Figure 1. The turntables intrinsic positioning precision allowed hereby to approach 240'000 different equally distributed positions in the course of a single 360° turn leading to an angular resolution of 0.0015° with a positioning uncertainty of 0.012° as stated by the manufacturer. This positioning mode was used to line up the objects and to acquire images with the varying imaging devices.
The turntables rotation mode was applied to capture videos with varying devices. The objects rotation was hereby captured at varying velocities in the range of 12'000, 10'000, 8'000, 6'000, 4'000, 2'000 motor steps per minute (corresponding to $\frac{1}{3}$, $\frac{2}{5}$, $\frac{1}{2}$, $\frac{2}{3}$, 1 and 2 *rpm*).

Camera calibration Preceding to each dataset acquisition, a calibration of camera parameter was conducted as proposed by Vogiatzis and Hernández in [21]. The resulting images with the calibration pattern are provided along with the retrieved intrinsic parameter provided for the download.

Illumination To ensure a well defined and reproducible illumination situation, we chose a windowless room for the object acquisition to be independent from any daylight changes. The turntable with the mounted objects was placed inside a light tent, which served as light diffuser. The illumination of the setup was then provided by 3 point light sources. The choice of halogen lamps allowed for a natural illumination compared to narrowband LED-spectra.

Illumination documentation To allow for a color calibration of the individual capturing devices, we acquired

Table 1: Exemplary listing of provided extrinsic and intrinsic parameter for a DSLR-camera. Note: *fx,fy,cx,cy* and α are provided as a joint camera matrix **K**, together with a distortion matrix **D**. Other camera types, such as lightfield cameras, depth or stereo cameras are provided with their individually adapted setting.

Name	Type	Description
hd	Extrinsic	Horizontal distance between cameras principle point and turn table base
vd	Extrinsic	Vertical distance between cameras principle point and turn table base
fx	Intrinsic	Cameras focal length, expressed in pixels
fy	Intrinsic	Cameras focal length, expressed in pixels
cx	Intrinsic	Horizontal coordinate of the cameras principle point
cy	Intrinsic	Vertical coordinate of the cameras principle point
α	Intrinsic	Skew value of the camera sensor

 ISBN 978-80-86943-65-7

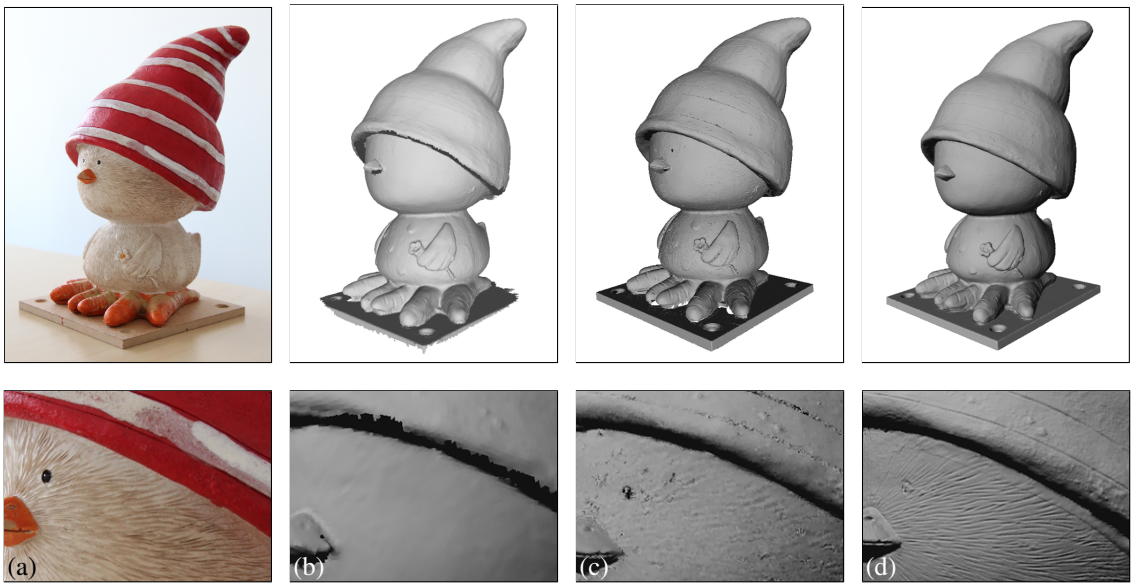


Figure 2: Detail of different acquisition approaches for reference generation: Closeup of the dataset (a). Resulting mesh from the hand held artec scanner acquisition consisting of 620k faces (b). Resulting mesh of the laser scanning approach consisting of 2.05M faces (c). Resulting mesh from the structured light approach as provided by [1] consisting of 45.9M faces (d).

a small set of images of a Macbeth ColorChecker board [22] before capturing the actual datasets. These images allow for a color calibration of the devices, being able to compensate automatic white balancing as enabled by some of the capturing devices. These acquired images are made available within the dataset, without using them to correct for any color balancing of the dataset images.

Logging and documentation To allow for a good understanding of the utilized camera setup and a good traceability of the performed steps and actions during the dataset acquisition, we setup and used a set of logging tools to check and log various types of data and parameters. Using this approach, we assured the documentation of each dataset acquisition with respect to currently chosen type of scene illumination, selected camera parameters and further information.

Further environments Some of the provided datasets were acquired in different environments: To add further characteristics to the benchmark, a small subset of datasets was acquired in a non reproducible manner with limited control of the environmental conditions. Exemplary, we refer to the provided freehand acquisitions in an outdoor environment, which expands the variety of reconstruction scenarios, but depends heavily on the experimenters camera handling and the current weather conditions, making it practically impossible to exactly reproduce an identical scenario for further capturings with different cameras.

4.3 Acquisition devices

The overall set of employed acquisition devices sums up to 7 different devices, while some of those were used for the acquisition of multiple datasets, differing in terms of acquisition mode and acquisition environment: One might consider DSLR cameras used in an *indoor* acquisition scenario in *video* mode and in *outside* acquisition scenarios taking hand held *images* of a dataset.
In general, the acquisition devices can be split up into different groups taking different characteristics into consideration:

Active vs. passive The majority of the applied acquisition devices is characterized by its passive acquisition process, exploiting exclusively incoming illumination emitted by the scene itself.
Active acquisition devices, characterized by their emission of sampling patterns are commonly susceptible to strong surrounding illumination. We therefore did not acquire any outdoor datasets using active acquisition devices. For the standardized indoor acquisition process, however, we used the probably most prominent representative, Microsofts Kinect 360 [13], which relies on the emission of a dot-pattern within the infrared frequency domain.

Image vs. video capturing The presented benchmark provides image-based as well as video-based datasets.
The image-based dataset acquisition of different objects consists hereby in a number of 200 images

 ISBN 978-80-86943-65-7

Figure 3: Visualization of the current download interface: The different datasets are organized in an array with respect to the different objects and the acquisition device (left). Datasets can be selected for download in three different ways: Column-based to download all datasets of a certain acquisition device, row-based to download all datasets of a certain object and individually selected to download a specific device-object-combination (right).

for the main acquisition environment, acquired at well defined object positions.
Some of the devices were ready to allow for video as well as image capturing. For those, we provide image and video based datasets.
Information, referring to the images or videos, such as resolution, frame rate or compression algorithms is provided with the individual datasets.

Mounting Regarding the mounting of the acquisition devices, we intended to satisfy two different demands: To allow for a good reproducibility, most acquisition devices were rigidly connected with a solid acquisition stage as shown in Figure 1. To handle acquisition scenarios, as performed by end users, we furthermore added video and image datasets with hand held acquisition.

For several acquisition devices, such as light-field cameras, special considerations were respected to provide an appropriate acquisition scenario. These considerations are then listed within the corresponding logging files of the datasets.
A complete tabular overview of all devices, which were considered for the dataset acquisition is provided in Table 3. For each dataset, we specify therein a set of extrinsic and intrinsic parameter, which is downloadable along with the imaging data.

4.4 Reference acquisition

In order to allow for a meaningful evaluation of different reconstruction approaches, we provide 3D models of the objects. To acquire those, we considered a variety of different approaches:

Artec Spider The reference data, which was acquired with this hand held 3D scanner was our first approach to provide 3D models of the objects. The resulting models were generated from multiple partial scans, which were aligned against each other using provided software.
Manual operation however results in SLAM-like acquisition approach, while the translation of the device during the acquisition leads possibly to a less precise registration of the camera positions (See Figure 2(b)).

Industry scanner We provided the objects furthermore to a laser scanning supplier, leading to reconstruction results as shown in Figure 2(c).

Structured light scaning approach Best reconstruction accuracies however were achieved using a structured light approach [1] as shown in Figure 2(d).

Figure 2 provides an overview over the provided reference datasets. Visual inspection demonstrates the varying level of reconstructed details for the different approaches.
Complying with the previously stated concept to provide an *Continuously Expandable Benchmark*, we do not consider these reconstructions as ground truth (implying to provide *perfect* data, but aim to provide reference reconstructions (*as good as possible*), leaving room to possible future improvements of reconstruction algorithms.

5 RESULTS AND DISCUSSION

We make the benchmark, as a result of the previously detailed acquisition work publicly available to the

 ISBN 978-80-86943-65-7

computer vision community. Acquired datasets as well as the introduced reference data is provided for download as shown in Figure 3.
We furthermore aim to occasionally provide new datasets to the benchmark within the future.

6 ACKNOWLEDGEMENTS

The authors would like to thank Norbert Schmitz for the turntable setup, Johannes Köhler for the reference generation, Bertram Taetz for the support in context of the Raytrix capturing, Moshin Munir for the Kinect and outdoor acquisitions and Santosh Shah for his implementation of the web interface.
The work was carried out during a research cooperation between the Computational Imaging Group at the Stuttgart Technology Centre of Sony Deutschland GmbH and the German Research Center for Artificial Intelligence (DFKI). We would like to thank in particular Yalcin Incesu and Oliver Erdler from Sony Stuttgart for their feedback and fruitful discussions. This work was cofunded by the BMBF-project DENSITY (01IW12001).

REFERENCES

[1] 3digify.com. http://www.3digify.com, 2015.

[2] Christian Bailer, Manuel Finckh, and Hendrik PA Lensch. Scale robust multi view stereo. In *Computer Vision–ECCV 2012*, pages 398–411. Springer, 2012.

[3] Simon Fuhrmann and Michael Goesele. Floating scale surface reconstruction. *ACM Transactions on Graphics (TOG)*, 33(4):46, 2014.

[4] Y. Furukawa and J. Ponce. 3d photography dataset http://www-cvr.ai.uiuc.edu/ponce_grp/data/mview/, May 2006.

[5] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:1362–1376, 2010.

[6] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M Seitz. Multi-view stereo for community photo collections. In *IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[7] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[8] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanaes. Large scale multi-view stereopsis evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 406–413, June 2014.

[9] Changil Kim, Henning Zimmer, Yael Pritch, Alexander Sorkine-Hornung, and Markus H Gross. Scene reconstruction from high spatio-angular resolution light fields. *ACM Trans. Graph.*, 32(4):73, 2013.

[10] Daniel Kondermann. Ground truth generation http://resources.mpi-inf.mpg.de/conferences/up2013/up2013_files/up2013-abstracts/kondermann/daniel-kondermann.pdf, 2013.

[11] Quan-Tuan Luong and Olivier D Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17(1):43–75, 1996.

[12] Paulo RS Mendonça and Roberto Cipolla. A simple technique for self-calibration. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, 1999.

[13] Microsoft. Microsoft Kinect 360 http://www.xbox.com/en-US/kinect.

[14] P. Moreels and P. Alatorre. 3d objects on turntable http://www.vision.caltech.edu/pmoreels/Datasets/TurntableObjects/.

[15] P. Moulon, P. Monasse, and R. Marlet. Global fusion of relative motions for robust, accurate and scalable structure from motion. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3248–3255, Dec 2013.

[16] Marc Pollefeys, Reinhard Koch, and Luc Van Gool. Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters. *International Journal of Computer Vision*, 32(1):7–25, 1999.

[17] Fabio Remondino and Sabry El Hakim. Image based 3d modelling: A review. *The Photogrammetric Record*, 21(115):269–291, 2006.

[18] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 519–528. IEEE, 2006.

[19] C. Strecha, W. Von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. 2008.

[20] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – a modern synthesis. *Vision algorithms: theory and practice*, pages 153–177, 2000.

[21] George Vogiatzis and Carlos Hernández. Automatic camera pose estimation from dot pattern, http://george-vogiatzis.org/calib/, 2010.

[22] xRite Inc. Colorchecker classic http://xritephoto.com/, 2014.

Table 2: Table of the acquired objects

Name	Picture	Plate, Height	Material	Geometry	Reflectivity	Further characteristics
Buddha		10x10cm, $\approx 8cm$	Coated plaster	Minor occlusions	Highly reflective	Due to the high reflectivity, the capturing environment tends to impact the reconstruction
Chicken		10x10cm, $\approx 18cm$	Plastics	Moderate complexity	Diffuse	Feather-like surface contains chamfers $< 1mm$ width, self-occlusions under the had brim
Cup		10x10cm, $\approx 9cm$	Glazed ceramic	Smooth and even	Specular surface in untextured regions	Untextured, specular concave interior of the cup, minor self-occlusions due to cups handle
Dragon		15x15cm, $\approx 20cm$	Coated plaster	Complex microscopic structures of the surface.	Diffuse	Surface contains chamfers $< 1mm$ width, contains partial self-occlusions
Elephant		15x15cm, $\approx 13cm$	Coated plaster	Moderate complexity	Highly reflective	Surface contains chamfers $< 1mm$
Elk		10x10cm, $\approx 15cm$	Wood	Moderate complexity	Diffuse	Antlers introduce self-occlusions
Mbdhct		30x30cm, $\approx 10cm$	(Painted) wood, plastics, ceramics, metal	Usage of multiple objects causes self-occlusions	Different types, mostly diffuse	Nomenclature: Mole, Box, Duck, Home (sweet Home), Clock, Teapot
Metal-objects		15x15cm, $\approx 16cm$	Metal (also plastic and Styrofoam®)	The usage of multiple objects causes self-occlusions	Mainly metallic surface implying reflectivity	The implied screw thread represents a highly repetitive pattern.
Owl		15x15cm, $\approx 25cm$	Thin metal sheets, implies minor self-occlusions	Painted metal sheets, transparent glass eyes	Diffuse. Exception: eyes	Upper body is flexibly mounted onto the lower body, allowing for nonrigid dataset acquisition
Santa		10x10cm, $\approx 15cm$	Painted clay	Smooth surface without occlusions.	Diffuse	Feature based reconstruction approaches might work best for high resolution images, which resolve minor texture variations of the object
Scw		30x30cm, $\approx 19cm$	Usage of multiple objects causes self-occlusions	Contains partially transparent surfaces	Metallic, transparent and semi-transparent surfaces	Contains repetitive, structures (Threads). Nomenclature: Shampoo, (CPU)-cooler, Wifi-card.

 ISBN 978-80-86943-65-7

Table 3: Table of considered acquisition devices (open to future extensions).

Device	Picture	Full name	Specification	Characteristics
Bloggie		Sony Bloggie 3D	Full HD Stereo camera	Provides full-hd stereo images (1920x1080, mpo, jpg) and hd video (mp4, 1920x1080px @30fps)
Eos5		Canon EOS5 Mark II	Professional DSLR camera	Provides raw (cr2) and jpg images (resolution 5616x3744pixel)
Eos500		Canon EOS500	DSLR camera	Provides raw (cr2) and jpg images (resolution 4752x3168pixel) and full-hd video (mov)
HTC		HTC Desire HD	Smartphone	Provides hd video (3gp, 1280x720px @30fps)
Kinect		Microsoft Kinect Xbox 360	RGBD camera	Provides frames of 640x480px @30fps (when capturing video)
Raytrix		Raytrix R5	Light field camera	4.2 Megarays, 2048x2048pixel @25fps (GigE)
Techsolo		Techsolo TCA-4810 Webcam	Webcam	640x480 @15fps (avi)

Explorative Analysis of 2D Color Maps

M. Steiger*, J. Bernard*, S. Thum*, S. Mittelstädt†, M. Hutter*, D. Keim†, J. Kohlhammer*
Fraunhofer IGD, Darmstadt*, University of Konstanz†, Germany

ABSTRACT

Color is one of the most important visual variables in information visualization. In many cases, two-dimensional information can be color-coded based on a 2D color map. A variety of color maps as well as a number of quality criteria for the use of color have been presented. The choice of the best color map depends on the analytical task users intend to perform and the design space in choosing an appropriate 2D color map is large. In this paper, we present the ColorMap-Explorer, a visual-interactive system that helps users in selecting the most appropriate 2D color map for their particular use case. ColorMap-Explorer also provides a library of many color map implementations that have been proposed in the scientific literature. To analyze their usefulness for different tasks, ColorMap-Explorer provides use case scenarios to allow users to obtain qualitative feedback. In addition, quantitative metrics are provided on a global (i.e. per color map) and local (i.e. per point) scale. ColorMap-Explorer enables users to explore the strengths and weaknesses of existing as well as user-provided color maps to find the best fit for their task. Any color map can be exported to be reused in other visualization tools.
The code is published as open source software, so that the visualization community can use both the color map library and the ColorMap-Explorer tool. This also allows users to contribute new implementations.

Keywords

explorative analysis, color maps

1 INTRODUCTION

Color is one of the most important visual variables in information visualization. Depending on the properties of the underlying data, different types of color maps can be applied to encode data attributes visually in the most accurate way. Qualitative color maps allow for the distinction between different categories of elements. Quantitative color maps allow for an identification of similar (and dissimilar) data elements with respect to a quantitative value domain. For quantitative color maps, the most relevant representatives are either sequential (unipolar) or diverging (bipolar). In those cases where a single data variable (attribute) is encoded, a one-dimensional color ramp can be used.

For high-dimensional data, 2D color maps are used to preserve similarity of the items in a visual variable. Data items with more than two attributes are first mapped into the two-dimensional space according to some transformation or projection method. The result of these upstream techniques is a mapping in 2D that can directly be used as position information in a 2D color map.

As a result, the viewer can estimate the relative similarity of high-dimensional data by comparing colors. As such, 2D color maps are appropriate for high-dimensional data; we do not recommend the direct use of two data attributes as coordinates in the map (cf. Wainer et al. [WF80]).

A variety of different static 2D color maps has been presented in the past. The survey of Bernard et al. gives an overview [BSM*15]. The authors review quality criteria and design guidelines for color maps and depict the huge design space for the *design* and the *use* of static 2D color maps.

In order to faithfully reflect the relative pair-wise distances of the original data as closely as possible, such a 2D color map should preserve the notion of perceived similarity in terms of color. The perceived distance between colors should be linearly related to the geometric distance in both the high- and the 2-dimensional space. Another quality criterion for a color map is to exploit the given color space, aiming for a maximum number of distinguishable colors. In many cases the choice of color maps is also made with respect to colorblindness sensitivity. For example, about 8-10 percent of the male population in Europe suffer from a color vision deficiency [Alb10]. Additional requirements to color maps may be based on user-centered constraints like corporate designs. In some cases, 2D color maps may also require a certain contrast against the background color so that the visual elements can be clearly identified as such. Some other visualizations may require that text

and other overlays are legible on a canvas that is drawn based on the color map.

A taxonomy of different color map design criteria is presented by Tominski et al. [TFS08]. According to the authors, meaningful color encodings strongly depend on the data, the task, the target user group, and the display device. A fourth dimension in the problem space is the large number of static 2D color maps presented in literature. Naturally, there is no color map that is perfect with respect to all requirements. To give an example, a color map can hardly be colorblind-safe and maximize color exploitation at the same time. Visualization designers need to balance a trade-off between different complementary design criteria. A premature color map decision may lead to false assumptions with respect to the underlying data properties. Consequently, choosing a 2D color map for a visualization should be done carefully.

To the best of our knowledge, a decision support system that supports the user in making such a choice has not yet been presented. We identify the following challenges:

- $\mathbf{R}_1$:Visual overview of existing color maps
- $\mathbf{R}_2$:Comparison of color maps with respect to global quantitative quality measures
- $\mathbf{R}_3$:Assessment of local properties of a color map
- $\mathbf{R}_4$:Visual analysis of the shape of a color map with respect to different color spaces.
- $\mathbf{R}_5$:Assessment of the maximum amount of discernible information that can be encoded
- $\mathbf{R}_6$:Showing the homogeneity of perceived similarity
- $\mathbf{R}_7$:Assessment of the interplay of color map with other visual variables

We present the ColorMap-Explorer, a visual-interactive decision support system for 2D color maps. The system assists visualization designers to find the best-fitting color map in this complex search space. At the moment, it contains 22 color map implementations that were discussed in the scientific literature. Visual access to these 2D color maps is provided in an overview visualization. For every color map, quantitative metrics are provided on a global (i.e. per color map) and local (i.e. per point) scale. For the comparison of multiple color maps, we provide a view utilizing the global measures. A detailed analysis of local properties is provided by several views, each shedding light from a different perspective. In particular, we allow for the detailed analysis of a) different color channels b) local perceptual linearity, and c) the shape of the area in different color spaces for every color map. In order to get a first impression of how the color map behaves in a targeted downstream visualization, several views stress the color map against other visual variables in different example scenarios. Finally, the selected color map can be exported for re-use in downstream visualization tools.

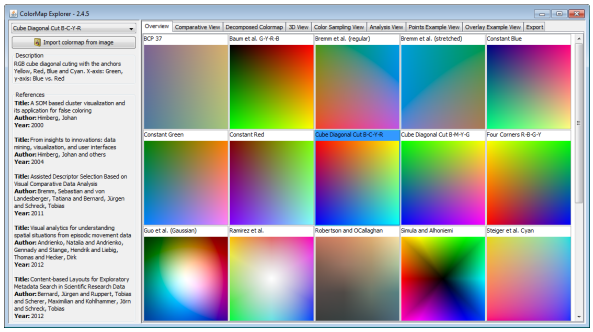


Figure 1: The main window of the ColorMap-Explorer: the config and info panel is placed on left side, the collection of views is stored in individual tabs at the right. The Overview tab enumerates all available color maps.

The workflow of the ColorMap-Explorer is as follows: starting with an overview of all color map implementations, the user can select up to three color maps which then are put in juxtaposition. This allows for direct comparison to narrow down the number of candidates with respect to the analytical task. Individual color maps are then investigated in more detail before the best fit is identified. When the decision on the best matching color map has been made, the user can save the color map as an image to disk. The user can always move backwards and forwards in this workflow pipeline as desired.

The paper is organized as follows: In Section 2, we discuss related software tools that support the user in finding colors for visualization tasks. Section 3 gives a definition of perceived color differences, before the ColorMap-Explorer is illustrated in detail in Section 4. We show some discoveries along an example application in Section 5. Conclusion and outlook are at the end of the paper in Section 6.

2 RELATED WORK

Appropriate color maps for specific tasks and specific data properties is a well discussed topic in the literature. General guidelines on selecting color maps can be found in [RO86, War88, RTB96, Rhe00]. In addition, linear color ranges (1D) for segmentation and categorical data have been discussed previously [Hea96, HB03]. For two-dimensional color maps there are few guidelines available. The study of Wainer et al. [WF80] showed that encoding of two dimensional data with two dimensional color maps is not intelligible. In contrast to this statement, Ware and Beatty [WB88] found that each additional color dimension (red, green, blue channel) is as effective as an additional spatial dimension

in the encoding of multivariate (more than two dimensional) data. As described in [MBS*14], there is a difference between encoding single data dimensions with color and encoding (multidimensional) data relations. The first case requires a precise mapping of one data dimension to one color dimension or a one dimensional color map. The second case involves multiple dimensions for each visual object whose characteristics and relations to other objects should be revealed by color.

In [MBS*14] the authors present data-driven quality measures that are used to perceptually optimize color mapping for high-dimensional data. These measures are very effective if and only if the data set and its distribution as well as subsets (e.g., classes or clusters within the data) are known apriori and should be preserved in the color mapping. In this paper, we focus on a data-independent approach, which focuses rather on the analysis tasks and not on data properties.

In multivariate data analysis applications, two dimensional color maps have been successfully applied [RO86, Him98, SA99, SvLB10, BvLBS11, SBM*14, BSW*14] (see Figure 2 for an overview of re-implemented color maps). From this background, many two dimensional color maps have been proposed in the literature, each with different strengths and weaknesses. A recent survey has been conducted by Bernard et al. [BSM*15], enriched with a quality assessment for different tasks. Our work uses their quality metrics and provides them in an interactive manner to the user. In many aspects, our tools is similar to *PRAVDAColor*, an IBM software module that aims at supporting the user in selecting the right color map [BRT95]. Its main feature, however, is a set of perception-based rules that makes suggestions depending on task and data type.

3 PERCEIVED COLOR DISTANCE

For the rest of this paper, we will refer to a measure that indicates how similar two colors are. A reliable measure has to take the human visual system into account. In this section, we give a definition of the metric used to measure perceived color differences, also known as ΔE. Such a difference is close to zero if two colors are perceived as equal and close to 1.0 when the difference between two colors is "just noticeable" (visible by half the observers).

This definition of ΔE is based on the standardized Color Appearance Model (CAM) CIECAM02 [MFH*02]. Luo et al. have defined a ΔE for CIECAM02, based on the idea that a CAM should be a natural candidate to define a ΔE because similarity of colors should be rooted in their appearance attribute correlates [LCL06]. The authors compare appearance attribute differences to well-known color difference data sets and obtained a color difference formula and different parameterizations for the formula (see below).

They report that the predictive performance of the overarching CAM02-UCS parametrization is comparable to the specific parameterizations for small and large distances. This property is of particular importance for the evaluation of color maps, because it enables a quantification of the data-perception mismatch even when color differences are large. Previous color difference formulas were only designed and validated for small color differences.

A short definition of the ΔE is given in Equation 1; we refer to the original work of Luo et al for an exhaustive one [LCL06]. We start with CIECAM02 color appearance attribute correlates J (Lightness), M (Colorfulness), and h (Hue) and constants K_L, c_1, and c_2. The constants serve the purpose of fitting to small color distance (SCD) and/or large color distance (LCD) data, resulting in CAM02-SCD, CAM02-LCD respectively and CAM02-UCS (i.e. uniform) when fitted in combination.

$$\begin{aligned} J' &= \frac{(1+100c_1)J}{1+c_1J} \\ M' &= (1/c_2)ln(1+c_2M) \\ a' &= M'cos(h) \\ b' &= M'sin(h) \\ \Delta E = \Delta E_{UCS} &= \sqrt{(\Delta J'/K_L)^2 + \Delta a'^2 + \Delta b'^2} \end{aligned} \tag{1}$$

Discounting for the constants, ΔE_{UCS} is an euclidean distance defined in a suitable derivate of CIECAM02. Effectively, J is being expanded by about 20%, with the coefficient c_1 actually being constant across the SCD, LCD, and UCS variants. On the other hand, the colorfulness M', is being compressed significantly, with noticeable differences between CAM02-LCD and SCD variants. According to Luo, this hints at unexplained psycho-visual differences in the chroma component when judging small and large color differences. However, most color maps do not rely on chromatic content alone to differentiate colors. The hue h remains unchanged.

In summary, ΔE is an approximation of perceived global and local color differences. Despite minor uncertainty regarding the role of the chroma component, is seems a very good assessment tool for quantifying the relation between value distance and perceived color distance inherent to color scales.

Being based on CIECAM02, ΔE_{UCS} could even account for differences in lighting conditions and surroundings, but this has not been studied. The measure is thus based on standard lighting conditions, the sRGB "typical lighting conditions" representative for office use.

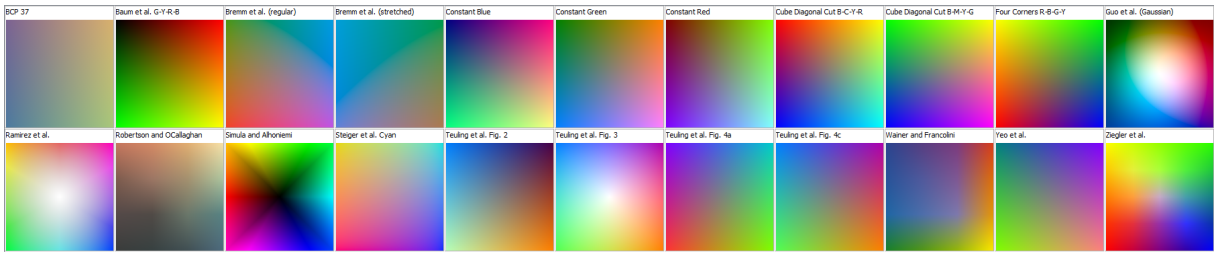


Figure 2: The initial overview lists all available color map implementations.

4 APPROACH

In this section, we present the different views of the ColorMap-Explorer along a typical workflow. The different views support the decision making process by showing individual features of the color maps. See Figure 1 for an overview screenshot of the software tool.

In total, 22 color maps mentioned in the information visualization literature have been re-implemented based on either functional description or digital images in publications. Software developers can extend the publicly available system by adding new implementations. In addition, an image-based file import enables non-experts to import custom color maps into the system. Thus, designers can easily extend the set of color maps and compare new designs with existing ones.

4.1 Overview Panel

Figure 2 shows all available color map implementations that currently exist in the ColorMap-Explorer. The Overview Panel lists all implementations as iconic images, annotated with name tags. The decision making workflow typically starts with this visualization, as this enables the analyst to gain an overview ($\mathbf{R}_1$). In this juxtaposition, the visualization designer can narrow down the set of candidates to the most relevant ones.

Criteria for this filtering step may be based on user preference such as the existence or lack of specific colors. The display device is yet another restricting aspect. For example, foreground and background colors influence the applicability for the visualization design. In addition, the analytical task may be a limiting aspect for the set of relevant color maps. A guideline for the fitness of specific color maps with respect to specific analytical tasks has been discussed by Bernard et al. [BSM*15]. Other criteria could be based on color theory or perceptual aspects such as brightness levels.

Individual color maps can be selected to get additional meta information such as scientific publications that define or reference the color map. In these publications, the user can find additional information on the construction, usage scenarios, etc. (see Figure 1, left). This information can be used to further narrow down the collection of candidates.

4.2 Comparative View

The Comparative View (see Figure 3) allows for the direct comparison of the most relevant candidates ($\mathbf{R}_2$). Six complementing quality measures indicate the fitness for a given analysis task (cf. [BSM*15]). These quality measures assess the global quality of the color map with a single value; we therefore refer to them as *global* measures. For every quality measure, score, and ranking information is provided to facilitate the comparison with all other color maps of the system. A boxplot chart displays the mean score (red line mark) and the range of 25% and 75% quantile (pink background). The 10% and 90% quantiles are indicated by a thin line (the whiskers). The color maps and their quality measures are put in juxtaposition. By that means, the visualization designer is enabled to directly compare global quality aspects of different color maps.

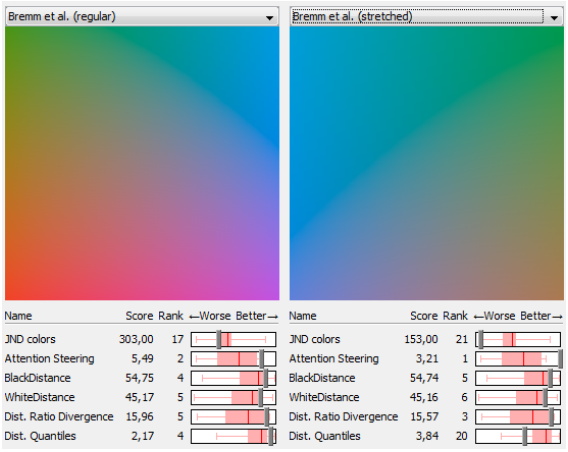


Figure 3: The Comparative View shows two selected color maps and their relative scores in different categories. This enables the user to compare scores.

As a result, the visualization designer can further reduce the number of candidates. An individual color maps can be analyzed further in the Decomposed View.

4.3 Decomposed View

The Decomposed View allows for the detailed analysis of single color maps. Two complementing aspects are considered. First, the color map is split into a set of color attributes from different color spaces ($\mathbf{R}_4$). Second, local features of every attribute can be analyzed ($\mathbf{R}_3$).

 ISBN 978-80-86943-65-7

Viewing multiple color attributes

In order to get an in-depth understanding of the properties of the color map, the map can be viewed from eight alternative perspectives. Each of them shows the same color map, but is filtered by a different color attribute.

We provide views for the red, green, and blue color components (center row in Figure 4), hue, saturation, and brightness (bottom row) as well as luma and attention steering (top row). Hue is special in that it highly depends on saturation. Without saturation, the value of hue is meaningless. Therefore, the tiles in the hue view are scaled according to its saturation. The original color map is shown in the top left panel. The first six values are directly extracted from the RGB and HSB color models.

Studies of Camgöz [CYG04] show that humans are predominantly attracted by bright and saturated colors. Attention steering effects may be harmful in several visual analysis tasks, because the analyst may be misled by striking features in the visualization that suppress less visual prominent features or patterns. Therefore, we approximate the potential of colors to attract the analyst's eye with $\sqrt{J^2+M^2}$ where J is the relative lightness and M the colorfulness. This definition accords to the findings of Camgöz et al. [CYG04]. However, it is an approximation of the attention steering effects and is yet to be evaluated. Therefore, we show both components J and M as decomposed views.

As a result, the homogeneity of a color map can be assessed. It also reveals how the color map is constructed. For example, the color map shown in Figure 4 is constructed by three diagonal color ramps in the RGB channels.

Revealing local characteristics

The spatial distribution of color in the different filtered views yields a variety of local features that can be validated. We support the user in identifying *variations* across the map with glyph-based annotations. As can been seen in Figure 4, the display of the individual views is discretized. This allows us to enrich the view with local glyphs, similar to vector field arrow grids that are well-known in the SciVis community.

We chose regular hexagons as spatial discretization, because this reveals equal spatial distances between tiles and all neighbors (in contrast to rectangular tiles). The number of tiles is automatically adjusted according to the viewport dimensions. Thus, the user can adjust the discretization level.

By default, each tile is annotated with a black arrow that indicates the perceived color distance with respect to its neighboring tiles. The length of the arrow represents the strength of the change, it points towards the strongest perceived change ($\mathbf{R}_6$). Changes are normalized across all color maps to allow for a fair comparison. The following pseudo-code illustrates the computation:

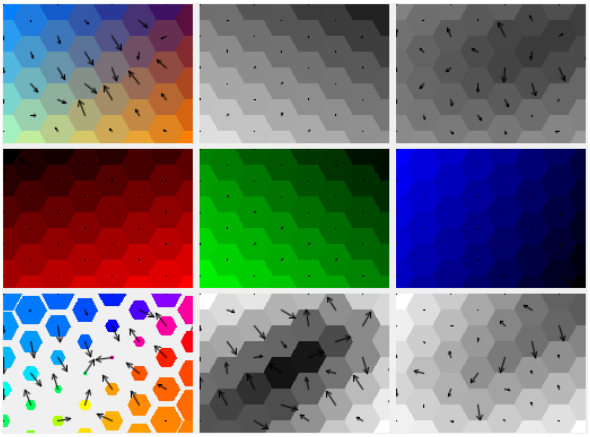


Figure 4: The color map (top left) is split into different components such as the color channels. The arrows point in the direction of the strongest perceived color change.

Algorithm 1 Compute difference arrows

```
Vector force ← [0, 0]
Color color ← tileColor(x, y)
for all Direction dir : directions(x, y) do
    Tile n ← tileModel.getNeighborFor(x, y, dir)
    Color ncolor ← tileColor(n.x, n.y)
    force += distance(color, ncolor) * dir
end for
return force
```

We compute it by averaging the color distances between the center of the tile and all neighbors in ΔE as defined in Equation 1. We avoid false assumptions caused by extrapolation by computing forces at border tiles only with a subset of tiles. As a consequence, arrows in border tiles always point along the border, never inside.

Interactive analysis of quantitative information

More detailed information on the ΔE distances of an individual tile is shown when hovering it with the mouse cursor (see Figure 5 – right). The arrow glyphs for the tile at the cursor and adjacent tiles are removed and a detailed glyph is shown instead. As a result relative color distances of a tile to the closest neighbors can be analyzed in detail. The glyph consists of six arrows, each pointing to the center of one of the neighboring tiles (i.e. they all have equal length). The stroke thickness indicates the perceived color change. Detailed quantitative information for the point in the color map at the cursor position is listed in tabular form in a separate info panel. This pane is partly depicted in Figure 5.

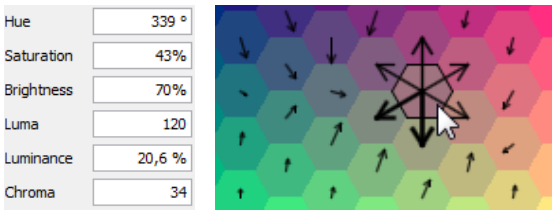


Figure 5: The detail arrow glyph shows individual differences to neighbor tiles. Quantitative information for that tile is given in the Info Panel at the left of the window.

The first two entries, X and Y, indicate the relative position of the mouse cursor on the color map in normalized coordinates. The next values represent the red, green, and blue component in the standard RGB color model that is used in many applications. The next three variables describe the color in the HSB color model, i.e. hue, saturation, and brightness. Hue is defined on a circular (connected) range from 0°-360°. Saturation and value are percentages.

CAM Lightness J is the brightness of a sample relative to the reference white. *CAM Hue* is the hue as defined in CIECAM02, which is not fundamentally different from other hue definitions, but is well-aligned to human perception because hue linearity is one of its design goals. That is, a human observer is likely to perceive the same hue when given another sample with the same CAM hue but different brightness and/or chromatic content. *Hue quadrature* is a hue measure derived from hue where the values 0, 100, 200, and 300 correspond to the psychologically meaningful hues of red, yellow, green, and blue, respectively. *CAM Chroma* is the colorfulness of a stimulus as compared to the reference white, with 0 representing neutral colors. It is designed to be independent of lighting conditions. *CAM saturation* is the colorfulness of a stimulus as a proportion of its brightness. It is designed to be independent of the perceived brightness differences observable for different hues. The CIECAM02 Brightness (Q) and colorfulness (M) have not been included due to their strong dependency to the assumed viewing conditions. The viewing conditions are chosen based on the sRGB "typical" conditions and the guidance given in the CIECAM02 technical report (CIE 159:2004).

4.4 3D View

In the 3D View, the visualization designer can assess how the shape of a single color map behaves in different color spaces ($\mathbf{R}_4$). We take advantage of the fact that the RGB, CAM02-UCS (based on CIECAM02 as detailed above), HSB, and CIELAB color space can be spanned by three parameters. We provide 3D visualizations of the shape of a color map for every color space. These four visualizations are shown side by side in the 3D View. The shape of the color maps allows for an in-depth analysis of their properties. For example, a plane in the CieLab or CieCAM02 space indicates high perceptual linearity ($\mathbf{R}_6$).

To transform the color map into the different 3D color spaces, we first sample the color map at regular grid coordinates. The color at the sampling points is then converted into the different color spaces. The Lab conversion is achieved by assuming sRGB primaries and an E reference white source as appropriate for self-luminous displays. The exact conversion routines can be found in the corresponding literature, which is comprised of CIE Publications (Lab: ISO 11664-4:2008(E) / CIE S 014-4/E:2007; CIECAM02: Technical Report 159:2004), the HSB proposal [Smi78], and the Luo et al. CAM02-UCS proposal [LCL06]. All of them have three components, and most of them can be used directly as spatial coordinates in a 3D surface plot. In order to represent the hue and saturation values of the HSB color space as spatial coordinates in 3D, we apply a transformation into polar coordinates where hue denotes the angle and saturation the radius. Consequently, the color space is a cylinder, not a cube as in the other cases.

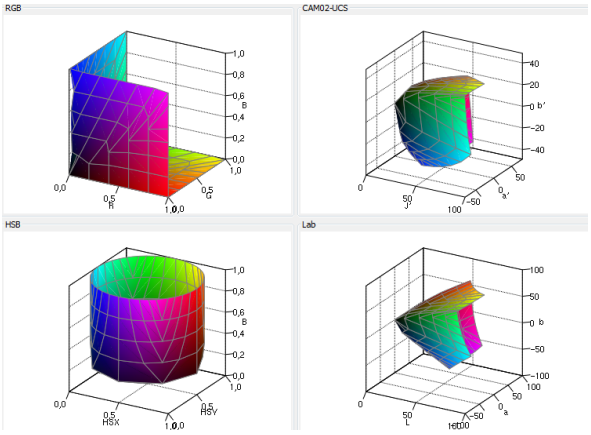


Figure 6: The 3D View of the map by Simula and Alhoniemi [SA99]: The color map is plotted in four different 3D spaces: the RGB and CIECAM02 cubes are at the top. The HSB space is actually a cylinder as the hue components defines a circle, the saturation its radius.

One of the benefits of the 3D View is that visualization designers are supported in the identification of the color space that was used for the design of the color map. As an example, many color maps are constructed as planar cuts through the RGB cube. An illustrative example based on the HSB color space is the map of Alhoniemi and Simula as shown in Figure 6. It covers the entire hue and brightness ranges at a constant saturation. This is why it appears as a cylinder in the HSB visualization. The individual 3D visualizations allow for interactive manipulation of the virtual camera. The designer can rotate the plot and adjust the axis scaling.

4.5 Color Sampling View

One of the most important quality aspects of a 2D color map is to faithfully represent spatial distances on the map with perceived color distances. This criterion is often called the perceptual linearity ($\mathbf{R}_5$). Another important aspect is the number of distinguishable colors ($\mathbf{R}_6$). The more colors a color map provides the more different information units can be encoded visually. We measure and illustrate the quality of both aspects in the Color Sampling View (see Figure 7).

Our approach first estimates the number of distinguishable colors based on the ΔE distance measure as described in Section 3. We solve an optimization problem trying to maximize the number of coordinates on the color map that fulfill the condition of a ΔE larger than a given threshold value t. The threshold is a user parameter and adjusts the minimum color distance in ΔE. A distance of $t = 1.0$ means that half of the observers are able to identify two colors as distinct. These points are depicted as white dots in Figure 7.

We compute the set of points using a circular sampling strategy: First, the center of the map is added to the result set. The algorithm then iterates on concentric circles around the center. Each of these circles is sampled in regular intervals. The number of sampling points on the circle increases with the radius of the circle increases to guarantee an equal sampling density. Once the set of points is defined, pair-wise distances are computed. For each sample point, the color distance to all points in result set is computed in ΔE. A point is added to the result set if the distance is always smaller than t. We note that is algorithm produces merely an approximation, but a valid lower bound for the number of points. Assuming that the approximation quality is similar for different maps, it also allows to compare the number of colors.

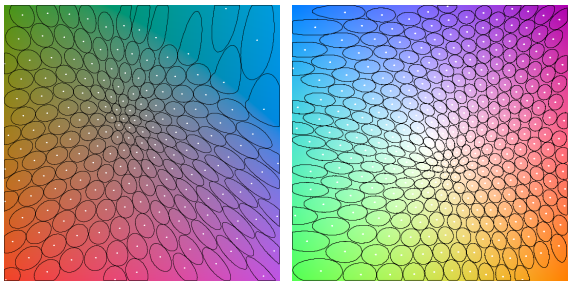


Figure 7: The Color Sampling View for the color maps of Bremm (left) and TeulingFig3 (right): white dots indicate centroids of distinguishable colors, black surrounding polygons are isolines similar to McAdam ellipses revealing information about local perceptual features.

In a second step, our approach estimates the perceptual linearity of the color map based on the regions with similar colors. Based on the set of points that define distinguishable colors, the areas with a distance of max. 1/2 t to the central point are approximated. Starting at the center, the algorithm samples along a set of straight line segments at different angles. For every line segment, we estimate the point where the threshold 1/2 t is crossed. These crossing points are connected to a iso-line polygon. The resulting polygon is similar to the MacAdam ellipse [Mac42]. Major differences are that MacAdam ellipses are defined in the xy-plane and are real ellipses based on the minimum and maximum ratio between geometric distance and perceived color distance (as measured by a human test person).

The Color Sampling View depicts the coordinates of the result set as white dots and the surrounding iso-lines as black polygons. The number of white dots indicates the number of distinguishable colors. The shape of an iso-line allows for an in-depth analysis of local perceptual features. Circular areas indicate a high local perceptual linearity, because the change of color is identical for all directions. On the contrary, distorted shapes indicate a varying local perceptual linearity. An example can be seen at the upper right of Figure 7 – left. While most shapes are rather circular, the upper right corner exhibits elliptical distortion. Individual divergences in shape can be identified easily by the user in this view.

The view also enables the visualization designer to compare different shapes. Variations in size indicate variances in the distribution of distinguishable colors. Thus, it can be seen that the perceptual linearity varies across the color map. In Figure 7, the color maps have 176 and 295 colors with a pair-wise distance of 5 ΔE. Interestingly, their distribution on the map is very different. In contrast, the map of Simula and Alhoniemi exhibits more than 600 colors.

4.6 Example Views

The usefulness of a color map for a visualization depends not only on intrinsic quality measures, but also on the usage context. Other visual environment parameters should be considered. With the example views, we support the visualization designer with test environments stressing color maps with other visual variables ($\mathbf{R}_7$).

Point Set Example

The first scenario illustrates the combination of the visual variables color (of the color map) and the position attribute. To that means, 100 equally-sized points with random colors are aligned at random positions in a point-based scatterplot. The test environment is shown in Figure 8. The visualization designer is enabled to assess the applicability of a color map for spatial object distributions. For the sake of comparability of different tests the randomization is deterministic.

Additional visual aspects of possible interest are the transparency of the colored points and the interplay of

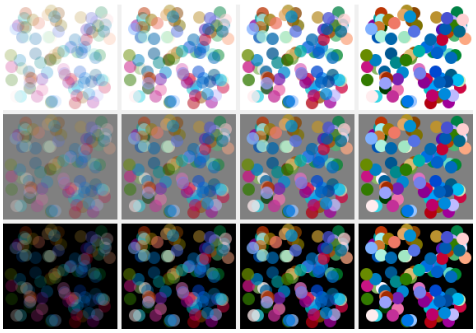


Figure 8: The Points Example View. A set of 100 points is plotted on different backgrounds with different levels of transparency. Overlapping circles are blended.

the color map with the background color. To this end, we utilize the small-multiples pattern and duplicate the test setup by means of a 4×3 juxtaposition. The 12 test setups differ in the color transparency level (25%, 50%, 75%, and 100% alpha channel) and the chosen background colors (white, gray, and black). This grid is depicted in Figure 8.

As a result, the user can immediately see whether the chosen color map has a significant distance to the background and to which extent transparency can be used.

Text Overlay View

An important requirement of many visualizations is that text must be legible. Therefore, the readability of fine visual structures (such as printed text) on colored background is illustrated in the Text Overlay View, as shown in Figure 9. Visualization designers are enabled to assess the readability of the provided text snippets in a qualitative way.

We use black, gray, and white as contrasting text colors. The text is printed at different sizes on a background that is generated from the color map. The two dynamic parameters (i.e. text color and font size) are varied in a small-multiples setup. The background of the test environments reflects the colors of the chosen color map.

Based on our experiments, the way the background is defined has a strong impact on the readability of the text. In particular, edges with sharp color contrast seem to distract the user's attention. To mitigate such effects, we use smooth (i.e. bilinear) interpolation of again pseudo-randomly selected color samples from the color map. This color is assigned to rectangles which are arranged in a two-dimensional grid in order to avoid irregular color changes. This view supports the user in comparing different environment variables in a text-based scenario.

5 USE CASE EXAMPLES

In this section, we demonstrate the usefulness of the ColorMap-Explorer. We show some of the findings that were made along an example analysis workflow.

Figure 9: The Text Overlay View. Text is printed at different sizes in different colors on space-filling background that is generated from the selected color map.

We conduct a scenario where a given color map is assumed to be ideal, be it on past experience or user preference. The visualization designer uses the ColorMap-Explorer to confirm this hypothesis. In this scenario, color should be used to encode information in a calendar-based visualization. Different results are depicted in Figure 10. The analytical task of the calendar view is mainly the comparison of individual (high-dimensional) data elements. Thus, the first important criterion is a large number of distinguishable colors to facilitate comparison tasks. The second criterion is perceptual linearity to adequately represent similarities of the data with color. Since the calendar grid is black, this color should be avoided.

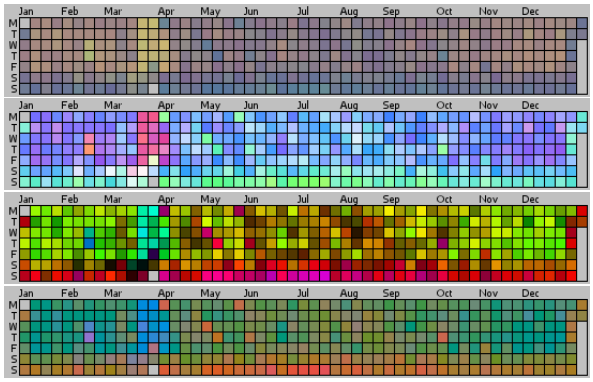


Figure 10: The calendar, rendered with different color maps. From top to bottom: BCP37, TeulingFig3, Simula and Alhoniemi, Bremm (regular).

The visualization designer uses the color map library that comes with the ColorMap-Explorer to experiment with different color maps. Some of the visualization drafts are depicted in Figure 10. While some of the visualizations are more colorful than others, it is unclear to what extend similarity of items from the original data set is preserved. She therefore starts the ColorMap-Explorer to find out which color maps are suited for this task.

After starting the tool, the overview panel comes up and the visualization designer can view all 22 color maps at a glance as shown in Figure 1 ($\mathbf{R}_1$). She realizes large discrepancies in the colorfulness of the different solutions. Her preference, Bremm et al. is about average compared to the others. Since candidate maps

should provide high color variations, the designer excludes maps such as BCP37 (the upper colormap in Figure 10) or Robertson and OCallaghan from further analysis due to their low colorfulness.

Based on the gained overview, the designer picks the most interesting color maps including the two variations of the color map of Bremm et al. [BvLBS11]. The Comparative View in Figure 3 shows them in juxtaposition ($\mathbf{R}_2$). Although the number of colors is about average in the regular map, it is superior in the other scoring categories. Bremm et al. (regular) has a higher score than the stretched version in most categories. As a consequence, the stretched version is excluded from the candidate set.

In a next step, the visualization designer continues to the Decomposition Panel with remaining candidates for detailed inspection. In Figure 4 (Teuling Fig2. [TSS11]), the second row shows that the red, green, and blue components increase across the map, but in different directions. The colormap exploits all three RGB channels yielding a large number of distinguishable colors. However, the top-left view shows large arrow vectors along the rising diagonal. This reveals that the perceived color varies strongly leading to an inhomogeneous perceptual linearity in general ($\mathbf{R}_3\mathbf{R}_6$). As a result, the designer rejects this color map for this task.

Aiming for high color exploitation, she picks a colorful map such as Simula and Alhoniemi [SA99] from the Overview Panel. The 3D view in Figure 6 confirms that the map covers large areas in the RGB and HSB color spaces ($\mathbf{R}_4\mathbf{R}_5\mathbf{R}_6$). However, in the CieLab and CIECAM spaces the shape divergences strongly from a plane indicating a lack of perceptual linearity. As a consequence, the visualization designer concludes that the similarity of data items is not preserved well enough.

Our visualization expert returns to Bremm (regular) and opens the Color Sampling View. As can be seen in Figure 7 (left), the largest part of the map comprises small and well-shaped ellipses. However, in the top right corner an anomaly can be identified. The black outlines are unproportionally large and distorted. The color variation in these regions is very low, leading to these large areas of similar color. Despite the fact that the map is mostly homogeneous, local features in the upper right corner hamper the homogeneity of the perceptual linearity ($\mathbf{R}_3$,$\mathbf{R}_6$). In contrast, in the TeulingFig3. map [TSS11] (Figure 7 – right) this deficiency is less prominent. Repeating this comparison with other colormaps reveals that (and why) TeulingFig3. scores well with respect to perceptual linearity.

In Figure 8, Guo's cone-shaped color map [GGMZ05] is plotted as a randomized Point Set Example. This color map has a very high color exploitation, which makes it easy to identify differently colored circles as such. However, some of the colors are very bright and hard to differentiate on white background, in particular at higher transparency levels ($\mathbf{R}_7$). It is therefore better suited on dark backgrounds.

Looking at the example in Figure 9, the ColorMap-Explorer reveals that black text is fairly easy to read with the TeulingFig3 color map [TSS11] – independent of the font size ($\mathbf{R}_7$). On the other hand, gray and white are not ideal. One possible explanation is the similar brightness of foreground and background. Since bright text is not used in the calendar, this is not critical.

The visualization designer concludes the decision support process of the use case. From the four initial drafts only Teuling Fig3. (the second color map in Figure 10) remains. BCP37 was rejected due to the limited number of distinguishable colors. Simula and Alhoniemi has a high color exploitation, but the 3D View revealed a lack of perceptual linearity. With the Color Sampling View, Bremm (regular) was ruled out in favor of Teuling Fig3. The map has high perceptual linearity and provides a fair color exploitation. The visualization designer therefore picks this map for the calendar view.

Having started with a set of preferred color maps, the ColorMap-Explorer enabled the visualization designer to reduce the number to a single colormap. While at the beginning of the process subjective criteria prevailed, the decision support system enriched the decision making with qualitative and quantitative means.

6 DISCUSSION & OUTLOOK

In this paper, we showcased the ColorMap-Explorer, a tool for the visual exploration of 2D color maps.

It gives an overview over many color maps that have been proposed in the literature on information visualization, provides different views for an in-detail analysis of strengths and weaknesses of color maps, and supports direct comparison, both visual and quantitative (i.e based on explicit quality measures). Each color map can be exported as a high-resolution image. This enables the data scientist not only to find the best fit for a given task, but also to directly re-use the color map in other visualization software tools.

Current color maps are general purpose and independent from the data set. Similar to the optimization approach, color maps can be tailored to specific data sets in order to achieve higher overall performance. The work of Mittelstädt et al. [MBS*14] already points in that direction. The integration of such customized color maps in the explorer could help fostering that research area.

One direction for future work is the adaption of existing color maps with respect to certain quality criteria. To the best of our knowledge, it has not yet been investigated if such auto-generated color maps can be superior to manually created ones.

7 REFERENCES

[Alb10] ALBRECHT M.: Color blindness. *Nature Methods 7* (2010), 775.

[BRT95] BERGMAN L., ROGOWITZ B., TREINISH L.: A rule-based tool for assisting colormap selection. In *Visualization* (Oct 1995), pp. 118–125, 444.

[BSM*15] BERNARD J., STEIGER M., MITTELSTÄDT S., THUM S., KEIM D. A., KOHLHAMMER J.: A survey and task-based quality assessment of static 2D color maps. In *To be published at the Conference on Visualization and Data Analysis* (2015).

[BSW*14] BERNARD J., STEIGER M., WIDMER S., LÜCKE-TIEKE H., MAY T., KOHLHAMMER J.: Visual-interactive Exploration of Interesting Multivariate Relations in Mixed Research Data Sets. *Comp. Graph. Forum 33*, 3 (2014), 291–300.

[BvLBS11] BREMM S., VON LANDESBERGER T., BERNARD J., SCHRECK T.: Assisted descriptor selection based on visual comparative data analysis. In *Computer Graphics Forum* (2011), vol. 30/3, Wiley Online Library, pp. 891–900.

[CYG04] CAMGÖZ N., YENER C., GÜVENC D.: Effects of hue, saturation, and brightness: Part 2: Attention. *Color Research & Application 29*, 1 (2004), 20–28.

[GGMZ05] GUO D., GAHEGAN M., MACEACHREN A. M., ZHOU B.: Multivariate analysis and geovisualization with an integrated geographic knowledge discovery approach. *Cartography and Geographic IS 32*, 2 (2005), 113–132.

[HB03] HARROWER M., BREWER C. A.: ColorBrewer. org: An online tool for selecting colour schemes for maps. *The Cartographic Journal 40*, 1 (2003), 27–37.

[Hea96] HEALEY C. G.: Choosing effective colours for data visualization. In *Visualization* (1996), IEEE, pp. 263–270.

[Him98] HIMBERG J.: Enhancing the som based data visualization by linking different data projections. In *Proceedings of 1st International Symposium IDEAL* (1998), vol. 98, pp. 427–434.

[LCL06] LUO M. R., CUI G., LI C.: Uniform colour spaces based on CIECAM02 colour appearance model. *Color Research & Application 31*, 4 (2006), 320–330.

[Mac42] MACADAM D. L.: Visual sensitivities to color differences in daylight. *Journal of the Optical Society of America 32*, 5 (May 1942), 247–273.

[MBS*14] MITTELSTÄDT S., BERNARD J., SCHRECK T., STEIGER M., KOHLHAMMER J., KEIM D. A.: Revisiting Perceptually Optimized Color Mapping for High-Dimensional Data Analysis. In *In Proceedings of the Eurographics Conference on Visualization (EuroVis 2014, Short Paper)* (2014).

[MFH*02] MORONEY N., FAIRCHILD M. D., HUNT R. W., LI C., LUO M. R., NEWMAN T.: The CIECAM02 color appearance model. In *Proceedings of the Color and Imaging Conference* (2002), vol. 2002, pp. 23–27.

[Rhe00] RHEINGANS P.: Task-based color scale design. In *28th AIPR Workshop: 3D Visualization for Data Exploration and Decision Making* (2000), International Society for Optics and Photonics.

[RO86] ROBERTSON P. K., O'CALLAGHAN J. F.: The generation of color sequences for univariate and bivariate mapping. *Computer Graphics and Applications, IEEE 6*, 2 (1986), 24–32.

[RTB96] ROGOWITZ B., TREINISH L., BRYSON S.: How not to lie with visualization. *Computers in Physics 10*, 3 (1996).

[SA99] SIMULA O., ALHONIEMI E.: Som based analysis of pulping process data. In *Engineering Applications of Bio-Inspired Artificial Neural Networks*. Springer, 1999, pp. 567–577.

[SBM*14] STEIGER M., BERNARD J., MITTELSTÄDT S., LÜCKE-TIEKE H., KEIM D., MAY T., KOHLHAMMER J.: Visual analysis of time-series similarities for anomaly detection in sensor networks. *Computer Graphics Forum 33*, 3 (2014), 401–410.

[Smi78] SMITH A. R.: Color gamut transform pairs. In *ACM Siggraph Computer Graphics* (1978), vol. 12, ACM, pp. 12–19.

[SvLB10] SCHRECK T., VON LANDESBERGER T., BREMM S.: Techniques for precision-based visual analysis of projected data. *Information Visualization 9*, 3 (2010), 181–193.

[TFS08] TOMINSKI C., FUCHS G., SCHUMANN H.: Task-driven color coding. In *Information Visualisation, 2008. IV'08. 12th International Conference* (2008), IEEE, pp. 373–380.

[TSS11] TEULING A. J., STÖCKLI R., SENEVIRATNE S. I.: Bivariate colour maps for visualizing climate data. *International Journal of Climatology 31*, 9 (2011), 1408–1412.

[War88] WARE C.: Color sequences for univariate maps: Theory, experiments and principles. *IEEE Computer Graphics and Applications 8*, 5 (1988).

[WB88] WARE C., BEATTY J. C.: Using color dimensions to display data dimensions. *Human Factors: The Journal of the Human Factors and Ergonomics Society 30*, 2 (1988), 127–142.

[WF80] WAINER H., FRANCOLINI C. M.: An empirical inquiry concerning human understanding of two-variable color maps. *The American Statistician 34*, 2 (1980), 81–93.

Pose-Specific Pedestrian Classification using Multiple Features in Far-Infrared Images

Dong-Seok Kim
Center for Integrated Smart Sensors
312 IT Convergence Center(N1)
291 Daehak-ro, Yuseong-gu
305-701, Daejeon, Korea
kds1130@kaist.ac.kr

Ki-Yeong Park
Center for Integrated Smart Sensors
312 IT Convergence Center(N1)
291 Daehak-ro, Yuseong-gu
305-701, Daejeon, Korea
cpky0@kaist.ac.kr

ABSTRACT

We present a multiple feature-based, pose-specific pedestrian classification approach to improve classification performance for far-infrared (FIR) images. Using pose-specific classifiers and multiple features has proved to be beneficial in visible-spectrum-based classification systems; therefore, we adapt both to an FIR-based classification system. For pose-specific classifiers, we separate poses into sets of front/back and right/left poses and estimate the pose using template matching. For feature extraction, we use histograms of local intensity differences (HLID) and local binary patterns (LBP). Experiments showed that the proposed approaches improve the classification performance of a baseline HLID/linSVM approach.

Keywords

Pedestrian classification, Multi-features, Pose templates, Far-infrared images

1. INTRODUCTION

Far-infrared (FIR) cameras (or thermal cameras), capture the heat emitted from objects, so pedestrians typically appear brighter than backgrounds in FIR images (see Fig. 1). Thus, FIR technology is advantageous for pedestrian detection at nighttime. Because of this characteristic, a major subject of previous studies has been candidate generation processes for finding regions in images that are highly likely to contain a pedestrian. For classification, most studies have simply used single feature-based classifiers, such as histograms of oriented gradients (HOG) [1, 2]. For pedestrian classification in visible spectrum images, it has been shown that an ensemble of classifiers that has been trained for particular pedestrian poses outperforms a single classifier that has been trained for the entire data set [3]. It has also been shown that methods that use a combination of multiple features are able to improve the classification performance as compared to methods that only use a single feature [3].

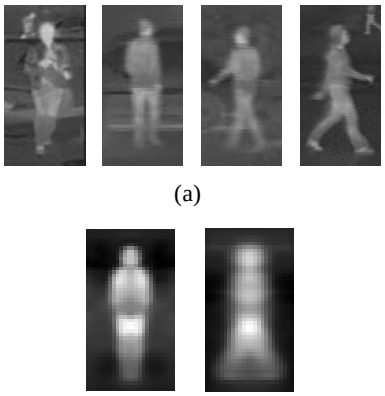
Despite the benefits of using pose-specific classifiers and multiple features that are expected in FIR images, only a few studies have dealt specifically with this subject. In [4], appearance-based classifiers, which consisted of "along-street," "across-street," and "bicyclist," were utilized and decisions were simply based on the logical OR process for all classifiers. However, the combination of these classifiers risks generating numerous false positives. In [5], multiple features were investigated, but decisions were based on the classification result using only one of the features after the feature selection process, instead of fusing the multiple features. Thus, it is difficult to expect better performance from this method as compared to that from single feature-based classification methods.

In this paper, we investigate the benefit of using pose-specific classifiers and multiple features in FIR images. For pose-specific classifiers, pose estimation is performed simply by correlation with pose

(a) (b)

Figure 1. Sample images of pedestrians. (a) Visible spectrum image. (b) Far-infrared image.

 ISBN 978-80-86943-65-7



(a)

(b)

Figure 2. Pedestrian example images. (a) Samples of pose variations of front, back, left, and right. (b) Templates of front/back and left/right

templates. For feature extraction, we introduce a novel combination of HLID [6] and cell-structured LBP [7] features. We compare the proposed approach to the HLID/linSVM classifier approach because the motivation of this study is to gain performance improvements over our previous work that used an FIR-specified feature [6].

2. PEDESTRIAN CLASSIFICATION

To improve classification performance, we adopt pose-specific classifiers and multiple features to FIR-based pedestrian classification. For the pose model, only two poses of front/back and right/left are considered instead of four poses (front, back, right, and left), as is the case in visible spectrum image-based pedestrian classification because of the similar contours between the combined poses. Furthermore, it makes the classification problem simple and reduces the computational power. Fig. 2 shows the samples as pose variations and the pose templates that are generated by averaging the intensity of manually separated positive samples of poses from training sets. Regarding features, we chose HLID features and cell-structured LBP features. HLID was selected because it has been shown to outperform HOG in FIR-based classifications [6]. Further, LBP was selected because it was expected to compensate the problems (sensitivity to noisy background edges) of HLID using its uniformity constraints [7].

For combining information from multiple poses and multiple features, we employed a mixture-of-experts (MoE) framework introduced in [3]. In the MoE approach, the posterior probability that a given sample (x_i) is a pedestrian class $((\omega_0))$ is $(P(\omega_0|x_i))$, which is approximated with a sample-dependent weight $w_k(x_i)$ and a pose-specific classifier output $H_k(x_i)$ with pose clusters k as

$$P(\omega_0|x_i) \approx \sum_k w_k(x_i) H_k(x_i) \qquad (1).$$

Given the pose-specific MoE model, the pose-specific expert classifier $H_k(x_i)$ was modeled in terms of our multiple feature set (f) as

$$H_k(x_i) = \sum_f v_k^f I_k^f\left(x_i^f\right) \qquad (2).$$

Here, $I_k^f(x_i^f)$ denotes a local expert classifier for the kth pose cluster with features f from a feature set, and v_k^f represents a pose and feature dependent weight with $\sum_f v_k^f = 1$. For expert classifiers I_k^f, we used a linear support vector machine (linSVM) to train the classifiers from the corresponding pose and feature only. Given K (2 of front/back and right/left) pose clusters and F (2 of HLID and LBP) features, we trained $K \times F$ classifiers I_k^f on the pose-specific training set. Weights v_k^f were used to model the contribution of the individual classifiers. Hence, we derived the weights by the discriminative power of the individual expert classifiers using a training dataset. The sample-dependent weight $w_k(x_i)$ was decided using similarity between pose templates t_k and the sample x_i as

$$\omega_k(x_i) = \frac{corr(x_i, t_k)}{\sum_k corr(x_i, t_k)} \qquad (3).$$

To measure the similarity, we used simple template matching using Pearson's correlation measures. Both templates and samples were normalized before matching. For the weight function, the weights of sample outputs of pose-specific classifiers were determined proportionally by their similarity to the pose template with $0 \le w_k(x_i) \le 1$ and $\sum_k w_k(x_i) = 1$. Using the weight function, this method can lower the risk of degradations in the classification process that are caused by incorrect pose decisions.

3. EXPERIMENTAL RESULTS

The proposed method was evaluated using 6573 FIR images that were taken from moving vehicles in an urban area at nighttime. We split the set of images into training sets and test sets according to the days of images captured: 4668 images for training and 1905 images for test. The training samples were cropped from the training set and then divided into two different pose sets of front/back and right/left. Test samples were cropped automatically from the test set using the sliding window technique based on the overlap ratio between the current window and the manually labeled pedestrian ground truth (we choose the current window as test sample when the overlap ratio exceeds over 70%). Table 1 gives an overview of the dataset. Samples were resized to 24 × 48 pixels.

	Pedestrians (front/back)	Pedestrians (right/left)	Non-pedestrian
Training set	2209	1879	8555
Test set	13123		10173

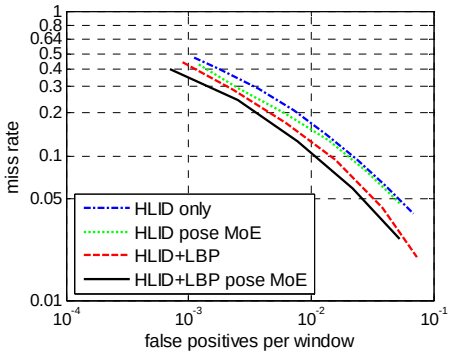
Table 1. Datasets for evaluation

We computed $HLID_{8,2}$ features using a cell size of 6 × 6 pixels, block size of 2 × 2 cells, overlap of 0.5 blocks, and *L2-norm* block normalization. To extract LBP features, we computed $LBP_{8,2}$ features using a cell size of 8 × 8 pixels and a maximum transition number of 2. Expert weights v_k^f were estimated by a linSVM on the training set (0.52 for HLID and 0.48 for LBP for both poses). To quantify the performance, we plotted the detection error tradeoff (DET) curves on a log-log scale on both a per-window and per-image evaluation. We followed the evaluation method used in [8].

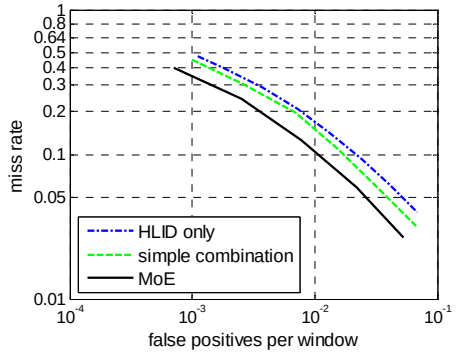
First, we compared our proposed pose-specific and multiple feature-based classification approach to the single feature-based classifier, pose-specific classifier, and multiple feature-based classifier approaches. We selected HLID as the baseline feature to show the performance improvements over the FIR-specified feature. Multiple feature-based classification was conducted by concatenating two features into a single feature vector. The results are shown in Fig. 3(a). As expected, the proposed approach that combines pose-specific and multiple feature-based classifiers outperformed other approaches. We also found that both the pose-specific classifier approach and the multiple feature-based classifier approach outperformed the single feature-based classifier. The results confirm that the pose-specific classifier approach performs better because the pose variations are relatively smaller than the classifier trained on a whole dataset irrespective of pose. Further, the combination of multiple complementary features boosts the performance.

Next, we compared the MoE framework to simple combination rules to see the performance if the fusion method is varied. For simple combination rules, the concatenated multiple features were classified based only on the selected pose-specific classifier of having maximum pose similarity. Fig. 3(b) shows that the MoE approach outperforms the simple combination approach. The differences were mainly caused by errors in pose estimation and by the use of the same weights for features without consideration of the discriminative power of each feature.

Finally, we evaluated our proposed method on a per-image basis to compare with the single feature-based classifier described in [6], and checked for



(a)



(b)

Figure 3. Performance comparison results with per-window evaluations. (a) Performance based on information of classification variations. (b) Performance based on fusion method variations.

improvements for pedestrian detection in FIR image sequences. Except for the classification method, all of the other procedures and evaluation methods are the same. Fig. 4 shows that our proposed method improves the pedestrian detection performance by reducing the miss rate by approximately 4% at 10^{-1} false positive per image (FPPI). These results

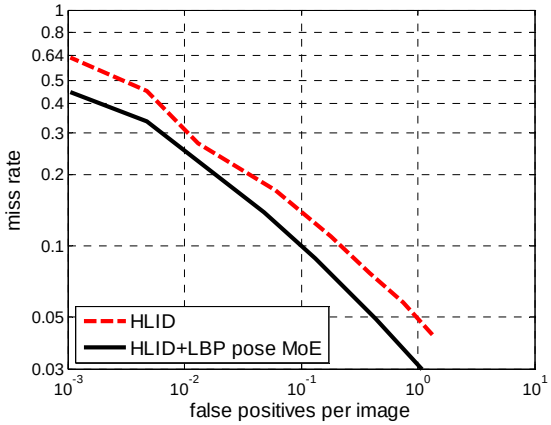


Figure 4. Pedestrian detection performance comparison between the proposed classifier and the baseline HLID/linSVM classifier.

 ISBN 978-80-86943-65-7

Figure 5. Detection examples of representative scenarios of single or multiple pedestrians with pose variations (the red boxes indicate the detection results).

demonstrate the benefit of our proposed method. In order to gain more performance improvements, it will be necessary to upgrade pose estimation accuracy. This will be investigated in a future work. Fig. 5 shows some detection examples in FIR images.

4. CONCLUSION

We proposed a pose-specific pedestrian classification using multiple features in FIR images. Experiments showed the proposed approaches outperform single feature-based classifier. Reducing pose variation is helpful for FIR-based pedestrian classification. Further, the newly introduced combination of HLID and LBP features proved to be beneficial. We hope that our results will help promote further research on classifiers in FIR-based pedestrian detection systems.

5. ACKNOWLEDGMENTS

This work was supported by the Center for Integrated Smart Sensors funded by the Ministry of Education, Science and Technology as Global Frontier Project (CISS-2011-0031863).

6. REFERENCES

[1] O'Malley, R. Jones, E. Glavin, M.: 'Detection of pedestrians in far-infrared automotive night vision using region-growing and clothing distortion compensation', I P & T, Vol.53, pp.439-449, Nov 2010.

[2] Dalal, N. Triggs, B.: 'Histograms of oriented gradients for human detection'. IEEE Conf., CVPR. 2005, Vol.2, pp.886–893, June 2005.

[3] Enzweiler, M. Gavrila, D, M.: 'A Multilevel Mixture-of-Experts Framework for Pedestrian Classification,' Image Processing, IEEE Trans., Vol.20, pp.2967-2979, Oct 2011.

[4] Fengliang, X. Xia, L. Fujimura, K.: 'Pedestrian detection and tracking with night vision,' ITS, IEEE Trans., Vol.6, pp.63-71, 2005.

[5] Li, Z. Bo, W. Nevatia, R.: 'Pedestrian Detection in Infrared Images based on Local Shape Features,' CVPR., IEEE Conf., pp.17-22, 2007.

[6] Kim, D, S. Kim, M. Kim, B, S. Lee, K, H.: 'Histograms of local intensity differences for pedestrian classification in far-infrared images,' Electronics Letters, Vol. 49, issue. 4, pp. 258-260, 2013.

[7] Wang, X. Han, T. X. Yan, S.: 'An HOG-LBP human detector with partial occlusion handling', CV, IEEE Conf., pp.32-39, 2009.

[8] Dollar, P. Wojek, C. Schiele, B. and Perona, P.: 'Pedestrian Detection: An Evaluation of the State of the Art,' Pattern Analysis and Machine Intelligence, IEEE Trans., Vol. 34, No. 4, pp. 743-761, Apr. 2012.

Projector-Leap Motion calibration for gestural interfaces

Pavel Najman Jiri Zahradka Pavel Zemcik

Faculty of Information Technology
Brno University of Technology
Bozetechova 1/2, Brno
Czech Republic
{inajman, izahradka, zemcik}@fit.vutbr.cz

ABSTRACT

Calibration of a projector and a tracking device is an essential step for interaction with projected content using gestures. We propose a novel technique for calibration of a data projector and a Leap Motion sensor. Using the proposed approach, users can calibrate the system by touching few points on the screen plane and in the space above it. No printed patterns, reflective markers, or additional tools are needed. The calibration process involves two steps. In the first step, we collect finger positions which we then use in the second step to find the calibration matrix and projector position. We compared the accuracy and precision of the proposed method to the accuracy and precision of a capacitive touchscreen in a touch based interaction task. During the evaluation we measured the Euclidean distance between the displayed and touched points. The best average distance for our method was 1.23 mm which is comparable to 0.79 mm for touch screen. The experiments demonstrate that the technique is suitable for an interaction with user interface elements designed in the usual way.

Keywords

projector, Leap Motion, hand tracking, calibration, dynamic mapping, HCI

1 INTRODUCTION

Human computer interfaces research in the last years becomes more and more focused on 3D gestural interfaces. 3D gestural interaction is natural and powerful method of communication between humans and computers [10]. The popularity of the gestural interfaces is supported by increasing number of different body and hand tracking devices, such as Microsoft Kinect, Asus Xtion or Leap Motion Controller. These sensors are relatively cheap and therefore available for wide range of standard computer users. In comparison to traditional keyboard and mouse interfaces, the gestural interfaces bring novel interaction techniques, that might be beneficial for many applications but especially for games or applications on tabletop displays.

The tabletop display interfaces are often equipped with capacitive touchscreens, which are quite expensive due to their large size [15, 7]. An alternative to touchscreen is a combination of a data projector and a motion tracking sensor. This approach has several advantages over the touchscreen solution. The image might be projected on various surfaces with different shapes and reflection properties. Unlike touchscreens, these surfaces are usually more scratch and fingerprint resistant and can be cleaned more easily. We also have additional information about the types and poses of fingers. This information comes from a sensor that detects and tracks user's hands and fingers during interaction. The detected poses are transformed into the projector image space, which allows the projected image modification to provide a feedback to the user. This approach brings the necessity of calibration, in order to obtain the spatial transformation between the projector's image coordinates and the sensor's space coordinates.

In this paper we propose a calibration technique for a data projector and the Leap Motion Controller, which might be applied to the tabletop interfaces. A unique contribution of our approach is that the technique doesn't require printed patterns, reflective markers, or other additional tools. The effective range of the Leap Motion extends from approximately 25 to 600 millimetres and its field of view is about 150 degrees. The sensor utilizes two stereoscopic IR cameras and three IR light emitting diodes for fast and accurate hand tracking [11]. The high accuracy (up to 0.01 mm) and the low latency (less than 10 ms) are the main reasons why we chose to use Leap Motion sensor instead of other body tracking sensors that are available.

The further parts of this paper are organized as follows. The following Related Work briefly reviews the calibration techniques of a data projector with a standard

camera, an infrared camera and depth sensors. The Proposed Method section defines a novel technique for the calibration of a data projector and the Leap Motion Controller. The section contains also a description of our experimental environment. The accuracy of the proposed method was evaluated with two experimental setups and compared to a capacitive touchscreen accuracy. All experiments and results are presented in Evaluation section. Finally, Conclusion section summarizes the paper contribution.

2 RELATED WORK

Projector-based user interfaces often utilize a standard camera as a simplest way to enable user interaction with the projected content. In these interfaces, the camera observes the projected image. The calibration of such systems is usually done semi-automatically by projection of structured light pattern over a printed pattern of a known size. The overall scene is captured by a camera [4, 3]. Although it is easy to find the calibration points, it is difficult to track hands or fingers because conventional skin color detection algorithms fail in the recognition of the hand overlapped by the projected image. Thus, the combination of a data projector and the standard camera is more suitable for structured-light scanners [17].

The problem with the image overlapping of the hands and fingers disappear, when the standard camera is replaced with an infrared (IR) camera. The visible light projected by the data projector is invisible for the IR camera and therefore the calibration point's measurement process must be different. The projected points might be marked by IR markers or user has to touch the points with the finger. Both approaches were presented in [19, 20], where the calibration was calculated just as a 2D homography between the camera and the projector image planes. The spatial relationship between a world and a projector coordinates was not required in these setups.

The above described difficulties might be overcome when the projector-based interface is equipped with a RGB-D devices such as the Asus Xtion[2] or the Microsoft Kinect. These devices are based on PrimeSense technology[8, 12], which combines an RGB camera with a depth sensor made using an IR camera and IR projector. A drawback of these devices is in their high latency and low depth accuracy, which has relative error between few millimeters and 40 mm [9]. The PrimeSense technology is used in combination with a pico-projector in the wearable interface for multi-touch applications on everyday surface called the OmniTouch[6]. The calibration of a data projector and Kinect is utilized in a tabletop multi-touch display system presented at [18]. This system is able not only to detect if the screen is being touched, but can also tell which finger was used to touch the surface.

Unlike the described previous works, our technique does not require printed patterns, reflective markers, or other additional tools. Also the hand tracking speed is far better, since we are using the Leap Motion controller. This controller has hand tracking latency less than 10 ms. For comparison, the Microsoft Kinect has latency around 60 ms [14].

3 PROPOSED METHOD

In this section, the proposed calibration method is presented. The calibration process involves two steps. The inputs to the first step are coordinates of points that will be projected onto the screen plane. The goal of the first step is to collect points in Leap Motion coordinate space that correspond to the projected points in screen coordinate space. The outputs of the first step are sets of points. Each set contains a point in screen coordinate space and several corresponding points in the Leap Motion coordinate space. These sets are then processed in the second step which computes the transformation matrix and projector position. The whole process is shown in Figure 1.

The first step begins with the projection of the calibration pattern, that contains at least three non-collinear points, onto the screen plane. The first pattern point is highlighted and the user positions his hand so that the tracked finger is touching the currently highlighted point on the screen surface. The position of the finger is captured and the same point is highlighted with different color. User then raises his hand from the surface and moves it towards the projector along the projection ray. During this motion the user keeps the projected point on his finger. When the hand movement stops, the spatial position of the finger is captured again. The height, in which the user stops above the screen plane, is not important and it can be different for each point. At this point, we can either highlight the same point with different color and continue by collecting more finger positions along the projection ray or highlight next pattern point and repeat the collection process.

We use the acceleration of finger motion to identify the appropriate moment for capturing the finger position. During the collection process the tracked finger moves either fast or slowly. The fast motion occurs when the user moves between two point positions and the slow motion usually means that the user stops to touch the highlighted point. To distinguish between the fast and slow motion, we utilize two thresholds. When the finger speed drops below the lower threshold we capture the current finger position. Afterwards we wait for the speed to raise above the upper threshold before we capture another point. The thresholds values reflect the registered speed of motion when the hand is still and when the hand moves between two points. Therefore, the lower threshold value depends to a large extent on

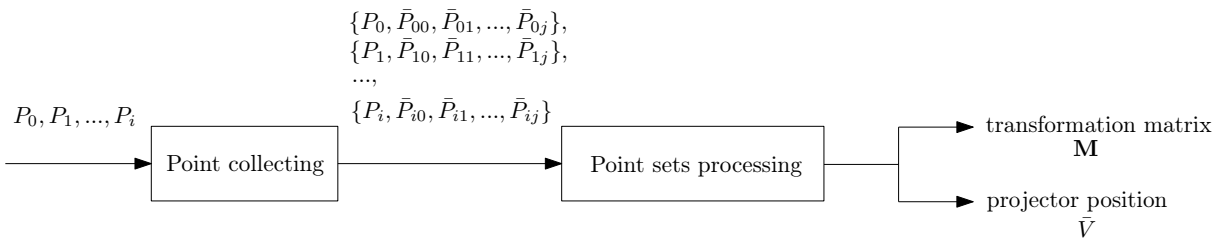


Figure 1: Calibration process involves two steps. The inputs to the first step are at least three non-collinear points that are projected. In the first step, points in Leap Motion coordinate space, that correspond to the projected points, are collected. The collected points, together with the projected points, are processed in the second step. The outputs of the calibration process are the transformation matrix and the projector position.

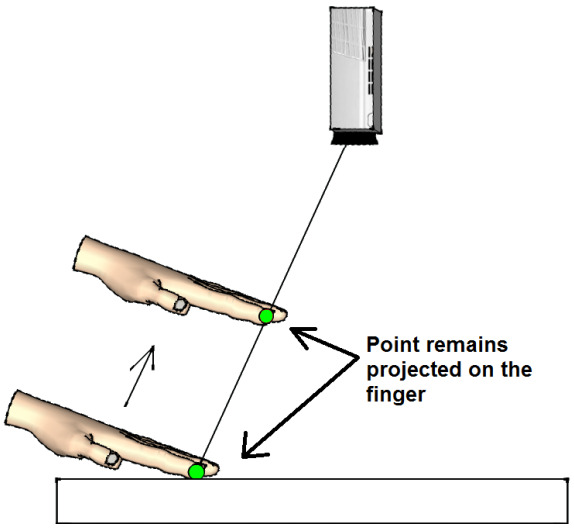


Figure 2: To capture the calibration points above the screen plane, the user moves his hand along the projection ray while keeping the projected point on his finger.

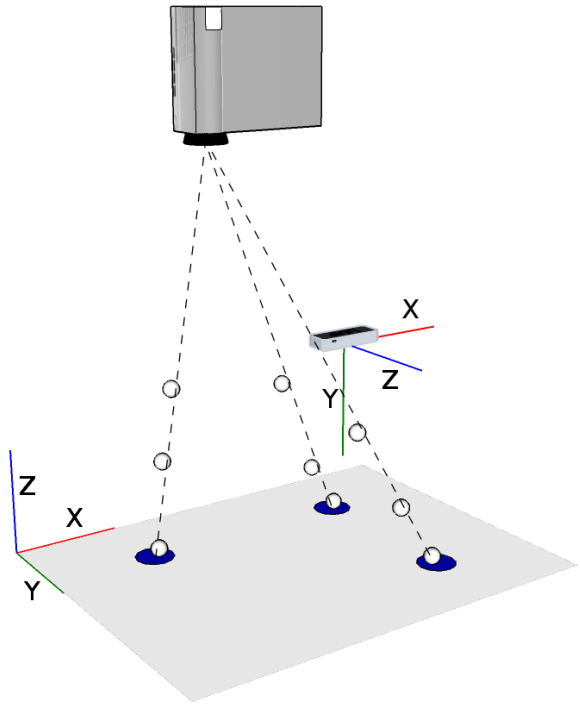


Figure 3: Input data measurements for proposed calibration technique. The white spheres represent collected finger positions in the Leap Motion coordinate space. The corresponding pattern points in the screen coordinate space are highlighted by blue circles.

the tracker noise and the natural hand tremor and the value of the upper threshold depends mainly on the distance between two pattern points. We use 1.5 mm/s and 75 mm/s as the values for lower and upper threshold respectively, which we found suitable for our environment. Alternatively, pressing of a button or timer based approach can be used for point capturing.

For each pattern point, we collect three finger positions, one on the screen surface and two above. The collected finger positions in Leap Motion coordinate space and corresponding point coordinates in projector screen space are processed in the next step. The situation is depicted in Figure 3.

In the second step, the projector position relative to Leap Motion and the transformation matrix is computed. The relationship between point $\bar{P} = [\bar{p}_x, \bar{p}_y, \bar{p}_z, 1]$ in the Leap Motion coordinate space and the corresponding point $P = [p_x, p_y, p_z, 1]$ in screen coordinate space is given by

$$P = \mathbf{M}\bar{P} \tag{1}$$

where $\mathbf{M}$ is the transformation matrix that can be further decomposed to

$$\mathbf{M} = \mathbf{SRT} \tag{2}$$

scale, rotation and translation matrices. There are eight parameters that need to be found in order to construct

 ISBN 978-80-86943-65-7

these matrices. These parameters are two scales s_x, s_y , three angles of rotation α, β, γ and three elements of a translation vector $\mathbf{t}$. We find their values by first computing their initial estimates which we then refine using Levenberg-Marquardt method [16].

To compute the initial estimates we need three non-collinear points P_0, P_1 and P_2 in screen coordinates and their counterparts $\bar{P}_{00}$, $\bar{P}_{10}$ and $\bar{P}_{20}$ in Leap Motion coordinate space captured on the screen plane. From these points two vectors in each coordinate space can be computed.

$$\mathbf{u} = P_1 - P_0 \qquad \bar{\mathbf{u}} = \bar{P}_{10} - \bar{P}_{00} \tag{3}$$

$$\mathbf{v} = P_2 - P_0 \qquad \bar{\mathbf{v}} = \bar{P}_{20} - \bar{P}_{00} \tag{4}$$

To construct the scale matrix $\mathbf{S}$ we need two scales s_x and s_y which represent the number of pixels per mm in x and y direction. Only one estimate

$$s = \frac{\|\mathbf{u}\|}{\|\bar{\mathbf{u}}\|} \tag{5}$$

can be used for both values because the difference between these two values is usually very small and will be compensated for in the following refinement. For the third scale s_z, we use fixed value of 1. By doing this the z coordinate of the point P will represent the height above the screen in mm.

For the rotation matrix $\mathbf{R_{xyz}}$, three angles are needed

$$\alpha = -\arctan 2\left(\frac{\bar{\mathbf{y}}\cdot\mathbf{z}}{\cos\beta}, \frac{\bar{\mathbf{z}}\cdot\mathbf{z}}{\cos\beta}\right) \tag{6}$$

$$\beta = -\arcsin\left(\bar{\mathbf{x}}\cdot\mathbf{z}\right) \tag{7}$$

$$\gamma = -\arctan 2\left(\frac{\bar{\mathbf{x}}\cdot\mathbf{y}}{\cos\beta}, \frac{\bar{\mathbf{x}}\cdot\mathbf{x}}{\cos\beta}\right) \tag{8}$$

which express the rotation around x, y and z axis respectively. Vectors $\mathbf{x} = [1,0,0,0]$, $\mathbf{y} = [0,1,0,0]$ and $\mathbf{z} = [0,0,1,0]$ are unit vectors that represent the orientations and the directions of the screen coordinate system. Vectors $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$ and $\bar{\mathbf{z}}$ are unit vectors that represent the same orientations and directions but in the Leap Motion coordinate space. Vector $\bar{\mathbf{z}}$ is the normal to the plane specified by $\bar{\mathbf{u}}$ and $\bar{\mathbf{v}}$. Vector $\bar{\mathbf{x}}$ can be found by first finding the angle between $\mathbf{u}$ and $\mathbf{x}$ and then rotating the $\bar{\mathbf{u}}$ vector by the same angle around the $\bar{\mathbf{z}}$ axis. Vector $\bar{\mathbf{y}}$ can be found analogously.

The initial estimate for vector of translation is given by

$$\mathbf{t} = -\left(\bar{P}_{00} - \frac{p_{00x}}{s}\bar{\mathbf{x}} - \frac{p_{00y}}{s}\bar{\mathbf{y}}\right) \tag{9}$$

To refine the initial estimates, we minimize

$$E_1 = \sum_{n=0}^{i} \|P_n - \mathbf{M}\bar{P}_{n0}\|^2 \tag{10}$$

using Levenberg-Marquardt method, as implemented in MPFIT [13] which contains a translation of MINPACK [16] algorithms to C. We use all the points $\bar{P}_i$ in the Leap Motion space that were captured on the screen plane together with their counterparts P_i in the screen space for this refinement.

To find the projector position, we first need to reconstruct the projection rays from the collected points. This can be done by fitting a straight lines through the points that were captured for the same pattern point in different heights above the screen plane. To fit the straight line $\bar{\mathbf{l}}$ through the points that were captured for the ith pattern point, we minimize

$$E_2 = \sum_{n=0}^{j} dist(\bar{P}_{in}, \bar{\mathbf{l}})^2 \tag{11}$$

the sum of square distances from these points to the line.

In an ideal situation, the projector would be positioned in the intersection of these lines. Because the lines are almost always skew with no intersections, we rather find the point of closest approach $\bar{V}$ for all lines. This point can be found by minimizing

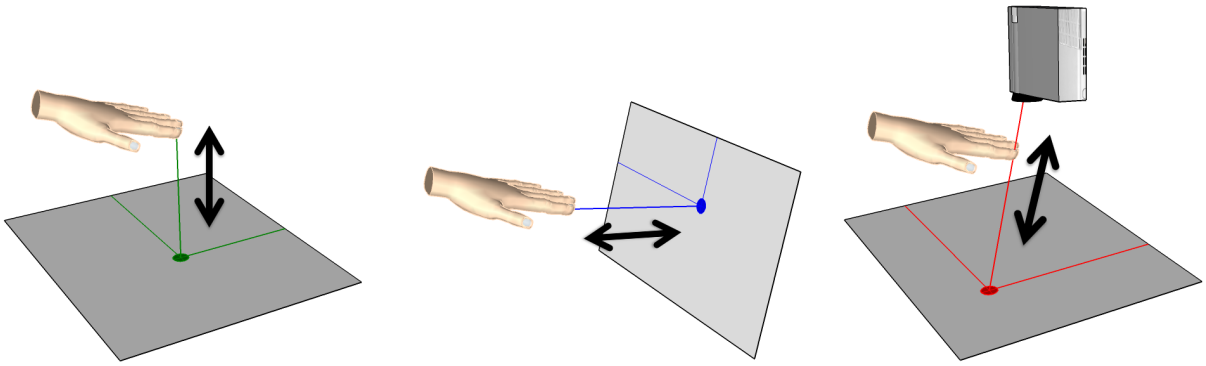
$$E_3 = \sum_{n=0}^{i} dist(\bar{V}, \bar{\mathbf{l}_\mathbf{n}})^2 \tag{12}$$

the sum of square distances from the point to all the lines.

Several ways exist in which we can use the results of the calibration. We present three of them that could be the most common ones. They are touch based interaction, direct pointing interaction, and dynamic finger/hand projection mapping. The first two are interaction styles in which we need to find the point on the screen with which we are trying to interact. The last one uses the results of calibration to project an image on user's hand or fingers that can contain additional information.

For touch based interaction, the point of interest is found by simply multiplying the finger position with the calibration matrix. The result is a point with x and y coordinates that represent the position on the screen in px. The z coordinate represents the height of the finger above the screen in mm which can be used to find out if the finger is touching the screen or not. See Figure 4a.

For direct pointing interaction we need to transform not only the finger position but also the direction in which the finger is pointing. These two pieces of information specify a line that is then intersected with a plane which is defined by a point $A = [0,0,0]$ and normal $\mathbf{n} = [0,0,1]$. By intersecting the transformed line with that plane, we obtain the position on the screen at which we are pointing. See Figure 4b.



(a) Touch interaction (b) Direct pointing interaction (c) Dynamic mapping

Figure 4: The results of the calibration can be used in several ways, for example - touch based interaction, direct pointing interaction, or dynamic hand/finger projection mapping.

To project the image on user's hand, we need to find the point on the screen plane that lies on the line connecting the projector position and user's hand. This line can be defined by a point on the line and direction vector. As a point on the line, we can use the projector position and the direction vector is given by subtracting the projector and hand positions. This line is then intersected with the same plane as above. See Figure 4c.

4 EVALUATION

In this section, the accuracy of the proposed calibration method is compared to the accuracy of the capacitive touch display in the touch based interaction task.

The task involved touching the displayed points as close to their center as possible. The points that had to be touched were arranged in a regular five by five testing pattern. The testing pattern is shown in Figure 6. Each point was touched five times for one calibration attempt. Five calibration attempts for each of the three calibration cases, were performed. These calibration cases differed in the number of points (3, 9 and 16) in their calibration patterns. This process was repeated twice, once on a small screen and once on a screen with larger dimensions. The small screen was the screen of the capacitive touch display and the large screen was projected screen. The diagonal of the touch display was 256 mm and the screen resolution was 1368x768 px. The diagonal of the projected screen was 443 mm with resolution of 800x600 px. On the capacitive touchscreen, we performed the same task, but since there was no way to recalibrate the touchscreen we touched each point five times in five attempts without recalibration. No measurements for the capacitive touchscreen with large size were recorded since we were unable to find the screen with larger diagonal than 256 mm.

In our experiments we used the data projector Acer K11, laptop Asus Transformer Book T100 with touch-

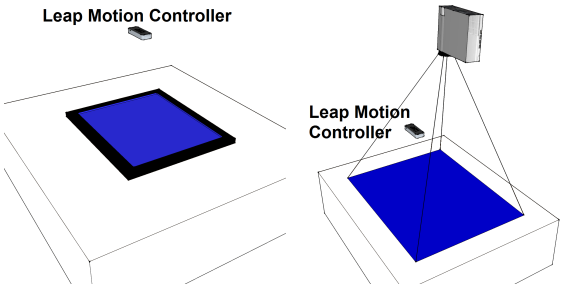


Figure 5: The experimental environment with the Leap Motion, the touchscreen (left) and the data projector (right).

screen and the Leap Motion sensor. The environment for the experiments on the small screen was composed of laptop Asus Transformer and the Leap Motion Controller. The laptop's touchscreen was laid down horizontally on a table, while the Leap Motion was mounted over the screen looking down to the table. The environment is shown in Figure5 left. A setup for the experiments on the large screen used the Acer K11 projector instead of the laptop. The data projector was mounted approximately 800 mm over the table, in distance of circa 500 mm from the Leap Motion. By positioning the projector perpendicularly to the projection screen, we compensated for the keystoning. Other possible projector distortions were not compensated. Both devices were looking straight down to the table as shown in Figure 5 right and Figure 6.

The Leap motion was positioned 280 mm above the screen surface for both screen sizes. At that height, the tracking of the hand and fingers was most reliable in every point of the screens. During the experiments we noticed an issue with a reflection of the emitted IR light in the Leap Motion camera images. The reflected light made a bright spot on the table surface as shown

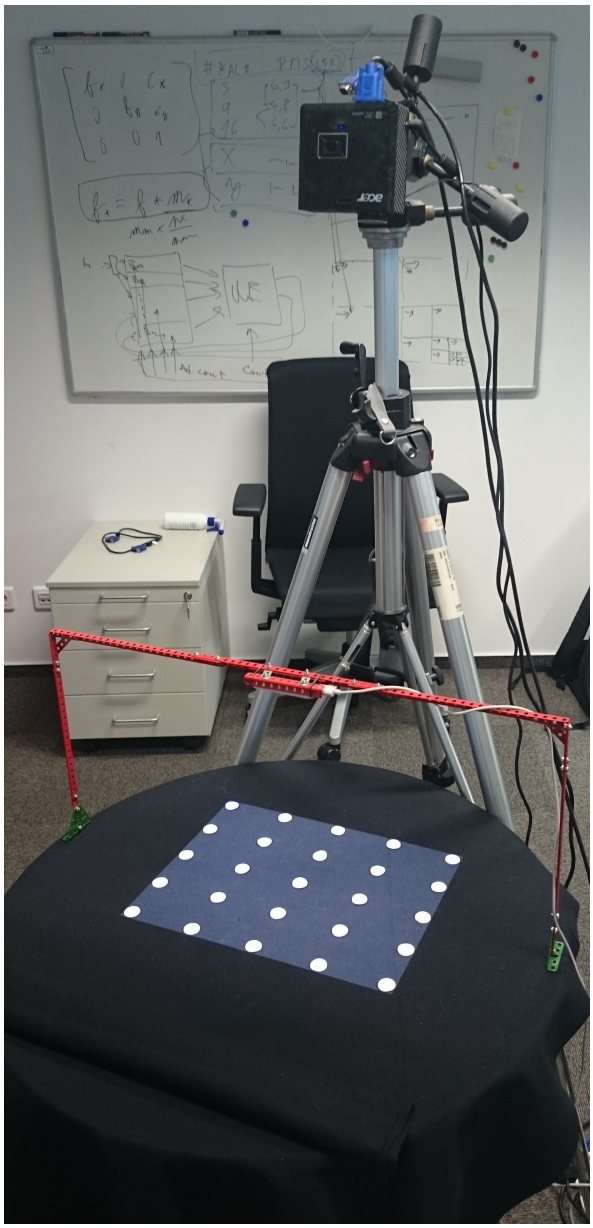
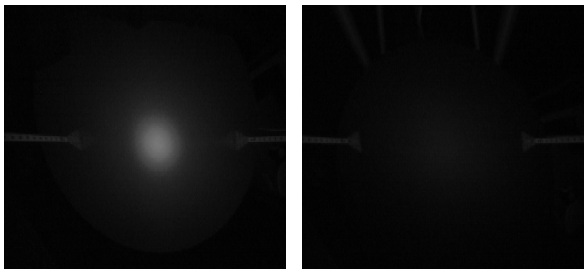


Figure 6: Our experimental tabletop setup with the data projector and the Leap Motion. Five by five pattern that was used for evaluation of the accuracy of our technique is being projected.

in Figure 7 left. Inside this spot the tracking was not reliable and often failed. Thus, it was necessary to cover the table desk with a black cotton fabric which absorbs IR light, but also reflects the visible light that forms the projected image. See Figure 7 right.

We used the Euclidean distance between the center of the displayed point and the point that was touched on the screen as a measure of accuracy. This distance was



(a) Table desk (b) Black cotton

Figure 7: The Leap Motion's view of the clear wooden table desk and the same desk covered by black cotton.

measured in px and then converted to mm using scale computed as

$$scale = \frac{\sqrt{H^2 + V^2}}{D} \tag{13}$$

where H and V are horizontal and vertical resolutions of the screen in pixels and D is diagonal size in mm. This conversion makes the results for the large and small screens comparable. We evaluated the touch accuracy for seven configurations, one small touchscreen (TS), three calibration cases on the small screen (3S, 9S, 16S) and three calibration cases on the larger screen (3L, 9L, 16L). For each of these configurations, we performed 625 measurements. The collected data are visualized in Figure 8. From these measurements we calculated the mean and the standard deviation of distances between the touched point and the displayed point. We also calculated the hit percentage of a target with a radius of 7.75 mm which is recommended by Apple as a suitable size for interactive elements [1]. All results are displayed in Table 1.

Config.		Mean (mm)	Stdev (mm)	Hit (%)
T	**S**	0.7897	0.4689	100
3	**S**	3.5420	2.2226	94.56
	L	3.8179	2.2223	96
9	**S**	1.7528	0.9585	100
	L	2.5187	1.2272	100
16	**S**	**1.2338**	**0.7468**	100
	L	2.1846	1.1162	100

Table 1: The mean and standard deviation are obtained from the distances between the displayed and measured point for each configuration together with hit percentages of the target with 7.75 mm radius. The values are measured for touchscreen (T) and proposed calibration method with 3, 9 and 16 calibration points on small (S) and large (L) screen sizes.

It is not surprising that the accuracy and precision of the calibration method rises with the number of points

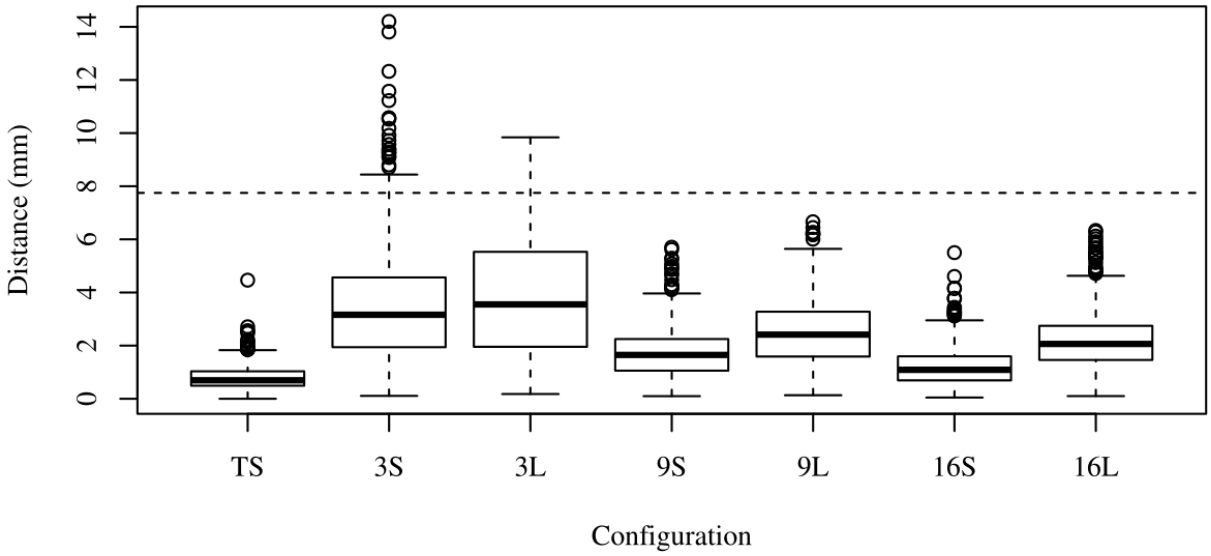


Figure 8: Boxplots of measured Euclidean distances in mm between the displayed and touched points for each configuration. The bottom and top of the box are the first (Q_1) and the third quartiles (Q_3). The band inside the box is the second quartile (Q_2) - the median. The whiskers extend from the lowest distance within $Q_1 - 1.5 * IQR$ to the highest distance within $Q_3 + 1.5 * IQR$. The circles are distances outside the whiskers. The dashed horizontal line represents the recommended target size.

in the used calibration pattern. Also, the accuracy and precision is better on the smaller screen. This can be caused by the projector distortions or by lower precision of hand tracking on the boundaries of the sensory space [5]. The best mean and standard deviation values for our method were obtained with 16 calibration points on the small screen. However, the results are approximately 60% worse than the touchscreen results, the mean value is slightly over one millimetre and the standard deviation is even less than one millimetre. The chance to hit a target of a recommended size are for all configurations 100%, except for calibrations that used 3 points in the calibration pattern. For these cases the hit chance is about 95%.

5 CONCLUSION

We have proposed a novel technique for calibration of a data projector and a Leap Motion Controller. Using the proposed approach, users can perform the calibration only by touching several points on and above the screen plane. No printed patterns, reflective markers, or additional tools are needed.

We have evaluated the proposed technique in a scenario that focuses on the touch based interaction. In this scenario we measured the distance between the center of the displayed point and the point that was touched on the screen. We compared the accuracy and precision of the proposed method with a capacitive touchscreen. The average accuracy of our method is about 1.2 mm on the small screens and 2.2 mm on the larger screens with 16 points calibration pattern. Although our calibration method does not outperform the accuracy and the precision of the touchscreen, it is comparable to it and can easily be used for effective touch based interaction as long as the target sizes follow the previously established recommendations for the touch screen interfaces. The results of the calibration can be used not only for touch based interaction but also for direct pointing and dynamic projection mapping.

Future research will focus on evaluation of the accuracy from the user's point of view. A user study will be used to evaluate whether the accuracy of our method affects user interaction experience. Another possibility is to focus on user interaction techniques suitable for the tabletop displays or to improve the calibration process by compensating for projector distortions.

ACKNOWLEDGEMENT

This work has been supported by the TACR centre of competence project V3C (no. TE01020415).

6 REFERENCES

[1] Apple Inc. iOS Human Interface Guidelines. Retrieved March 10, 2015, from `https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/LayoutandAppearance.html`.

[2] ASUSTeK Computer Inc. Multimedia - Xtion - ASUS. Retrieved March 10, 2015, from http://www.asus.com/Multimedia/Xtion/.

[3] A. Ben-Hamadou, C. Soussen, C. Daul, W. Blondel, and D. Wolf. Flexible calibration of structured-light systems projecting point patterns. *Comput. Vis. Image Underst.*, 117(10): 1468–1481, Oct. 2013. ISSN 1077-3142. URL http://dx.doi.org/10.1016/j.cviu.2013.06.002.

[4] G. Falcao, N. Hurtos, J. Massich, and D. Fofi. Projector-Camera Calibration Toolbox Tech. Rep. , 2009, available at http://code.google.com/p/procamcalib.

[5] J. Guna, G. Jakus, M. Pogacnik, S. Tomazic, and J. Sodnik. An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking. *Sensors*, 14 (2):3702–3720, 2014. ISSN 1424-8220. URL http://www.mdpi.com/1424-8220/14/2/3702.

[6] C. Harrison, H. Benko, and A. D. Wilson. Omnitouch: Wearable multitouch interaction everywhere. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 441–450. ACM, New York, NY, USA, 2011. ISBN 978-1-4503-0716-1. URL http://doi.acm.org/10.1145/2047196.2047255.

[7] Ideum. Touch Tables and Multitouch Coffee Tables. Retrieved March 15, 2015, from http://ideum.com/touch-tables/.

[8] iPiSoft. Depth Sensors Comparison. Retrieved March 10, 2015, from http://wiki.ipisoft.com/Depth_Sensors_Comparison.

[9] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012. ISSN 1424-8220. URL http://www.mdpi.com/1424-8220/12/2/1437.

[10] J. J. LaViola, Jr. An introduction to 3d gestural interfaces. In *ACM SIGGRAPH 2014 Courses*, SIGGRAPH '14, pages 25:1–25:42. ACM, New York, NY, USA, 2014. ISBN 978-1-4503-2962-0. URL http://doi.acm.org/10.1145/2614028.2615424.

[11] Leap Motion Inc. Leap Motion. Retrieved March 10, 2015, from https://www.leapmotion.com/.

[12] K. Litomisky. Consumer rgb-d cameras and their applications. Tech. rep. University of California, Tech. Rep., 2012.

[13] C. B. Markwardt. Non-linear Least-squares Fitting in IDL with MPFIT. In D. A. Bohlender, D. Durand, and P. Dowler, editors, *Astronomical Data Analysis Software and Systems XVIII*, volume 411 of *Astronomical Society of the Pacific Conference Series*, page 251, Sept. 2009.

[14] Microsoft. Kinect for Windows. Retrieved March 10, 2015, from https://www.microsoft.com/en-us/kinectforwindows/.

[15] Microsoft. Samsung SUR40. Retrieved March 10, 2015, from http://www.microsoft.com/en-us/pixelsense/whatsnew.aspx.

[16] J. Moré. The Levenberg-Marquardt algorithm: Implementation and theory. In G. Watson, editor, *Numerical Analysis*, volume 630 of *Lecture Notes in Mathematics*, pages 105–116. Springer Berlin Heidelberg, 1978. ISBN 978-3-540-08538-6. URL http://dx.doi.org/10.1007/BFb0067700.

[17] D. Moreno and G. Taubin. Simple, accurate, and robust projector-camera calibration. In *Proceedings of the 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, 3DIMPVT '12, pages 464–471. IEEE Computer Society, Washington, DC, USA, 2012. ISBN 978-0-7695-4873-9. URL http://dx.doi.org/10.1109/3DIMPVT.2012.77.

[18] S. Murugappan, Vinayak, N. Elmqvist, and K. Ramani. Extended multitouch: Recovering touch posture and differentiating users using a depth camera. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 487–496. ACM, New York, NY, USA, 2012. ISBN 978-1-4503-1580-7. URL http://doi.acm.org/10.1145/2380116.2380177.

[19] A. D. Wilson. Playanywhere: A compact interactive tabletop projection-vision system. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, UIST '05, pages 83–92. ACM, New York, NY, USA, 2005. ISBN 1-59593-271-2. URL http://doi.acm.org/10.1145/1095034.1095047.

[20] A. D. Wilson. Touchlight: An imaging touch screen and display for gesture-based interaction. In *ACM SIGGRAPH 2005 Emerging Technologies*, SIGGRAPH '05. ACM, New York, NY, USA, 2005. URL http://doi.acm.org/10.1145/1187297.1187323.

Content-Aware Re-targeting of Discrete Element Layouts

Stefan Hartmann Björn Krüger Reinhard Klein
University of Bonn
Institute for Computer Science II
Friedrich-Ebert-Allee 144
Germany, 53113 Bonn
{hartmans,kruegerb,rk}@cs.uni-bonn.de

ABSTRACT

Example-based modeling is an active line of research within the computer graphics community. With this work we propose a re-targeting scheme for polygonal domains containing a layout composed of discrete elements. Re-targeting such domains is typically achieved employing a deformation to the initial domain. The goal of our approach is it to compute a novel layout within the deformed domain re-using the discrete elements from the initial layout. We show that the deformed interior of the initial domain can guide the layout algorithm that places the discrete elements. We evaluate our algorithm by re-targeting several challenging city blocks and the results confirm that generated layouts are visually similar to the original ones.

Keywords

example-based synthesis, data-driven re-targeting, layout algorithms.

1 INTRODUCTION

The automatic generation of high quality content is an important research topic in computer graphics. Especially with the emergence of a large variety of publicly available content repositories and community mapping services, modeling by example has increasingly become an active line of research in recent years. This simple and efficient modeling metaphor allows generating novel content either from a single exemplar (*e.g.* a small texture patch) or by first analyzing a larger set of input data (*e.g.* a model collection) to 'learn' structural information, and use the learned knowledge to guide automatic content generation algorithms in a second step.

With the work at hand we contribute a simple but efficient modeling metaphor for re-synthesizing existing layouts that contain sets of discrete element arrangements. Examples for this type of layouts are city blocks (Figure 1a), where building footprints and/or community places serve as discrete atomic elements that are primarily arranged along road segments. Other typical examples are floor plans composed by various types of connected discrete rooms (Figure 1b), interior rooms filled with discrete furniture instances or other types of objects composed of atomic substructures.

Modeling such content from scratch is a tedious task and typically much effort is invested to achieve high quality results. When existing content shall be re-used it shall be 'bended' or re-structured in a way to fit new demands. However, re-targeting and editing such structures usually involves continuous deformations to the initial layout, thus distorting the original arrangements and even their discrete elements. Instead of deforming and repairing distortions, our goal is to re-synthesize a plausible layout that resembles style of the original layout locally and even globally using discrete atomic building blocks. Our key insight for synthesizing such a new layout that mimics the style of the original one is to feed the synthesis technique with a guidance map, that encodes the style present in the initial layout. Such a guidance map can be retrieved from a modification of the initial domain, such as the appliance of a continuous deformation to the initial layout elements. Using this

(a) City block with discrete building footprints.

(b) Office floor plan with discrete room instances.

Figure 1: Real world examples of domains with discrete element arrangements. Both examples are taken from *OpenStreetMap*.

guidance has two major advantages: First, it allows our synthesis technique to resemble the global style of initial layout and it even allows introducing discrete copies of the original elements to maximize the layout compliance, *i.e.* the local style. Second, it guides the algorithm during the exploration of the layout space and allows for efficient pruning this typically large search space.

The main contributions of our work are in particular:

- We present a simple but efficient labeling algorithm for re-targeting polygonal structures incorporating a content-aware continuous guidance map for re-synthesizing arrangements groups.
- We show that the usage of such a guidance map transfers the style of the initial layout to the synthesized one globally and even locally.
- We present an in-depth evaluation on a large set complex layouts and structures containing a varying number of discrete element arrangements, namely urban city blocks.

2 RELATED WORK

City Block Synthesis: An important sub-step in urban modeling is computing interior layouts for city blocks or to be more specific generate parcels to provide space for additional urban sub-structures: buildings or parks. Usually these parcels are generated procedurally, by applying various techniques. In Parish and Müller [14], space for buildings is distributed, employing a subdivision scheme on the oriented bounding box of the city block outline. Aliaga *et al.* [1] compute a voronoi subdivision from perturbed points sampled along the principal direction of the city blocks' oriented bounding box. Vanegas *et al.* [16] generalize parcel generation, by introducing an adapted subdivision scheme on the oriented bounding box of the city block. Their second approach smartly merges regions resulting from the medial-axis to subdivide the city block into larger regions. These are then tessellated along the adjacent street edges.

In Yang *et al.* [18] an initial parcel layout is selected from a set of manually prepared template patterns. To achieve a tight packing of the city block, the template pattern is expanded in discrete steps and the final parcel layout is computed by a sophisticated warping approach. In contrast to these procedural approaches, our goal is to synthesize a new layout that resembles the style of an existing one. In terms of urban modeling, we want to compute a city block layout, that resembles the style of a real world city block *e.g.* taken from *OpenStreetMap*.

Model Re-targeting: Re-synthesis of novel 3D models from small pieces of exiting exemplars was early studied by Merrel *et al.* [8]. They use adjacency rules to guide their model synthesis technique to preserve the local look/style. They generalized their method, incorporating local object self-similarity and non-uniform grids in [9, 10]. The analysis of self-similarities to derive construction rules as context-free grammar that captures the larger object structure was introduced in [2]. In cases where the global structure of one or multiple methods is present *e.g.* provided as scene graph, conforming grammars can be 'learned', being able to reproduce mixtures of exemplar models as demonstrated by Talton *et al.* [15].

Apart from grammar based methods Lin *et al.* [6] suggest to re-target architectural models with dominating irregular structures. They manually annotate and construct an axis aligned bounding box hierarchy, used to automatically extract dominant/longest sequences. The model is then iteratively modified by repeating or scaling the geometry located inside the bounding boxes along these sequences. Very recently Wu *et al.* [17] presented a promising approach for interactive model re-targeting. They derive a set of principal re-targeting directions from descriptors measuring self-similarities. The model is replicated inside a 3D grid, and a multi-label graph cut optimization technique is used to compute the re-targeted result. Our approach in contrast does not rely on computation of self-similarities. It mainly relies on the existence of a guidance map, that captures the style present in the original layout in a higher level.

Orthogonal to our approach are discrete element layout algorithms that use stochastic optimization techniques, *e.g.* Markov Chain Monte Carlo (MCMC). These methods learn object relations from a set of examples [21], are feed with well-known design guidelines [11], or use object relations encoded in factor graphs [20, 19] in order to synthesize novel element layouts. However, this kind of algorithms rely on carefully designed probability distributions and typically are computationally expensive. Thus, they are usually not suited for interactive exploration of the solution space.

3 DOMAIN ANALYSIS

Herein we now explain our approach for re-targeting existing content. First, we briefly explain preliminaries to understand the basic data types. Second, we give a definition of the problem we want to solve. A mathematical formulation to tackle the problem is then introduced in Section 4.

3.1 Preliminaries

A large variety of real-world structures can be abstracted as a set of polygons enveloped by a closed boundary. Typical examples for such real-world structures are building footprints located inside a city

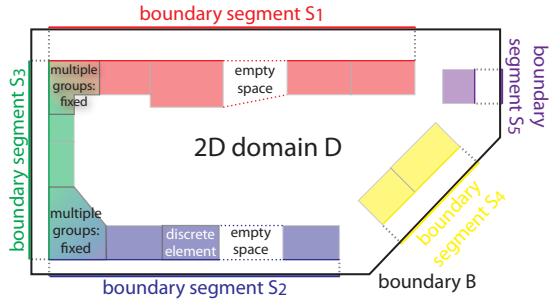


Figure 2: Abstraction of a complex scene as 2D domain D with a set of discrete elements. The domain is enveloped by a boundary B split into subregions S_i, each assigned with a set of discrete elements.

block or rooms located in a floor plan (Figure 1). We define $\mathscr{P} = \{P_1, \ldots, P_n\}$ as the set of n polygons, each representing a 2D outline of discrete elements, placed within a complex scene. Each of them may carry a set of application specific annotations. These elements located in a scene are typically enveloped by a closed curve denoting the boundary $B \in \mathbb{R}^2$ of the scene. The area enclosed by B represents the scene itself and is a planar domain $D \in \mathbb{R}^2$. Inside this area, we expect that the elements are arranged into a set of m groups $\mathscr{G} = \{G_1, \ldots, G_m\}$, each containing a subset of the discrete elements (Figure 2). However, the initial layout does not necessarily have to be a tightly packed layout, *i.e.* elements placed directly next to each other, there may be empty space in between the discrete elements

3.2 Detecting Arrangement Groups

Before we state the problem definition, we first need to analyze the atomic entities and merge them into a set of arrangement groups G. In our setting we expect that the discrete elements are located within an application specific distance near the boundary B. This curve might be further split into a set of boundary segments, separating the curve into l smaller entities $B = \{S_1, \ldots, S_l\}$. The segments $S_j \in B$ typically represent a road segment of a city block or a wall inside a room. In order to merge the elements into groups they need to be assigned to the segments S_j along the boundary B. In order to decide which element is assigned to a certain group we determine the Hausdorff distance $h(P_i, S_j) \forall S_j \in B$ between the outline of the discrete element P_i and the boundary segments $S_j \in B$. In case $h(P_i, S_j) \leq \tau$ falls below an application specific threshold ($\tau = 25.0$ meters in our city block application), the discrete element instance is assigned to that certain arrangement group G_j that corresponds to a boundary segment S_j. It typically happens that a few elements will be assigned to multiple boundary segments, this is an indicator that the element is located at the end of an arrangement group and might need special, application specific, treatment (Figure 2).

Most of the elements are typically assigned to at least one boundary segment S_j. The count of assignments of a discrete element P_i to different S_j is used to deduce three behavioral attributes, *i.e.* the element type, for the element instances:

- *fixed* (F): elements with two or more assignments, are not allowed to be repeated.
- *repeatable* (R): elements with a single assignment, allowed to be placed multiple times.
- *empty space* (E): element that serves as fill region, preferably placed in regions, where no discrete elements covers the space within the group G_j. Note: that element type is a virtually generated element instance and is automatically assigned to each group. In our examples we used a rectangular domain with a fixed dimension $w \times d$ with $w = 1.0$ *meter* and d being the average depths of the discrete elements assigned to a G_j.

3.3 Problem definition

Re-targeting existing models can be achieved by applying various operations to the initial object or domain. Our notion of re-targeting is to apply a deformation to the boundary B or one of its segments S_j. This implies, that the interior of the initial domain is affected by the deformation by applying a map $\phi : D_{in} \rightarrow D_d$, where D_{in} denotes the initial scene domain that shall be re-targeted and D_d denotes the deformed domain. Computing such mappings is well researched and can be achieved by employing different techniques *e.g.* the well known *Mean Value Coordinates* [4] or (quasi)-conformal mappings [7]. In the work at hand we used *Mean Value Coordinates* to realize the mapping $D_{in} \rightarrow D_d$. When ϕ is directly applied to the discrete elements and their corresponding geometry located within the scene, visually unappealing artifacts might be the consequence *i.e.* the elements get distorted and unnaturally stretched/warped. Typically, such effects might be reduced by applying smart deformation techniques, however, this claims for additional knowledge about the objects' structure, which is not the scope of this work.

Another possibility to reduce deformation artifacts is treating the object as a single point, represented by its centroid. This might avoid object deformations, however, will usually lead to different kinds of artifacts such as overlaps, penetration or getting objects dissolved from their initial grouping. In order to circumvent such artifacts described above our major goal is to re-use the undeformed building blocks of the initial scene layout and compute a novel layout which preserves the style of the initial layout. In order to compute a novel layout that mimics the style of the original one we utilize a

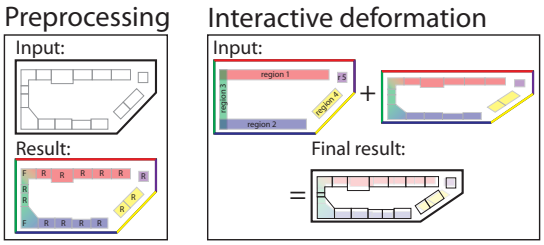


Figure 3: Illustration of our re-targeting concept: As preprocessing step a domain is analyzed and its discrete elements are assigned to arrangement groups (left). When interactively editing the domain, deformations are applied to the boundary. The interior of the domain is deformed and serves as prior to our iterative synthesis technique, that computes a novel layout guided by the deformed discrete elements.

guidance map. The guidance map is a set of deformed discrete elements $\mathscr{P}'$. It results from applying the map ϕ to the interior, *i.e.* the discrete elements $\mathscr{P}$ of the initial domain $\phi : \mathscr{P} \rightarrow \mathscr{P}'$. Although the elements are deformed the overall deformed layout still mimics the style of the initial layout. Therefore, we utilize $\mathscr{P}'$ to guide the algorithm in order to compute a novel layout.

4 RE-TARGETING ARRANGEMENTS

Herein, we present our solution for re-targeting domains that preserves the layout style of the initial domain. Before we present a mathematical formulation to tackle the problem, we comprehensible explain our notions of style that can be preserved by our algorithm. The notion of style covers a wide range of attributes, therefore, we focus on the following three properties in the work at hand:

1. element groupings present in the initial domain shall be preserved,
2. elements prefer to be located at similar relative positions, and
3. elements prefer their initial neighborhood.

In our setting, the term neighborhood does not express the local similarity of geometric attributes. Our goal is that we simply want to preserve the local element ordering found in the initial domain. Up to now we have discussed the input data and its segmentation into a set of discrete elements and their grouping into arrangement groups G_j. From now on, we focus on formulating the re-targeting problem as an optimization problem which can be solved efficiently, thus allowing to interactively explore solutions for different domain deformations.

4.1 Problem formulation

Re-targeting of arrangement groups: The key idea for re-synthesizing a deformed domain is the formulation as labeling problem. This type of problem formulations are well studied in the computer vision community and several efficient solution techniques exist [3]. While the formulation of a labeling problem is typically done for 2D problems *e.g.* image segmentation, we exploit the sub-structures found in our problem setting, namely the linear element arrangement groups. This allows us to solve the layout problem employing efficient algorithms. These considerations result in the following equation to be minimized:

$$L(f_1,\ldots,f_N) = \sum_{k=1}^{N} O(f_k) + \sum_{k=1}^{N-1} \delta(f_k, f_{k+1}) \quad (1)$$

The f_k's are representations of discrete elements that might get placed at different discrete positions located within the parameter domain of the boundary segments S_j. Thus, a $f_k = (P_i, m, d)$ is a tuple that consists of a reference to a discrete element P_i, its current discrete position m and d being the length of P_i projected onto the boundary segment S_j. In our setting the d's are rounded to be a multiple of 1.0 *meter*. The energy we want to minimize in order to find an optimal sequence of discrete elements is given in Equation 1. This energy is composed of two terms, that can be mapped to our notion of style preservation outlined above. The data term $O(f_k)$ measures the cost for assigning an element f_k to a specific location m. Thus, it penalizes elements that will be placed at positions, where they do not overlap with their corresponding distorted instance P_i' within the guidance map. From a different view it can also be seen as prior term pulling the undeformed elements towards the position, where the deformed instance of the particular discrete element is located. We designed $O(f_k)$ to be the area difference between the undeformed element P_i and the area of the intersection of both undeformed P_i and deformed P_i' defined as follows:

$$O(f_k) = A(P_i) - A(P_i \cap P_i')$$

Thus $O(f_k)$ penalizes space not covered between its discrete element P_i and its corresponding deformed P_i'. Further it depends on the current location m, where f_k might be positioned inside the result sequence and it changes every time the domain is modified. The transition between two consecutive discrete elements is penalized by $\delta(f_k, f_{k+1})$ that is defined as follows

$$\delta(f_k, f_{k+1}) = \begin{cases} \alpha & \text{if } f_k \in E \wedge f_{k+1} \in E \\ (1-\alpha) & \text{if } f_k = f_{k+1} \\ 0 & \text{if } f_k \neq f_{k+1} \end{cases}$$

This function encourages omitting two identical discrete elements P_i getting placed next to each other and instead prefers different instances. In order to achieve

 ISBN 978-80-86943-65-7

continuous empty space regions in case if present, we penalize two consecutive empty space elements less severe, than two identical P_i. In our examples we used $\alpha = 0.05$. As described in Section 3.2 we exploit that, elements can be grouped to arrangements along segments of the boundary polygon. For each of them the optimization problem boils down to computing a sequence of elements along a curve, that in total fulfills a certain minimal length M along the parameter domain. In order to employ an efficient algorithm, that minimizes Equation 1 the f_k's are restricted to be placed at discrete positions along the curve. The discretization depends on the specific application (we used discrete steps of 1 meter in our examples). This allows us to reformulate the objective function in the following, recursive way:

$$\begin{aligned} L[f_1, m_{f_1}] &= O(f_1) \\ L[f_k, m_{f_k}] &= O(f_k) + \min(L[f_{k-1}, m_{f_{k-1}}] + \delta(f_{k-1}, f_k)) \end{aligned} \tag{2}$$

We solve this recursive formulation using a graph based dynamic programming approach adapted from Lefebvre *et al.* [5]. In our setting we need to incorporate both, node costs O, *i.e.* overlapping cost and edge costs δ, *i.e.* 'concatenation costs' when growing the graph implicitly. We can terminate the growing in case if the sequence length m including the current top node of the queue is $m \geq M$. As the graph nodes are managed by a priority queue sorted by current sequence cost, the current top f_k^* that satisfies $m \geq M$ node is part of the optimal sequence.

This optimal sequence

$$\underset{m^* \geq M}{f_k^*} = \operatorname{argmin}(L[f_k, m_{f_k}] + \delta(f_{k-1}^*, f_k)) \tag{3}$$

minimizing Equation 1 can then be found, by tracing back the optimal predecessors from f_k^* back to the start of the sequence.

4.2 Iterative Synthesis

While the proposed formulation in Section 4.1 only guarantees a global optimal solution in case the initial layout consists of a single arrangement group located within the initial domain, we employ a meaningful heuristic in case multiple arrangements groups exist. Instead of combining and linking them together and optimize for a globally optimal layout, we have decided to choose an iterative algorithm, that synthesizes each arrangement group separately. This has two major reasons: (1) one of our design goals is interactivity, (2) a global optimization scheme, might affect the performance severely. We decided to sort the groups by their cardinality of the discrete elements assigned to them. Thus, the synthesis technique starts with the most dominant group located inside the initial layout. In order to avoid overlaps with already synthesized sequences, we constrain the $j-th$ arrangement group by the previous $k \in [j-1, \ldots, 1]$ that potentially placed elements (fixed, repeatable) will not penetrate the convex hull of existing sequences.

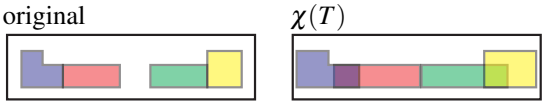


Figure 4: Concept modified guidance map. Left: the original map. Right: The guidance map is transformed by applying anisotropic scaling to the individual elements. Due to this deformation the elements in the guidance map may overlap partially. This allows for a less restrictive positioning of the elements in larger areas.

4.3 Modified guidance maps

As mentioned above the guidance map $\mathscr{P}'$ is the result of applying the map ϕ to the interior elements of the initial domain $\mathscr{P}$. Typically, the creation of such a guidance map is not restricted to a specific deformation method. It is possible to modify the size of the discrete elements in advance or apply a deformation scheme, where weights can be distributed using a simple brush metaphor e.g Möser *et al.* [12]. When using the guidance map $\mathscr{P}$, it might happen that the algorithm presents solutions, that may contain multiple consecutive occurrences of discrete elements. This results from the guidance map, which encourages the algorithm placing elements in regions, where they overlap with their corresponding distorted instance, is cheaper than placing them somewhere else. This may result in visually unappealing repetitions, *i.e.* the same elements is repeated multiple times (Figure 14(a)).

In order to overcome this unappealing side effect we use a modified version of the guidance map $\chi(T)$. Instead of applying the map to the initial discrete elements directly, we apply a scaling operator T to each P_i before applying the deformation map, thus $\chi(T) : \phi(T(P_i)) \rightarrow P_i''$. As a result of this scaling, the deformed discrete elements now partially overlap and, therefore, the algorithm is able to temporarily bypass the original ordering. In our examples we use anisotropic scaling along the direction of the boundary segment S_j by a factor of 1.5 and apply it to each discrete element present in the corresponding arrangement group G_j. An example for such a modified guidance map $\chi(T)$ is shown in Figure 4.

5 EXPERIMENTAL RESULTS

In order to showcase the versatility of our re-targeting approach, we evaluated our algorithm on a set of challenging test cases. Our primary application is the re-targeting of real-world city blocks. We extracted a set of

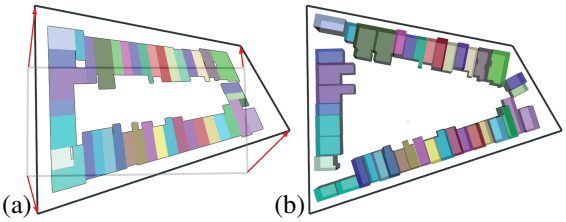


Figure 5: Re-targeting of a tightly packed city block. (a) Deformations (red arrows) applied to the initial domain (grey boundary) result in the edited domain (black boundary), distorted elements (various colors) serve as guidance map. (b) Re-synthesized block layout.

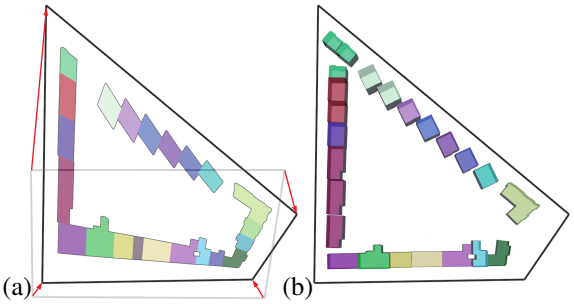


Figure 6: (a) Heavy enlargement of a city block. The applied deformation destroys the noticeable structure of diamond shaped buildings. (b) Re-synthesized block layout: the style of the diamond shaped buildings is preserved and additional elements are spawned in case of the heavily enlarged edge.

city blocks from the well-known community mapping service *OpenStreetMap* (OSM) [13]. When we speak of a city block we strictly speak of a simply connected and piecewise-linear closed boundary labeled as *street*. Inside, such a city block a set of polygons is located that are labeled as buildings in our examples.

Our first experiment, tackles the deformation of tightly packed city blocks, meaning that buildings and their 2-dimensional footprints are densely placed along the street segments.

In Figure 5(a) the deformations applied boundary intersections is highlighted by the red arrows. Applying this deformation to the initial layout results in distortions heavily shearing the elements and changing their size. The block re-layout computed by our algorithm (Figure 5(b)), resembles the style of the original block in the sense, that again a tightly packed layout is computed.

Figure 6 and 7 show different deformations applied to the same initial city block. Again the red arrows highlight the applied deformations. In Figure 6 the block was heavily enlarged, in Figure 7 the blocks was moderately shrunk.

In both cases, the noticeable structure and the orientation of the diamond shaped buildings is preserved.

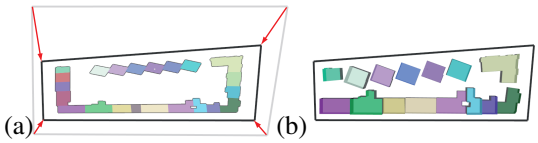


Figure 7: A city block moderately shrunk: notice, that elements get discarded form the layout in case the boundary regions reduced their size. However, the overall layout style is still recognizable.

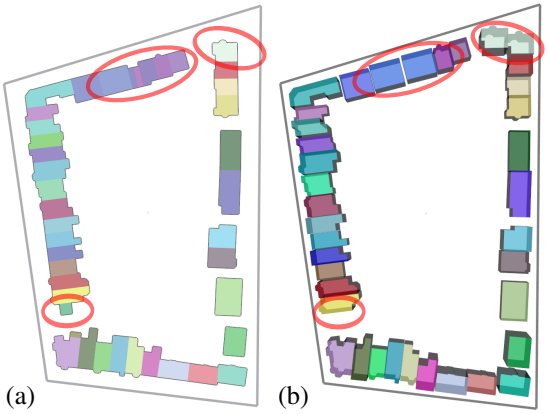


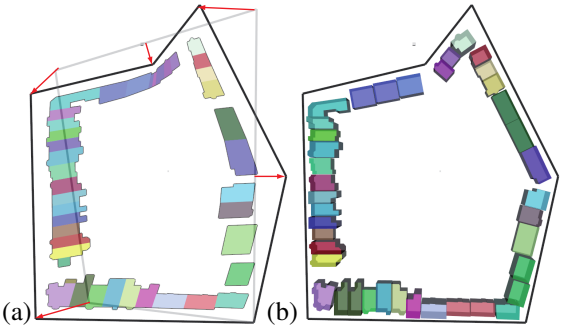
Figure 8: Re-synthesis of an undeformed domain. The original layout (a) is used to guide the algorithm in order to synthesize the novel layout (b) into the undeformed domain.

The result further illustrates that if boundary regions are heavily shrunk or enlarged the algorithm removes or adds discrete elements.

In Figure 8(b) shows a re-synthesis of the initial layout of a city block shown in Figure 8(a). Notice, the similarity between the initial layout shown in Figure 8(a) and the synthesized layout Figure 8(b). The algorithm is not able to reproduce the exact layout, due to the discrete nature of our approach. Figure 9 shows a result, where the outline of the original city block (Figure 8) was topologically modified by adding two additional intersections, leading to strongly visible shearing and bending of the elements within the guidance map. Using this guidance map as prior for the layout, our algorithm was able to produce a plausible new layout, although some footprints present in the initial layout were discarded.

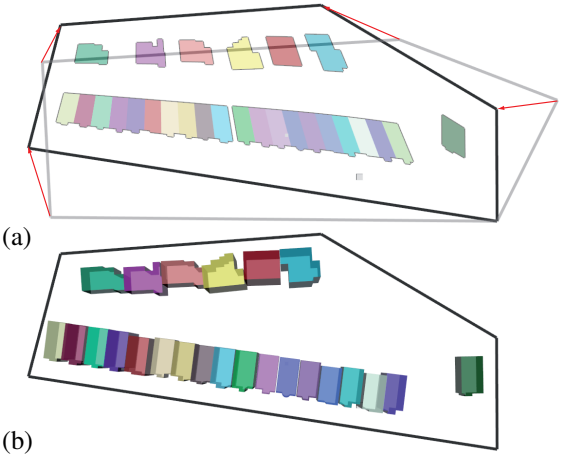
Other, examples demonstrating the strength our method are sparsely packed city blocks. Figure 10, illustrates that if empty space is present between buildings (five at the top street), the algorithm discards first empty space rather than discarding whole buildings since this would result in higher layout costs. For the result shown in Figure 11 we inserted two additional intersections along the edge with densely packed buildings. Notice that along the segment, where the split was introduced, the style (tightly packed footprints) is preserved, and even

 ISBN 978-80-86943-65-7



(a) (b)

Figure 9: Topological modification of the city block outline. Two additional points were inserted and the resulting outline was modified. (a) Applied transformations and warped interior of the layout shown in Figure 8 (a). (b) Resulting layout: Our algorithm splits up the sequences at the newly inserted points. Thus, the unappealing bending of the buildings is avoided.



(a)

(b)

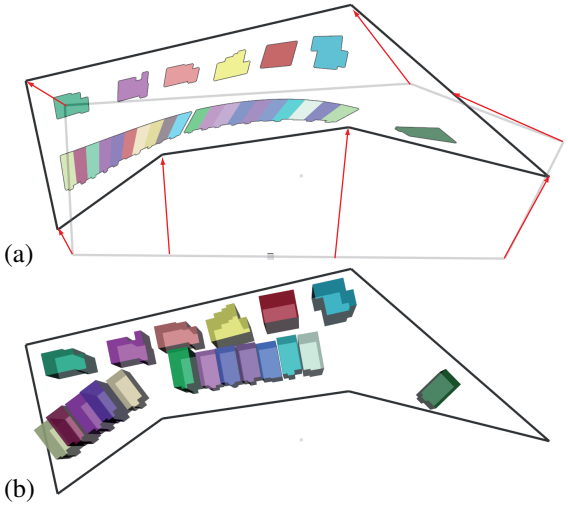
Figure 10: Sparsely packed city block, if edges get shrunk and empty spaces are available, the algorithm first discards empty space rather than discarding buildings.

the space between the five buildings (top segment) is preserved in the synthesized result.

Figure 13 presents an additional result, which combines changing the topology of the boundary and applying a heavy deformation to the initial domain. Even in this totally distorted domain, plausible building arrangements are synthesized.

In Figure 12 we present a result, where our approach was employed to a modeled city block populated with buildings taken from *Trimble Warehouse3d*. Please note, how additional buildings along the enlarged edges are introduced and even the single tree present in the original block is replicated.

Finally, Figure 14 illustrates the usage of the modified guidance map. Notice that when an unmodified guidance map is employed, multiple consecutive repetitions



(a)

(b)

Figure 11: Deformed block after two additional intersections where added to the boundary. It can be noticed, that the continuous tight building layout is split in two disjoint sequences, which are still tightly packed. Further note the space is preserved between the upper five buildings.

are present in the result Figure 14(a) and (c). Employing a simple anisotropic scaling to the individual elements present within the guidance map reduces, the consecutive repetitions (Figure 14(b) and (d)). Finally, we want to note that generating the results found inside the paper took no longer than 0.15*s*, using a desktop workstation with *Core i7 4930K (3.4 GHz)* and *32GB RAM*.

6 DISCUSSION AND LIMITATIONS

In the previous section we presented a large set of different re-targeting results. Although, all these results look plausible, we identified a few limitations that need to be discussed inhere. In our current implementation we rely on two conditions found in the input data (1) the atomic discrete elements blocks can be combined into larger groups, and (2) the elements are dominantly arranged along the boundary. Our algorithm is not restricted to element groups located near the boundary. In principle any group of elements that dominantly follow a curve can be synthesized with our algorithm. Only a few modifications need to realized to handle this case. (1) a method to fit a plausible curve, along the elements will be synthesized and (2) an additional data term, that handles, how 'well' the elements are oriented to the curve locally. Even in this case, we could again exploit the structure of the problem and still have an efficient algorithm to compute solutions. However, if no such groups are identifiable, our algorithm will produce failure cases. Further we may note, that we currently do not see a promising straightforward extension of our approach for 'real' 2D dimensional domains.

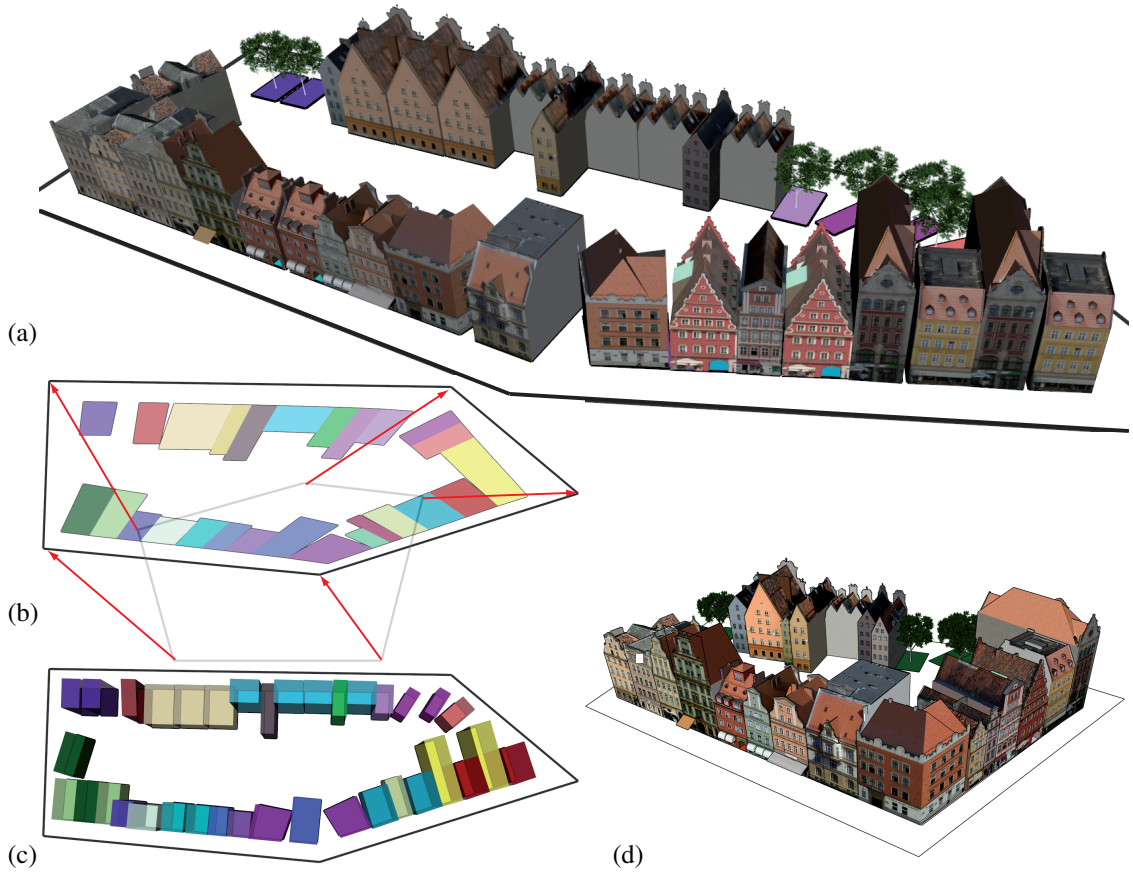


Figure 12: (a) A deformation of a city block populated with detailed modeled buildings. Notice, that in case of enlargement additional buildings and even trees are inserted. (b) We show the deformation and the guidance map. (c) Top-view of the resulting layout. (d) The original building block.

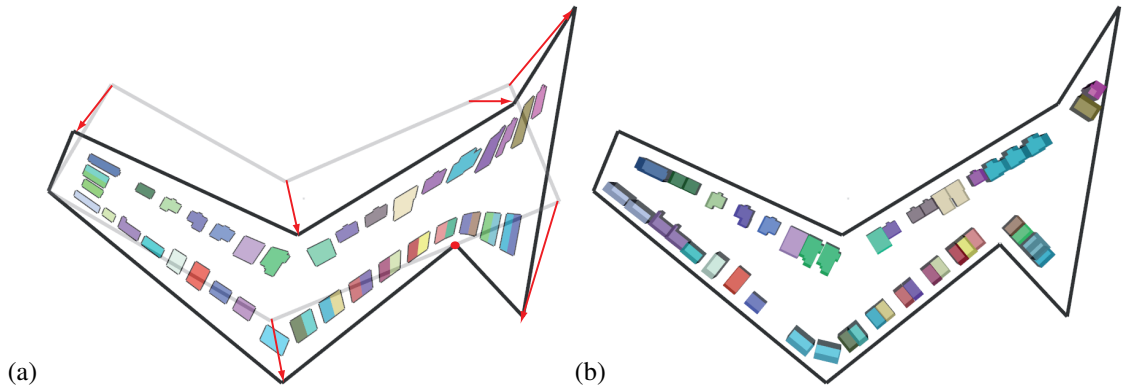


Figure 13: Heavily distorted block with additional intersections added to the boundary (a). Please note, that even applying such heavy deformations to the initial boundary, the element grouping (consecutive groups of two buildings) are present in the result (b).

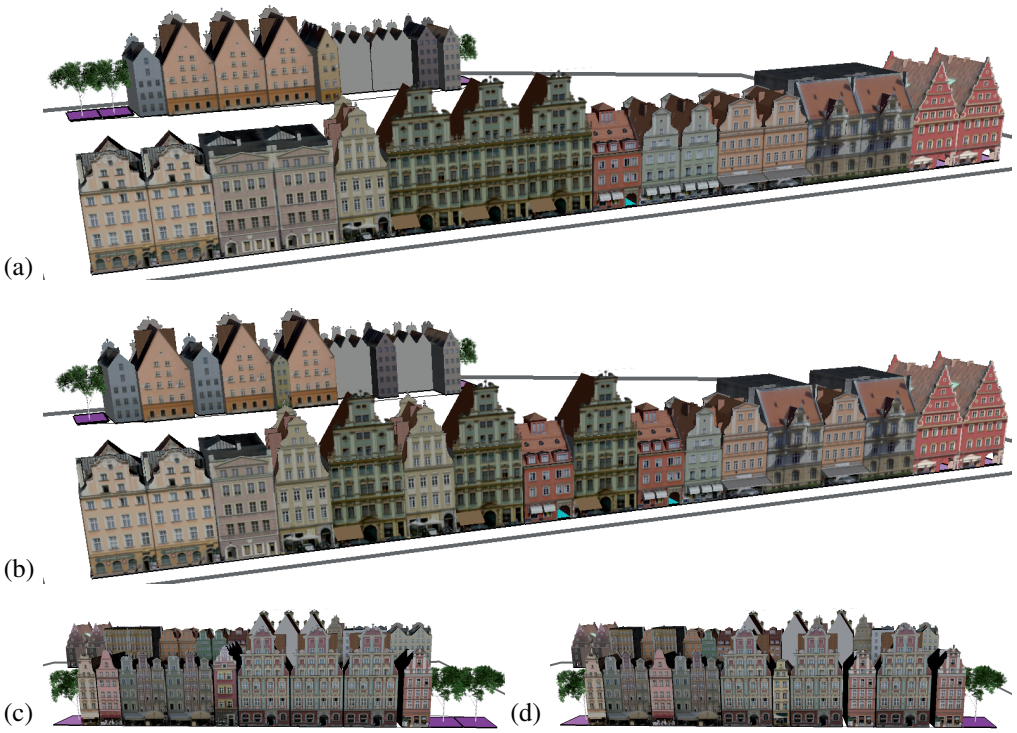


Figure 14: Illustrating the effect of using modified guidance maps. We show additional modifications to the block introduced in Figure 12. The results shown in (a) and (c) where obtained by simple warping of the guidance map. In this case multiple consecutive repetitions of the same building are visible. To create the results in (b) and (d), simple anisotropic scaling (referred to as blurring in Figure 4) was employed. This simple modification already reduced the number of consecutive repetitions significantly.

In addition, we identified another limitation introduced by our continuous deformation method acting globally across the polygon: It happens that even if the boundary edges do not change their length, the layout changes along these edges, because the underlying guidance map has changed. This effect, might be avoided by choosing a different deformation method, that can locally control the allowed deformation. We plan to investigate the approach presented by Möser *et al.* [12] for further evaluation. Furthermore, we see a promising set of different interesting research directions. First, we are interested in finding an efficient way to extend the algorithm to 2D or even 3D structures. In the 2D case similar deformations could be employed, however, presenting an at least near optimal solution at interactive rates, seems to be an important research direction. An orthogonal direction would be exploring the possibilities to combine the guidance map with existing forward procedural modelers, in order to guide the construction of the derivation tree. Finally, we would like to integrate our approach into an interactive data driven urban modeling framework.

7 CONCLUSION

We presented a simple but efficient way for re-synthesis of polygonal domains with groups of discrete building blocks. We employed our method to a large variety of different of city blocks and showed how the style of the initial layout can be preserved even under complex deformations. Our synthesis technique allows for exploring the layout space of different deformations at interactive rates, thus allowing further integration into an urban modeling and editing system.

8 ACKNOWLEDGMENTS

We thank the AiF Project GmbH for partial funding the work at hand. In addition the authors would link to thank the reviewers for their valuable comments and kindly thank Max Hermann for inspiring discussions.

9 REFERENCES

[1] Daniel G. Aliaga, Carlos A. Vanegas, and Bedřich Beneš. Interactive example-based urban layout synthesis. In *ACM SIGGRAPH Asia 2008 Papers*, pages 160:1–160:10, 2008.

[2] Martin Bokeloh, Michael Wand, and Hans-Peter Seidel. A connection between partial symmetry and inverse procedural modeling. In *ACM SIGGRAPH 2010 papers*, SIGGRAPH '10, pages 104:1–104:10. ACM, 2010.

[3] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, November 2001.

[4] Kai Hormann and Michael S. Floater. Mean value coordinates for arbitrary planar polygons. *ACM Trans. Graph.*, 25(4):1424–1441, October 2006.

[5] Sylvain Lefebvre, Samuel Hornus, and Anass Lasram. By-example synthesis of architectural textures. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 84:1–84:8, New York, NY, USA, 2010. ACM.

[6] Jinjie Lin, Daniel Cohen-Or, Hao (Richard) Zhang, Cheng Liang, Andrei Sharf, Oliver Deussen, and Baoquan Chen. Structure-preserving retargeting of irregular 3d architecture. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2011)*, 30(6):Article 183, dec 2011.

[7] Yaron Lipman, Vladimir G. Kim, and Thomas A. Funkhouser. Simple formulas for quasiconformal plane deformations. *ACM Trans. Graph.*, 31(5):124:1–124:13, September 2012.

[8] Paul Merrell. Example-based model synthesis. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, I3D '07, pages 105–112, 2007.

[9] Paul Merrell and Dinesh Manocha. Continuous model synthesis. In *ACM SIGGRAPH Asia 2008 papers*, SIGGRAPH Asia '08, pages 158:1–158:7, 2008.

[10] Paul Merrell and Dinesh Manocha. Model synthesis: A general procedural modeling algorithm. *IEEE Trans. Vis. Comput. Graph.*, 17(6):715–728, 2011.

[11] Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. Interactive furniture layout using interior design guidelines. *ACM Trans. Graph.*, 30(4):87:1–87:10, July 2011.

[12] Sebastian Möser, Patrick Degener, Roland Wahl, and Reinhard Klein. Context aware terrain visualization for wayfinding and navigation. *Computer Graphics Forum*, 27(7):1853–1860, October 2008.

[13] Foundation OpenStreetMap. Openstreetmap, 2015.

[14] Yoav I. H. Parish and Pascal Müller. Procedural modeling of cities. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 301–308, New York, NY, USA, 2001.

[15] Jerry Talton, Lingfeng Yang, Ranjitha Kumar, Maxine Lim, Noah D. Goodman, and Radomír Měch. In *Learning Design Patterns with Bayesian Grammar Induction*, 2012.

[16] Carlos A Vanegas, Tom Kelly, Basil Weber, Jan Halatsch, Daniel G Aliaga, and Pascal Müller. Procedural generation of parcels in urban modeling. In *Computer graphics forum*, volume 31, pages 681–690. Wiley Online Library, 2012.

[17] Xiaokun Wu, Chuan Li, Michael Wand, Klaus Hildebrandt, Silke Jansen, and Hans-Peter Seidel. 3d model retargeting using offset statistics. *International Conference on 3D Vision*, -(-):–, 2010.

[18] Yong-Liang Yang, Jun Wang, Etienne Vouga, and Peter Wonka. Urban pattern: Layout design by hierarchical domain splitting. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2013)*, 32:Article No. xx, 2013.

[19] Yi-Ting Yeh, Katherine Breeden, Lingfeng Yang, Matthew Fisher, and Pat Hanrahan. Synthesis of tiled patterns using factor graphs. *ACM Trans. Graph.*, 32(1):3:1–3:13, February 2013.

[20] Yi-Ting Yeh, Lingfeng Yang, Matthew Watson, Noah D. Goodman, and Pat Hanrahan. Synthesizing open worlds with constraints using locally annealed reversible jump mcmc. *ACM Trans. Graph.*, 31(4):56:1–56:11, July 2012.

[21] Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher. Make it home: automatic optimization of furniture arrangement. In *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, pages 86:1–86:12. ACM, 2011.

A Weight Adjustment Strategy to Prevent Cascade of Boosted Classifiers from Overfitting

Ki-Yeong Park
Center for Integrated Smart Sensors
312, IT Convergence Center
Daehak-ro, Yuseong-gu,
Daejeon 305-701, Republic of Korea
cpky0@kaist.ac.kr

Dong-Seok Kim
Center for Integrated Smart Sensors
312, IT Convergence Center
Daehak-ro, Yuseong-gu
Daejeon 305-701, Republic of Korea
kds1130@kaist.ac.kr

ABSTRACT

We propose a weight adjustment strategy to prevent a cascade of boosted classifiers from overfitting and to achieve an improved performance. In cascade learning, overfitting often occurs due to the iterative applications of bootstrapping. Since false positives that the previous classifier misclassifies are collected as negative examples through bootstrapping, negative examples more similar to positive examples are prepared as stages go on, and thus classifiers become tuned to the positive examples. When overfitting occurs, the classifier cascade shows performance degradation more in the detection rate than in the false alarm rate. In the proposed strategy, the imbalance between the detection rate and the false alarm rate is evaluated by computing the weight ratio of positive examples to negative examples and it is compensated by adjusting the weight ratio prior to boosting at each stage. Experimental results confirm the effectiveness of the proposed strategy. For experiments, face and pedestrian classifier cascades were trained by employing previous approaches and the proposed strategy. By employing the proposed strategy, the detection rate of classifier cascades was significantly improved for both face and pedestrian.

Keywords

AdaBoost, bootstrapping, cascade of boosted classifiers, overfitting, face detection, pedestrian detection

1. INTRODUCTION

Cascade of boosted classifiers is an object detection method popularly employed in real-time systems. Since Viola and Jones introduced a real-time face detector based on classifier cascade [Vio04], the cascade structure has been successfully adopted in detecting various objects such as faces [Li11, Liu12, Cev13], vehicles [Cui10, Siv12], and pedestrians [Che11, Xin11, Hoa12, Pri13], and now it serves as a foundation for modern detectors [Dol12]. Many state-of-the-art object detectors utilize the cascade structure alone or combined with other object detection methods [Dol09, Che11, Xin11].

The success of classifier cascade is mainly due to its fast processing speed. In object detection domain, where a few objects have to be distinguished from an extremely large number of non-objects, classifiers have to be trained to achieve a very high detection rate (e.g., 95%) and an extremely low false alarm rate (e.g., 10^{-6}). This asymmetric performance goal can be efficiently achieved by employing the cascade structure. Classifier cascade achieves a fast processing speed by pre-filtering most of non-objects with simple classifiers at early stages and a high detection accuracy by using more complex classifiers at later stages [Vio01]. Enzweiler and Gavrila compared several pedestrian detectors and reported that the pedestrian detector based on the cascade structure was approximately 20 times faster than the other detectors [Enz09].

Successful cascade learning requires extensive trial-and-error. In cascade learning, each classifier in a cascade is trained just until a given performance goal is achieved. Therefore, the performance of a classifier cascade cannot be simply improved by adopting a more sophisticated algorithm for training each classifier. Furthermore, the detection rate of a classifier cascade is definitely degraded while the false alarm rate will be improved by appending more classifiers to the cascade.

In this paper, we propose a cascade learning strategy to achieve an improved performance. In cascade learning, negative examples required for training each classifier are collected through bootstrapping [Sun98]. Overfitting often occurs due to iterative applications of bootstrapping. As stages go on, negative examples which are more similar to positive examples are

collected and classifiers become tuned to the positive examples. In the proposed strategy, the imbalance between the detection rate and the false alarm rate is evaluated at each stage to detect overfitting and the weights of training examples are adjusted to avoid overfitting. The rest of this paper is organized as follows: We briefly review related works in the following subsection and describe conventional cascade learning algorithms in Section 2. The proposed strategy is described in Section 3, and experimental results are presented in Section 4. In Section 5, the conclusions are drawn.

1.1 Related Work

Multi-exit cascade has been studied as an improved cascade structure [Pha08]. While the information obtained by the previous classifier is discarded prior to boosting at each stage in the Viola-Jones cascade, it is inherited to the subsequent classifier in the multi-exit cascades such as boosting chain [Xia03], nested cascade [Wu04], soft cascade [Bou05], and embedded cascade [Sab12]. By recycling information at each stage, the information redundancy between classifiers will be reduced and more efficient classifier cascade can be constructed [Sun14].

In cascade learning, each classifier is trained to achieve a very low false negative rate (e.g., 0.5%) and a rather high false alarm rate (e.g., 50%). Training a classifier to achieve such an asymmetric goal is not a task typically addressed by machine learning algorithms. In the Viola-Jones' scheme, a very low false negative rate is achieved by adjusting threshold of each classifier [Vio04], however, with the penalty of sharply increased false alarm rate [Xia03]. By adopting an asymmetric AdaBoost, the asymmetric goal can be achieved more effectively [Vio01, Mas07, Sun07, Wu08, Lan12, Wan12]. Masnadi-Shirazi and Vasconcelos [Mas07] presented a theoretically solid asymmetric boosting algorithm based on the statistical view of boosting, and Sun et al. [Sun07] investigated several asymmetric boosting algorithms which assign larger weights to false negatives by manipulating weight update rule. Landesa-Vázquez and Alba-Castro [Lan12] showed that AdaBoost can be used as an asymmetric learning algorithm by manipulating initial weights of training examples instead of manipulating weight update rule.

Cascade learning has several parameters such as the number of classifiers in a cascade and the performance goal for each classifier. Since the processing speed and the performance of a classifier cascade vary non-intuitively with these parameters, a successful cascade learning requires extensive trial-and-error [Sab12]. Several cascade optimization algorithms have been proposed [Sab12, Wan12, Lud13, Pai14]. These algorithms search for optimal trade-off between the detection performance and the processing speed.

2. CONVENTIONAL CASCADE LEARNING ALGORITHM

In this section, AdaBoost algorithm and conventional cascade learning algorithms are described. Among the several variants of AdaBoost algorithms, the gentle AdaBoost [Fri00] is presented, which is also used for describing the proposed strategy in Section 3. For cascade learning algorithms, both the Viola and Jones' algorithm [Vio04] and the multi-exit cascade learning algorithm [Xia03, Pha08] are described.

2.1 AdaBoost

AdaBoost is a machine learning algorithm for constructing a strong classifier as a linear combination of weak classifiers [Fre97]. AdaBoost maintains a distribution of weights over the training examples by increasing the weights of misclassified examples and decreasing those of correctly classified examples at each boosting round. With this weight update rule, AdaBoost focuses on training examples so far misclassified.

For given training examples $(x_1, y_1), \ldots (x_N, y_N)$, where x_i is an example image and y_i is class label (+1, –1 for positive and negative examples, respectively), the weights of training examples are evenly initialized before boosting begins. At each boosting round, a weak classifier is learned from the weight distribution, and the weight of each training example is updated according to the prediction of the weak classifier as

$$w_{t+1}(i) = \frac{w_t(i) \cdot \exp\left(-y_i \cdot h_t(x_i)\right)}{Z_{t+1}}, \qquad (1)$$

where $w_t(i)$ and $w_{t+1}(i)$ are the weights of a training example x_i at the rounds t and t+1, respectively, and Z_{t+1} is the normalization factor. $h_t(x_i)$ is a weak classifier which outputs a confidence-rated prediction (a value between –1 and +1) for each example. The weak classifier learning and the weight update are repeated until a performance goal is achieved, and the boosted classifier is given as a linear combination of the weak classifiers as

$$H(x_i) = sign\left(\sum_{t=1}^{T} h_t(x_i) + b\right), \qquad (2)$$

where T is the number of weak classifiers in the boosted classifier and b is the bias applied for the classifier. In cascade learning, a positive bias ($b > 0$) is applied to the boosted classifier to achieve a lower false negative rate, with less weak classifiers, by sacrificing its false alarm rate.

To achieve the asymmetric performance goal more efficiently, an asymmetric AdaBoost algorithm can be adopted. As Landesa-Vázquez and Alba-Castro [Lan12] showed, AdaBoost can be directly used as an asymmetric learning algorithm. Boosting becomes to

focus more on reducing the false negative rate when larger initial weights are assigned to the positive examples as

$$w(i) = \begin{cases} \frac{\gamma}{2N_+}, & \text{for } y_i = +1 \\ \frac{(1-\gamma)}{2N_-}, & \text{for } y_i = -1 \end{cases}, \quad (3)$$

where $\gamma \in (½, 1)$ is the asymmetric parameter, and N_+ and N_- are the number of positive examples and that of negative examples, respectively.

2.2 Viola-Jones Cascade Learning

AdaBoost [Fre97, Fri00] is used to train each classifier in a cascade. Each classifier is trained to achieve a very low false negative rate and a rather high false alarm rate. In object detection domain where there exist only a few objects contrary to an extremely large number of non-objects in an image, it is possible to construct a simple classifier with a very low false negative rate by sacrificing its false alarm rate [Vio01]. Cascade of boosted classifiers achieves both fast processing speed and high accuracy by discarding negatives with these simple classifiers at early stages and by using more complex classifiers at later stages [Vio04].

Negative examples for training each classifier are prepared through bootstrapping. False positives which the previous classifier misclassifies are collected and used as negative examples for training the subsequent classifier [Sun98]. By the iterative applications of bootstrapping, cascade learning becomes to face more difficult negative examples as stages go on.

The number of classifiers in a cascade can be determined from a given goal for the detection rate and the false alarm rate [Vio04]. For example, a detection rate of 95% and a false alarm rate of 6×10^{-6} can be achieved by constructing a 10-stage cascade and training each classifier to achieve a detection rate of 99.5% ($0.995^{10} \approx 0.95$) and a false alarm rate of 30% ($0.3^{10} \approx 6 \times 10^{-6}$). One thing we have to remember in the cascade design is that the detection rate of a classifier cascade as well as its false alarm rate decrease as more classifiers are appended to the cascade, which makes the cascade optimization much complicated.

2.3 Multi-exit Cascade Learning

Xiao et al. [Xia03] proposed the multi-exit cascade structure called boosting chain where each classifier inherits score from its previous classifier. In the Viola-Jones cascade learning, weights of training examples are evenly initialized before boosting begins at each stage, and thus information obtained by its previous classifier is discarded. For a multi-exit cascade, each classifier is trained after the weights of training examples are initialized and then updated according to the predictions of the weak classifiers of its previous classifier [Xia03]. Weight of each training example is initialized and updated as

$$w_{M+1}(i) = \frac{w_1(i) \cdot \exp\left(-y_i \cdot \sum_{t=1}^{M} h_t(x_i)\right)}{Z_{M+1}}, \quad (4)$$

where $w_1(i)$ is evenly initialized weight, $h_t(x_i)$ is a weak classifier which outputs a confidence-rated prediction for each training example, M is the number of weak classifiers in the previous classifier, and Z_{M+1} is the normalization factor. Boosting proceeds with the weights initialized and updated.

3. PROPOSED CASCADE LEARNING STRATEGY

In this section, the proposed cascade learning strategy is presented, which evaluates the imbalance between the detection rate and the false alarm rate and compensates it to avoid overfitting.

3.1 Overfitting in Cascade Learning

While a classifier trained to be overly complex may classify the training examples perfectly, it is unlikely perform well on new patterns. This situation is known as overfitting [Dud01]. Even though it is often believed that AdaBoost does not suffer from overfitting [Fri00], cascade learning which employs AdaBoost to train each classifier in the cascade often undergoes overfitting. In cascade learning, false positives which the previous classifier misclassifies are collected and used as negative examples for training the subsequent classifier [Sun98, Vio04]. As stages go on and the bootstrapping is iterated, negative examples which are more similar to positive examples are collected, and thus the generalization of the classifier cascade becomes degraded.

Even though the overfitting occurs due to the bootstrapping iterations, it will not be solved by reducing the number of bootstrapping iterations. The bootstrapping iterations can be reduced by designing the cascade to have less number of classifiers and each classifier to achieve a lower false alarm rate (to reject more negatives) as we described in Subsection 2.2. However, if each classifier in a cascade achieves a lower false alarm rate, negative examples even more similar to the positive examples will be collected through bootstrapping. This will worsen the overfitting problem.

3.2 Proposed Cascade Learning Strategy

When overfitting occurs, the classifier cascade shows performance degradation more in the detection rate than in the false alarm rate, since each classifier is trained with true positives (positive examples) and false positives (bootstrapped negative examples). In the proposed strategy, the imbalance between the detection rate and the false alarm rate of the previous

classifier is evaluated and it is compensated by adjusting the weights of training examples. The multi-exit cascade structure described in Subsection 2.3 is employed in the proposed strategy. In the multi-exit cascade learning, the weights of training examples are updated using the weak classifiers of the previous classifier before boosting begins at each stage [Xia03]. The imbalance between the detection rate and the false alarm rate can be evaluated from the updated weight distribution by computing the weight ratio of positive examples to negative examples, and it can be compensated by adjusting the weight distribution.

Fig. 1 shows the proposed learning algorithm for each classifier in a cascade. This classifier learning has to be repeated to construct a classifier cascade. After the weights of training examples are initialized and updated, the ratio of the sum of positive example weights to that of negative example weights is computed as

$$r_M = \frac{\sum_{y_i=+1} w_1(i)\cdot\exp\left(-\sum_{t=1}^{M} h_t(x_i)\right)}{\sum_{y_i=-1} w_1(i)\cdot\exp\left(\sum_{t=1}^{M} h_t(x_i)\right)}, \quad (5)$$

where $w_1(i)$ is an evenly initialized weight, $h_t(x_i)$ is a weak classifier which outputs a confidence-rated prediction (a value between –1 and +1) for example x_i, and M is the number of weak classifiers used in the previous classifier.

Since each classifier except the first classifier in a cascade is trained with bootstrapped negative examples (false positives), the sum of negative example weights is larger than that of positive examples weights ($r_M < 1$) after the weight update. When larger weights are given to the negative examples at the weight initialization, boosting will focus more on reducing the false alarm rate [Lan12]. To prevent boosting from focusing more on reducing the false alarm rate, the weight ratio has to be adjusted to be balanced. The weight of each training example is adjusted by adding a bias b_w to the weight update rule of the multi-exit cascade as

$$w_{M+1}(i) = \frac{w_1(i)\cdot\exp\left(-y_i\cdot\left(\sum_{t=1}^{M} h_t(x_i) + b_w\right)\right)}{Z_{M+1}}, \quad (6)$$

where Z_{M+1} is the normalization factor. The bias b_w for the weight adjustment is computed as

$$b_w = \frac{1}{2}\cdot\ln\left(\frac{r_M}{W_a}\right), \quad (7)$$

where W_a is the weight adjustment factor devised for experimental purpose, which is a desired weight ratio of positive examples to negative examples. $W_a = 1$ should be used to compensate the imbalance between the detection rate and the false alarm rate. If the weights are adjusted with $W_a < 1$, boosting will focus more on reducing the false alarm rate, and vice versa. To confirm this, several different weight adjustment factors will be tested in our experiments.

After the weights of training examples are adjusted, boosting is proceeded as in the conventional cascade learning. Since the weights are updated using the weak classifiers of the previous classifier before boosting begins, the prediction of each boosted classifier has to be computed by summing the predictions of all the weak classifiers used in the previous classifier as well as in the current classifier as

$$H(x_i) = sign\left(\sum_{t=1}^{M+k} h_t(x_i) + b\right), \quad (8)$$

where M is the number of weak classifiers used in the previous classifier, k is the number of newly learned weak classifiers, and b is the bias applied to the current classifier to reduce the false negative rate by sacrificing its false alarm rate.

- Given training examples (positive examples and bootstrapped negative examples):
  - Initialize the weights of examples evenly.
  - Update the weights of examples using the weak classifiers of the previous classifier and compute the weight ratio r_M using the equation (5).
  - Compute the bias b_w for weight adjustment with the weight adjustment factor $W_a = 1$ using the equation (7).
  - Adjust the weights of examples with the bias b_w using the equation (6).
- Repeat the following process until a given performance goal is achieved:
  - Proceed with the conventional boosting (Train a weak classifier and update the weights with the predictions of the weak classifier).
  - Determine the bias b that is applied to the boosted classifier to achieve the goal for the false negative rate as in the equation (2).
- Output the boosted classifier, which is a linear combination of all the weak classifiers learned at the previous stage as well as at this stage as in the equation (8).

Figure 1. The proposed classifier learning algorithm for constructing a classifier cascade

 ISBN 978-80-86943-65-7

4. EXPERIMENTAL RESULTS

For experiments, we trained classifier cascades for face and pedestrian by employing the Viola-Jones cascade (VJ), the multi-exit cascade (Multi-exit), the Viola-Jones cascade with the asymmetric AdaBoost (Asymmetric), and the proposed strategy (Proposed). Each classifier in the cascades was trained using the gentle AdaBoost [Fri00] with the same set of Haar-like features [Vio04], and was trained to achieve the same performance goal: a detection rate of 99.5% and a false alarm rate of 50%. The classifier cascades employing the asymmetric AdaBoost [Lan12] were trained with the asymmetry parameter γ = 4/5 in the equation (3) to assign four times of weights to positive examples. The weight adjustment factors W_a = 0.5 and 2.0 were also tested in the experiments, which assign twice weights to negative examples and to positive examples, respectively.

Face examples were obtained by cropping the images in Labeled Faces in the Wild-a [Hua07, Wol11]. All of the 13,233 face examples were resized to 18 x 22, and 2,000 examples of them were used for training and the rest was used for test. Pedestrian examples were cropped from test images in the Daimler Stereo Pedestrian Detection Benchmark Dataset [Kel11], and were resized to 14 x 28. Among 13,714 pedestrian examples, 5,000 examples were used for training and the rest was used for test. For test, 1,000,000 negative examples were prepared by randomly cropping from more than 8,000 images which do not contain any objects.

4.1 Impact of Weight Adjustment

Fig. 2 shows the learning curves of the classifier cascades evaluated at each stage with test examples. The false negative rate and the false alarm rate of the classifier cascades are presented separately to show the imbalance between them. The performance goal for the cascade learnings is also presented in the figure (Goal).

The imbalance problem is obviously observed in the face cascade learnings. All the cascades employing the previous approaches overachieved the false alarm rate goal while they failed to achieve the false negative rate goal. In case of the pedestrian cascade learnings, all the cascades employing the previous approaches underachieved both the false negative rate and the false alarm rate goals. However, the degradation in the false negative rate was more severe.

The experimental results show that the performance imbalance problem can be solved by employing the proposed strategy. When larger weights were assigned to negative examples (W_a = 0.5), the false negative rate was similar to or worse than that of cascades employing the conventional approaches, and it was improved when the same or larger weights were assigned to positive examples ($W_a \geq 1.0$). The biggest improvement on the false negative rate was achieved when larger weights were assigned to positive examples (W_a = 2.0). However, in this case, there was degradation in the false alarm rate. Moreover, the pedestrian classifier cascade failed to reduce the false alarm rate anymore at 10th stage as shown in Fig. 2(b).

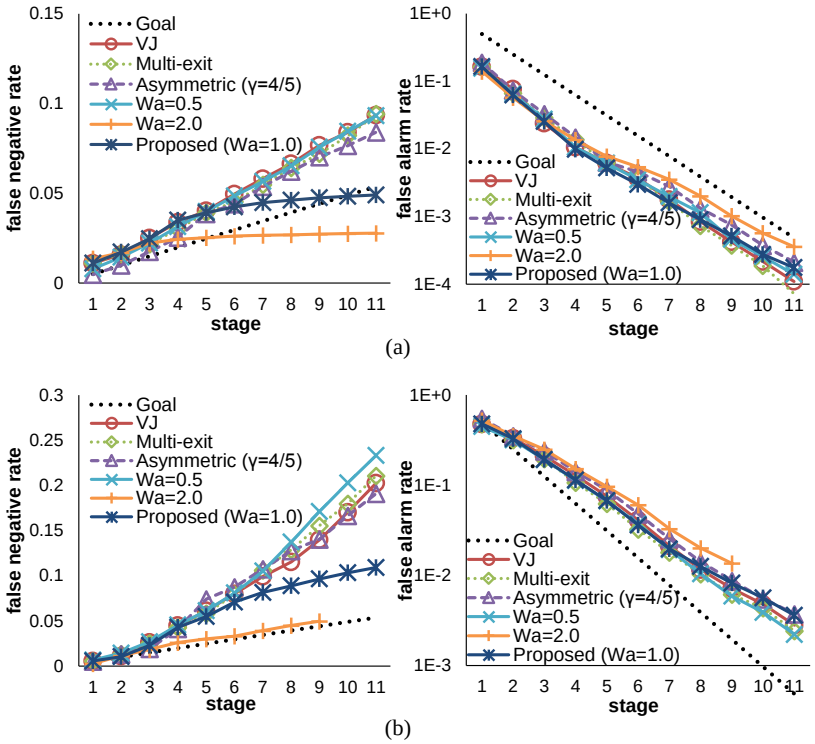


Figure 2. Learning curves of classifier cascades. (a) Face, (b) pedestrian.

 ISBN 978-80-86943-65-7

These experimental results confirm that the weight ratio should be balanced (W_a = 1) to achieve a higher detection rate without sacrificing the false alarm rate.

The figure also shows that the performance improvement gained by adopting the multi-exit cascade structure or the asymmetric AdaBoost is marginal. By adopting a sophisticated algorithm for training each classifier, the performance goal for each classifier can be achieved with less number of weak classifiers. However, the performance of the classifier cascade is hardly improved, since each classifier is trained just until a given goal is achieved.

4.2 Detection Performance Comparison

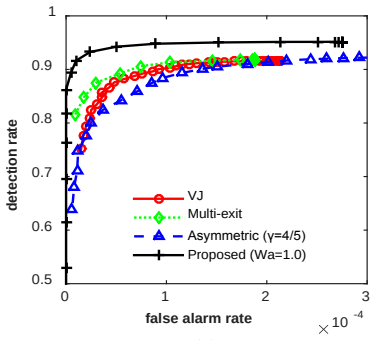
We compared the performance of the classifier cascades employing the proposed strategy against that of the classifier cascades employing the previous approaches. Fig. 3 shows the receiver operating characteristic (ROC) curves for the classifier cascades. Even though all the classifier cascades were trained to achieve the same performance goal with the same set of training examples and the same set of Haar-like features, those employing the proposed strategy show significantly improved detection rate at the same false alarm rate.

5. CONCLUSIONS

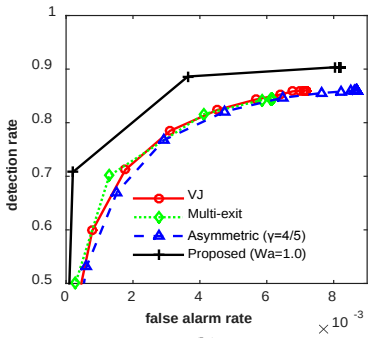
We propose a weight adjustment strategy to achieve an improved performance in cascade learning. Cascade learning often underachieves the detection rate goal even when it overachieves the false alarm rate goal due to overfitting. In the proposed strategy, the weight ratio of positive examples to negative examples is computed to evaluate the imbalance between the detection rate and the false alarm rate, and it is adjusted to be balanced to prevent cascade learning from overfitting.

Since both the detection rate and the false alarm rate definitely decrease as more classifiers are appended to the classifier cascade, maintaining a higher detection rate is far more important than achieving a lower false alarm rate in cascade learning. By adopting the proposed strategy, an improved performance can be achieved by preventing the degradation in the detection rate at later stages, which often occurs in cascade learning.

Experimental results confirm the effectiveness of the proposed strategy. For experiments, face and pedestrian classifier cascades were trained by employing previous approaches and the proposed strategy. Even though each classifier cascade was trained to achieve the same performance goal with the same set of training examples and the same set of features, the performance of the classifier cascades employing the proposed strategy is significantly improved.



(a)



(b)

Figure 3. ROC curves of classifier cascades. (a) Face, (b) pedestrian.

6. ACKNOWLEDGMENTS

This work was supported by the Center for Integrated Smart Sensors funded by the Ministry of Science, ICT & Future Planning as Global Frontier Project (CISS-2013M3A6A6073718).

7. REFERENCES

[Bou05] Bourdev, L. and Brandt, J., Robust object detection via soft cascade. In CVPR, vol. 2, pp. 236–243, 2005.

[Cev13] Cevikalp, H., Triggs, B., and Franc, V., Face and landmark detection by using cascade of classifiers. in FG, pp. 1–7, 2013.

[Che11] Cheng, W. C. and Jhan, D. M., A cascade classifier using Adaboost algorithm and support vector machine for pedestrian detection. in SMC, pp. 1430–1435, 2011.

[Cui10] Cui, J., Liu, F., Li, Z., and Jia, Z., Vehicle localisation using a single camera. in IV, pp. 871–876, 2010.

[Dol09] Dollar, P., Tu, Z., Perona, P., and Belongie, S., Integral channel features. in BMVC, pp. 1–11, 2009.

[Dol12] Dollar, P., Wojek, C., Schiele, B., and Perona, P., Pedestrian detection: an evaluation of the state of the art. TPAMI, vol. 34, no. 4, pp. 743–761, 2012.

[Dud01] Duda, R. O., Hart, P. E., and Stork, D. G., Pattern classification. 2nd ed., John Wiley & Sons, Inc., 2001.

[Enz09] Enzweiler, M. and Gavrilla, D. M., Monocular pedestrian detection: survey and experiments. TPAMI, vol. 31, no. 12, pp. 2179–2195, 2009.

[Fre97] Freund, Y. and Schapire, R. E., A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting. J. Comp. Syst. Sci., vol. 55, pp. 119–139, 1997.

[Fri00] Friedman, J., Hastie, T., and Tibshirani, R., Additive logistic regression: a statistical view of boosting. Ann. Stat., vol. 28, no. 2, pp. 337–407, 2000.

[Hoa12] Hoang, V., Vavilin, A., and Jo, K. H., Pedestrian detection approach based on modified Haar-like features and AdaBoost. In ICCAS, pp. 614–618, 2012.

[Hua07] Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E., Labeled faces in the wild: a database for studying face recognition in unconstrained environments. University of Massachusetts, Amherst, Technical Report 07–49, 2007.

[Kel11] Keller, C., Enzweiler, M., and Gavrila, D. M., A new benchmark for stereo-based pedestrian detection. in IV, pp. 691–696, 2011.

[Lan12] Landesa-Vázquez, I. and Alba-Castro, J. L., Shedding light on the asymmetric learning capability of AdaBoost. Pattern Recognit. Lett., vol. 33, no. 3, pp. 247–255, 2012.

[Li11] Li, J., Wang, T., and Zhang, Y., Face detection using SURF cascade. in ICCV Workshops, pp. 2183–2190, 2011.

[Liu12] Liu, S., Dong, Y., Liu, W., and Zhao, J., Multi-view face detection based on cascade classifier and skin color. in CCIS, vol. 1, pp. 56–60, 2012.

[Lud13] Ludwig, O., Nunes, U., Ribeiro, B., and Premebida, C., Improving the generalization capacity of cascade classifiers. IEEE Trans. Cybernetics, vol. 43, no. 6, pp. 2135–2146, 2013.

[Mas07] Masnadi-Shirazi, H. and Vasconcelos, N., Asymmetric boosting. in ICML, pp. 609–619, 2007.

[Pai14] Paisitkriangkrai, S., Shen, C., and van den Hengel, A., Asymmetric pruning for learning cascade detectors. IEEE Trans. Multmed., vol. 16, no. 5, pp. 1254–1267, 2014.

[Pha08] Pham, M.-T., Hoang, V.-D. D., and Cham, T.-J., Detection with multi-exit asymmetric boosting. in CVPR, pp. 1–8, 2008.

[Pri13] Prioletti, A., Møgelmose, A. Grisleri, P., Trivedi, M. M., Broggi, A., and Moeslund, T. B., Part-based pedestrian detection and feature-based tracking for driver assistance: real-time, robust algorithms, and evaluation. IEEE Trans. Intelli. Transport. Syst., vol. 14, no. 3, pp. 1346–1359, 2013.

[Sab12] Saberian, M. J. and Vasconcelos, N., Learning optimal embedded cascades. TPAMI, vol. 34, no. 10, pp. 2005–2018, 2012.

[Siv12] Sivaraman, S. and Trivedi, M. M., Real-time vehicle detection by parts for urban driver assistance. in ITSC, pp. 1519–1524, 2012.

[Sun98] Sung, K.-K. and Poggio, T., Example-based learning for view-based human face detection. TPAMI, vol. 20, no. 1, pp. 39–51, 1998.

[Sun07] Sun, Y., Kamel, M. S., Wong, A., and Wang, Y., Cost-sensitive boosting for classification of imbalanced data. Pattern Recognit., vol. 40, no. 12, pp. 3358–3378, 2007.

[Sun14] Sun, C., Hu, J., and Lam, K. M., A biased selection strategy for information recycling in boosting cascade visual-object detectors. Pattern Recognit. Lett., vol. 40, pp. 11–18, 2014.

[Vio01] Viola, P. and Jones, M., Fast and robust classification using asymmetric AdaBoost and a detector cascade. NIPS 14, MIT Press, pp. 1311–1318, 2001.

[Vio04] Viola, P. and Jones, M., Robust real-time face detection. IJCV, vol. 57, no. 2, pp. 137–154, 2004.

[Wan12] Wang, P., Shen, C., Barnes, N., and Zheng, H., Fast and robust object detection using asymmetric totally corrective boosting. IEEE Trans. Neural Networks Learn. Syst., vol. 23, no. 1, pp. 33–46, 2012.

[Wol11] Wolf, L., Hassner, T. and Taigman, Y., Effective face recognition by combining multiple descriptors and learned background statistics. TPAMI, vol. 33, no. 10, pp. 1978–1990, 2011.

[Wu04] Wu, B., Ai, H., Huang, C., and Lao, S., Fast rotation invariant multi-view face detection based on real AdaBoost. in AFGR, pp. 79–84, 2004.

[Wu08] Wu, J., Brubaker, S. C., Mullin, M. D., and Rehg, J. M., Fast asymmetric learning for cascade face detection. TPAMI, vol. 30, no. 3, pp. 369–382, 2008.

[Xia03] Xiao, R., Zhu, L., and Zhang, H., Boosting chain learning for object detection. in ICCV, vol. 1, pp. 709–715, 2003.

[Xin11] Xin, Y., Xiaosen, S., and Li, S., A combined pedestrian detection method based on Haar-like features and HOG features. in ISA, pp. 1–4, 2011.

A Robust Temporal Depth Enhancement Method for Dynamic Virtual View Synthesis

Can Liu
National ASIC System
Engineering Research Center
Southeast University
Nanjing, 210096, P.R.China
liucan@seu.edu.cn

Weizheng Zhang
Chien-Shiung Wu College
Southeast University
Nanjing, 210096, P.R.China
213113460@seu.edu.cn

Zhi Qi
National ASIC System
Engineering Research Center
Southeast University
Nanjing, 210096, P.R.China
q_zhi@yahoo.com

Longxing Shi
National ASIC System
Engineering Research Center
Southeast University
Nanjing, 210096, P.R.China
Lxshi@seu.edu.cn

ABSTRACT

Depth-image-based rendering (DIBR) is a view synthesis technique that generates virtual views by warping from the reference images based on depth maps. The quality of synthesized views highly depends on the accuracy of depth maps. However, for dynamic scenarios, depth sequences obtained through stereo matching methods frame by frame can be temporally inconsistent, especially in static regions, which leads to uncomfortable flickering artifacts in synthesized videos. This problem can be eliminated by depth enhancement methods that perform temporal filtering to suppress depth inconsistency, yet those methods may also spread depth errors. Although these depth enhancement algorithms increase the temporal consistency of synthesized videos, they have the risk of reducing the quality of rendered videos. Since conventional methods may not achieve both properties, in this paper, we present for static regions a robust temporal depth enhancement (RTDE) method, which propagates exactly the reliable depth values into succeeding frames to upgrade not only the accuracy but also the temporal consistency of depth estimations. This technique benefits the quality of synthesized videos. In addition we propose a novel evaluation metric to quantitatively compare temporal consistency between our method and the state of arts. Experimental results demonstrate the robustness of our method for dynamic virtual view synthesis, not only the temporal consistency but also the quality of synthesized videos in static regions are improved.

Keywords

FTV, DIBR, Temporal consistency, Depth enhancement

1. INTRODUCTION

Virtual view synthesis is being considered as an important component of 3D Video (3DV) [1, 2] and Free-viewpoint Television (FTV) [3, 4, 5] systems. Depth-image-based rendering (DIBR) [6, 7] is one of the widely accepted key techniques of view synthesis.

The synthesis quality of DIBR system highly depends on the accuracy of depth map. For static scenarios, depth estimation has been well-studied using images provided by standard datasets, such as the Middlebury dataset [8]. Numerous methods [9, 10, 11] have presented impressively excellent results of depth estimation as published on the website of Middlebury stereo evaluation [12]. However, for dynamic scenarios, the depth maps estimated using even the most powerful method may still have the problem of temporal inconsistency [13], which leads to flickering artifacts in the synthesized video. There is a much higher possibility that these flickering artifacts exhibit in the challenging areas containing static texture-less regions or static scenes with non-Lambertian surfaces.

In order to remove the flickering artifacts mainly in these static regions, some temporal depth enhancement methods have been proposed. It was shown that spatiotemporal bilateral filter introduced by Richardt *et al.* [14] improved the temporal consistency of depth sequences, but it would spread depth errors to neighboring regions or subsequent frames for overlooking depth reliabilities. As an improvement, Cheng *et al.* [15] presented a quad-lateral filter that took the depth reliability into consideration through measuring the difference of

depth values between two pixels. This work has achieved alleviated depth contamination in neighboring regions to a certain extent, whereas its pixel based processing was quite time consuming. This kind of correction of depth may improve the depth consistency over the entire image. However, it is unable to maintain the correct depth variations due to moving objects in dynamic regions.

The work by Richardt *et al.* [16] and Min *et al.* [17] utilized optical flow to supply motion cues for temporal depth filter, thus they sustained the correct depth variations in motion regions. However, the optical flow analysis suffered from its unaffordable computational expenses and poor performance in texture-less regions. To improve the efficiency of motion extraction, Fu *et al.* [18] proposed a light block-based motion detection algorithm to introduce motion cues into temporal depth filter, which deployed high weight to stationary pixels and low weight to motion pixels. This fast and effective work has been used as the standard temporal depth enhancement framework by Moving Pictures Experts Group (MPEG) to View Synthesis Reference Software (VSRS) [19]. However, Fu's motion detection suffered from inaccuracy and inflexibility problems caused by fixed local window size and thresholds.

The previous methods [16, 17, 18], which only took advantage of the texture, might fail to detect the motion regions when there contained much less texture information. Lu *et al.* [20] used high quality of depth information from depth sensors instead to support correct detection of the scene static structure. Consequently, they successfully preserved the depth variations of dynamic regions in the output of temporal depth filter. However, Lu's work has only been tested with high quality depth videos from Kinect and ToF, while the performance with depth videos obtained by stereo matching methods is to be verified.

Most of the previous methods concentrated on how to precisely separate motion regions from static regions and remained correct depth variations in motion regions. However, for static regions, their depth enhancement methods could not satisfy both temporal consistency and depth accuracy.

In conclusion, the current methods exhibit the following limitations: 1) they sacrifice the quality of depth for temporal consistency, or they cannot maintain temporal consistency and depth quality at the same time; 2) most of the algorithms are computationally too expensive to be real-time.

In this paper, we figure out a robust temporal depth enhancement (RTDE) method to improve depth quality especially in static regions, and keep the good results as temporally consistent as possible. First, we deploy an effective filtering strategy across the entire image based on motion block detection [21]. Given the motion detection, we then process depth values only in static regions and leave the correct depth variations in motion regions untouched. Second, in order to improve the robustness with regarding to the first limitation mentioned above, we introduce the depth reliability information as an important filtering weight into the temporal depth filter. Experimental results show that our method is fast and achieve satisfactory synthesized videos with regarding to both rendering quality and temporal consistency. In addition, inspired by the work of Fu *et al.* [18] and Solh *et al.* [22], we propose a simple and feasible evaluation metric to measure the temporal consistency quantitatively. We believe the similarity of Mean Square Difference (MSD) curves of adjacent frames is robust to illumination variations from different views, and reveals the degree of the temporal consistency when we compare the synthesized videos from the enhanced and original depth sequences.

The rest of this paper is organized as follows. Section 2 addresses the proposed robust temporal depth enhancement algorithm and temporal consistency evaluation metric. Experimental results are shown in Section 3, finally Section 4 concludes this paper.

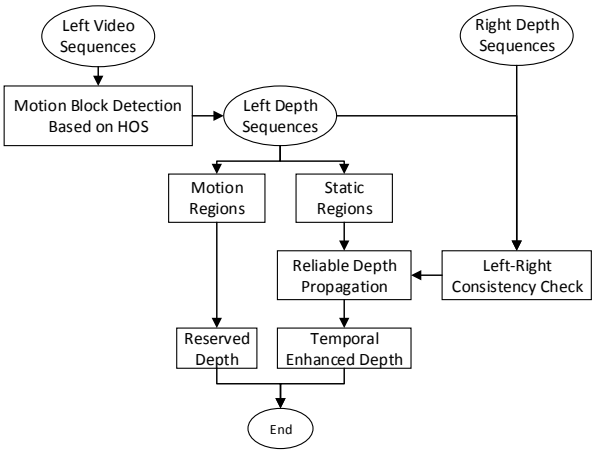


Figure 1: Enhancement flowchart of the left depth sequences

2. ALGORITHMS

In this paper, Depth Image-based Rendering (DIBR) deploys the framework that synthesizes the intermediate virtual view from both left and right reference views. Our proposed temporal enhancement method is a component of DIBR system as the depth preprocessing procedure. The flowchart illustrated in Figure 1 shows the preprocessing of left depth sequences, which is the same to the right depth sequences. Using motion block detection based on High-order Statistics (HOS) [21], we separate one input depth frame into static and motion regions. We only suppress depth inconsistency in static regions while maintain reasonable depth variations in motion regions. The depth values that pass through the

consistency check between the left and right images will be used to recover those spoiled depth values in the same static regions, not only in the current frame but also in succeeding frames. In such a manner, the reliable estimation of depth has been propagated.

The crucial steps of our proposed method will be elaborated subsequently. In the last part of this section, we will discuss the proposed objective evaluation metric for temporal consistency which is utilized in our experiments.

2.1 Motion detection based on HOS

As described in previous section, the depth inconsistency problem mainly occurs in the static regions, thus the motion detection should be conducted to supply motion cues to temporal depth filter for appropriate depth enhancement in static regions, without interfering reasonable depth variations in dynamic regions. Most current motion detection studies based on inter-frame difference, such as [18] and [25], may suffer from the inaccuracy and inflexibility in their results due to fixed local window size and thresholds. Because of the unavoidable misalignment between the depth map and color image [26], this triggers depth destruction after depth enhancement near moving object boundaries, as shown in Figure 2.

(a) (b)

Figure 2: Depth destruction near moving object boundaries caused by misalignment problem. In the 39th frame, the misalignment between the depth map and the color image, especially in the head and leg regions, as in shown in (a), leads to depth artifacts in the subsequent frame as shown in the yellow marked rectangle regions in (b).

Instead of using a nicely aligned motion segmentation result that is difficult and expensive to achieve, we prefer a bounding box (i.e. motion block) to separate the static regions from the dynamic areas in images. In this way, the misalignment problem can be effectively avoided and it also runs fast. Usually the accuracy of bounding box segmentation depends excessively on the threshold of inter-frame difference. Since a unified threshold is usually too coarse to capture the noise variation in video sequences, a step-forward processing on inter-frame difference map before binaryzation is expected to generate more accurate bounding box based motion detection.

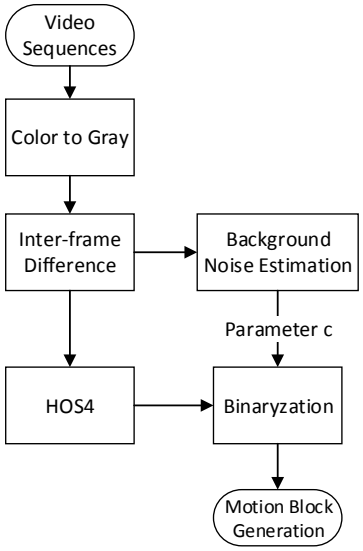
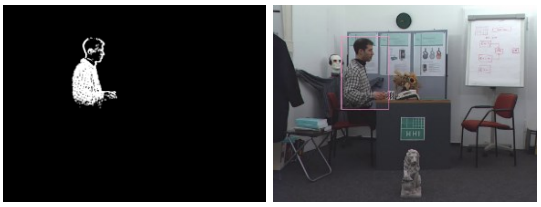


Figure 3: Workflow of motion detection based on HOS

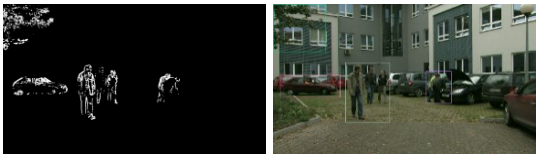
In our work, we use motion block detection based on HOS, which is adaptable to noise variations in video sequences thus it is able to receive precise motion segmentation results. The workflow is illustrated in Figure 3. The video sequences are firstly altered to gray-level images and then the inter-frame difference maps are calculated. Since the noise obeys Gaussian-distribution while moving objects have strong structure, which contributes to the high order statistics of frame difference. Consequently, we can accomplish motion block detection by separating the non-Gaussian signals form the Gaussian one. Considering the computation effort and accuracy, we calculate the 4-order moment of inter-frame difference as high order statistics (i.e. HOS4) and compare it with an adaptive threshold T, which is determined by the estimated background noise $\hat{\sigma}_d^2$ in static regions and a constant parameter c, i.e. $T = c(\hat{\sigma}_d^2)$. If the 4-order moment of a pixel is higher than T, we attribute this pixel to motion regions, and static regions on the contrary. The detected results of three datasets are shown in Figure 4. Additionally, parameter c and the calculation of the 4-order moment could be completed simultaneously, thus making motion detection more time-saving.



(a) Bookarrival

 ISBN 978-80-86943-65-7

(b) Pantomime



(c) Carpark

Figure 4: Detected motion blocks of three datasets

2.2 Robust temporal depth enhancement

The purpose of depth enhancement in static regions is to smooth depth map without causing any degradation in accuracy, however, this target is difficult to achieve. Most of the current methods [16][17][18], which achieve temporal depth consistency to a certain degree, overlook the reliability of depth values thus eventually lead to annoying perception, as illustrated in Figure 5.

2.2.1 Depth reliability check

Depth reliability is crucial to guarantee the consistent good quality of depth maps, especially in static regions, over the sequence of synthesized videos. Since it is unrealistic to judge the reliability of depth map using genuine depth information, in this paper we combine a left-right consistency checking (LRCC) method to detect then propagate the qualified depth values to successive frames.

Left-right consistency check is also called the two-view constraint [33]. It distinguishes outliers caused by occlusion, texture-less areas, and false match points. Assume that the disparity of a pixel p(x, y) in the left camera is $d_{LR}(x,y)$, and $d_{RL}(x,y)$ vice versa. The depth reliability mask is then defined as equation (1), where we set the threshold T to 1.

$$Mask(x,y)=\begin{cases}1, & |d_{LR}(x,y)\text{-}d_{RL}(x-d_{LR}(x,y),y)|<T\\ 0, & otherwise\end{cases} \quad (1)$$

2.2.2 Temporal depth filter

Depth values in the non-zero region of reliability mask that present good qualities are fed to our temporal depth filter to eliminate the depth contamination due to errors in neighboring area. This step is called Robust Temporal Depth Enhancement (RTDE). Specifically, our method handles three different conditions as described by the pseudo code of RTDE below.

In ***Case 1***, the depth pixel $\boldsymbol{p(n)}$ in the current frame $\boldsymbol{n}$ is reliable while the enhanced corresponding pixel in previous frame $\boldsymbol{p'(n-1)}$ is unreliable, in this occasion we reserve the depth value of current frame, therefore the high reliability depth value will not be contaminated by previous depth errors. In ***Case 2***, both the depth pixel $\boldsymbol{p(n)}$ and enhanced corresponding pixel $\boldsymbol{p'(n-1)}$ are detected to be reliable, the enhanced depth value of current frame $\boldsymbol{p'(n)}$ is calculated as the weighted sum of $\boldsymbol{p(n)}$ and $\boldsymbol{p'(n-1)}$, in this occasion we pledge the reliability of enhanced depth value and alleviate the negative effects caused by misdetection of LRCC. In ***Case 3***, the depth pixel $\boldsymbol{p(n)}$ in current frame $\boldsymbol{n}$ is detected as unreliable, we inherit the previous enhanced depth value $\boldsymbol{p'(n-1)}$, in this occasion we could guarantee both the consistency and quality of depth information.

for original depth pixel $\boldsymbol{p(x,y,n)}$ ***in current frame n, and*** $\boldsymbol{p'(x,y,n)}$ ***is the enhanced pixel of p, let*** $\boldsymbol{p(n)=p(x,y,n)}$, $\boldsymbol{p'(n)=p'(x,y,n)}$;

Case 1. ***if*** $\boldsymbol{p(n)}$ ***reliable &&*** $\boldsymbol{p'(n-1)}$ ***unreliable***

$\boldsymbol{p'(n)=p(n)}$;

end if

Case 2. ***else if*** $\boldsymbol{p(n)}$ ***reliable &&*** $\boldsymbol{p'(n-1)}$ ***reliable***

$\boldsymbol{p'(n)=\alpha\times p(n)+(1-\alpha)\times p'(n-1)}$;

end if

Case 3. ***else***

$\boldsymbol{p'(n)=p'(n-1)}$;

end if

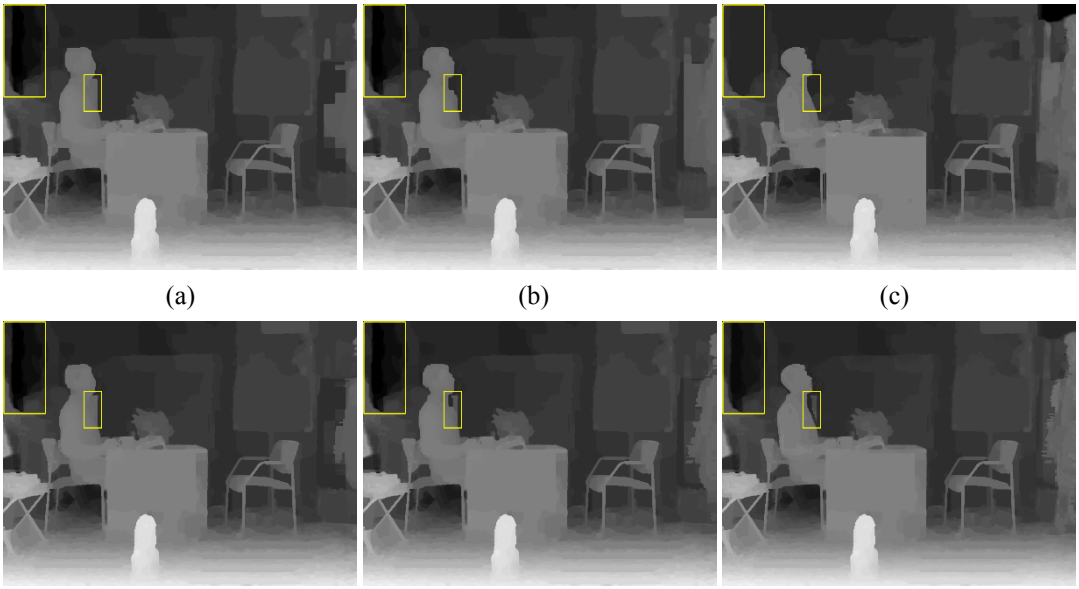
end for

In our enhancement scheme, the unreliable depth values in current frame are continuously replaced by reliable depth values in previous frames, in the meanwhile, the reliable depth values in current frame are preserved and propagated to the following frames to upgrade the depth quality of static regions. Moreover, the continuous delivery of dependable depth values used in ***Case 2*** and ***Case 3*** also suppress the fluctuation of depth values in the temporal domain. After being enhanced, the rectified depth maps will be utilized in further DIBR schemes.

Compared with other temporal enhancement methods, the proposed RTDE concentrates on both the temporal consistency and the depth quality improvement in static regions. The experimental results in Section 3 will demonstrate the robustness of our method.

2.3 Evaluation of Temporal Consistency

Current temporal consistency evaluation of the synthesized video is mainly prioritized to subjective evaluation in the absence of simple but efficient objective assessment metrics. Inspired by [18, 22], in this paper we introduce a novel objective metric to assess temporal consistency of the synthesized video. The idea is to compare the similarity of Mean Square Difference (MSD) curves of adjacent frames between



(a) (b) (c)

(d) (e) (f)

Figure 5: Depth quality degradation illustration. (a)-(c) corresponds to original depth for frame 9 to 11 of "Bookarrival", (d)-(f) corresponds to enhanced depth by TDE. As shown in (f), the yellow rectangle marked regions inherit errors of previous frames, leading to depth quality degradation.

Datasets	Frames	Resolution	View Synthesis (left, right → center)	c	T	α
Bookarrival	1→100	1024*768	(10,6) → 8	100	1	0.25
Pantomime	1→150	1280*960	(37,41) → 39	85	1	0.25
Carpark	1→100	1920*1088	(5,3) → 4	125	1	0.25

Table 1: Experimental parameters

the synthesized video and the original one. The temporal consistency is satisfying when two MSD curves are coherent. The similarity is measured as the Standard Deviation (STD) of differences of MSD values.

Assume $I_{ori}(x,y,k)$ and $I_{syn}(x,y,k)$ represent the intensity of pixel (x,y) in the original and synthesized frame respectively, k is the frame number. The MSD of $I_{ori}(x,y,k)$ and $I_{syn}(x,y,k)$ is determined by equation (2) and (3), where m, n represent the width and height of selected evaluation region.

$$MSD_{ori}(k) = \frac{1}{mn}\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}|I_{ori}(x,y,k) - I_{ori}(x,y,k-1)|^2 \quad (2)$$

$$MSD_{syn}(k) = \frac{1}{mn}\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}|I_{syn}(x,y,k) - I_{syn}(x,y,k-1)|^2 \quad (3)$$

The Temporal Inconsistency Index (TII), or the similarity of MSD curve is determined by equation (4), where STD is determined by the difference of MSD, written in expression (5).

$$TII = \frac{1}{Similarity} = STD(Differenece\ of\ MSD) \quad (4)$$

$$Difference\ of\ MSD(k) = MSD_{ori}(k) - MSD_{syn}(k) \quad (5)$$

As described in equation (4), lower STD value indicates higher similarity and better temporal consistency of synthesized video. Experimental results in next section will demonstrate that our objective evaluation metric accesses the temporal consistency of synthesized video as efficiently as subjective human perceptions do.

3. EXPERIMENTAL RESULTS

In this section, we compare the proposed RTDE method with the temporal depth enhancement (TDE) method by Fu *et al.* [18] that is implemented in MPEG-VSRS. We test out method on the databases of 'Bookarrival' [27], 'Carpark' [28], and 'Pantomime' [29]. To our knowledge, the depth maps of 'Bookarrival' and 'Carpark' are generated using DERS [30], while 'Pantomime' is generated by the depth estimation software [31] from Nagoya University. All three datasets share the problem of temporal inconsistency in their depth sequences, especially in static regions, which leads to flickering artifacts in synthesized video sequences. To synthesize video sequences, we deploy the software VSRS provided by MPEG. Parameters of our experiment are concluded in Table 1, where **c** is the constant parameter in motion block detection based on HOS(see in Section 2) to adjust noise threshold. **T** is the parameter in LRCC (see in Equation 1). And **α** is the parameter in RTDE (see in Figure 5).

(a) (b) (c)

Figure 6: Temporal depth enhancement in static regions. (a) The original depth map without being processed. (b) The enhanced result using TDE. (c) Our result.

To evaluate our test results, we assess both rendering quality and temporal consistency of synthesized videos. Videos from multiple cameras may suffer from uneven illumination in the environment, thus making PSNR an unstable quality measurement for its sensitivity to variances of illumination. In our case, we combine SSIM [32] with PSNR as the assessment criteria. The comprehensive judgment with both criterions provides a fair enough measurement of similarity between the reference view and the virtual view regardless of annoying illumination condition. For temporal consistency, we use the proposed Temporal Inconsistency Index (TII), which is defined in equation (4).

3.1 The evaluation in static regions

In this section, we mainly analyze the robustness of our method in static regions. In the next subsection, we will evaluate the performance of our method across the entire image.

Figure 6 illustrates the specific results of depth processing in static regions of dataset Bookarrival (frame No. 79). In this frame depth errors in the marked static regions are inherited by TDE method. We achieve better results than TDE in static regions since we replace the unreliable depth values in the current frame by previous reliable depth values.

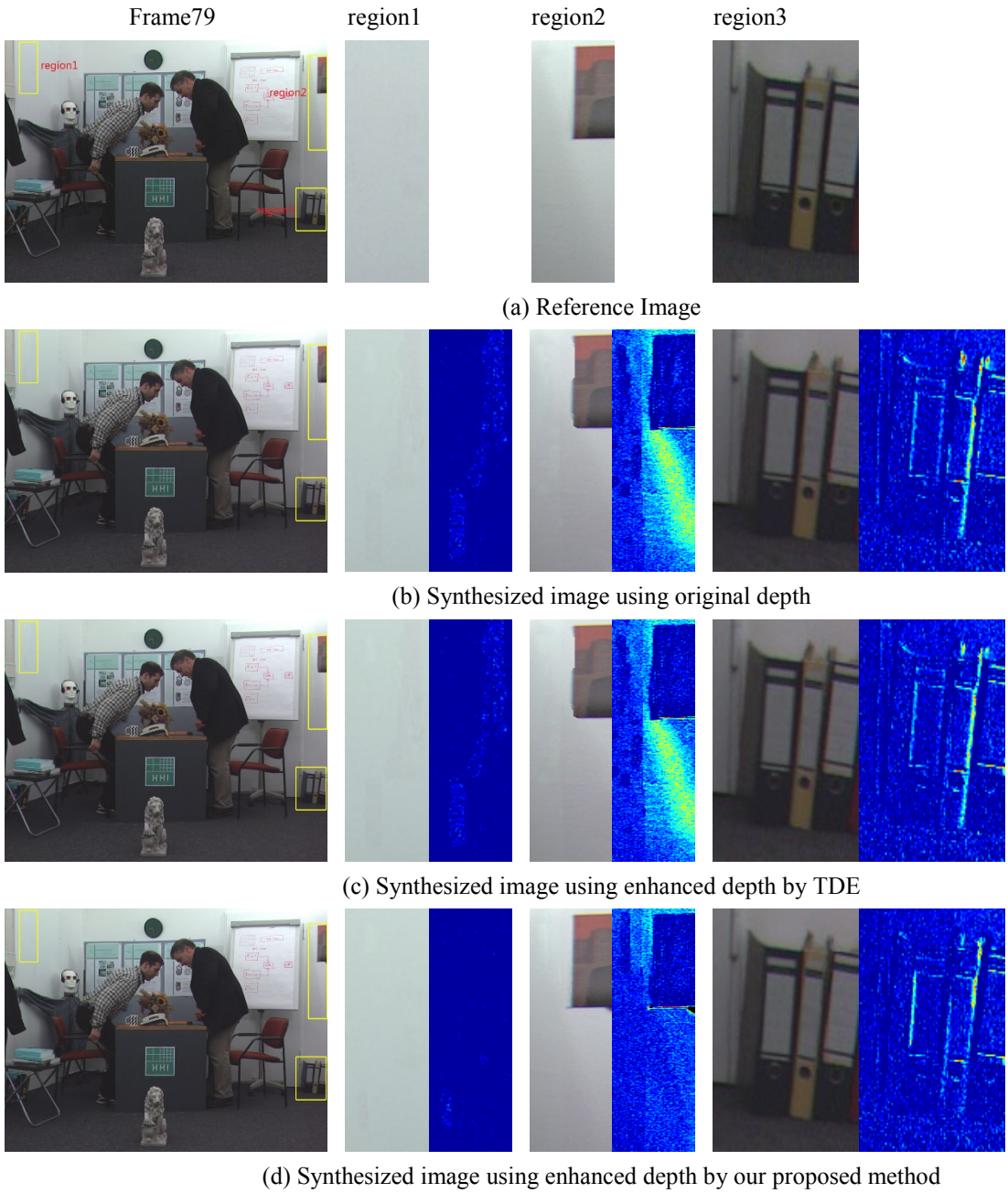
Figure 7 illustrates the qualitative analysis of synthesis quality in static regions. We separate the entire image into several regions, among which we mainly concentrate our analysis on region 1, 2, and 3. The reason we isolate these static regions is that the depth values in these regions share an apparent fluctuation, i.e. depth inconsistency in timeline, thus they are suitable to demonstrate the capability of our analysis. In Figure 7(c), the synthesized result is unsatisfying and similar to the result using original depth information. This is because TDE inherits errors from previous frames shown in Figure 6. In the proposed method, we replace the poor depth estimation with the reliable depth information from previous frames, therefore our results are better in quality, both perceptually and numerically (see Table 2).

Figure 8 illustrates the temporal consistency in static regions of the synthesized video, where x axis represents frame number and y axis indicates the Difference of MSD defined in Equation (5). The distortion in y axis represents inconsistency of video sequences. For region 1 in Figure 8(a), the inconsistency problem of original depth sequences mainly occurs in frame 11, 51 and 81 to 83, thus causing flickering artifacts in the synthesized video. After being processed by TDE, the fluctuation of depth is weakened, while inconsistency still remains. Since TDE just smooths the depth sequences which only reduces the range of depth fluctuation. In our proposed method, the fluctuation could be reduced to a great extent by continuously delivering reliable depth in the temporal domain. In Figure 8(a), the inconsistency around frame 11 cannot be suppressed, because depth values in region one are not reliable in preceding frames. In the case of the first condition when applying the temporal depth filter, the reliable depth values in frame 11 are preserved, furthermore, they are propagated to the subsequent frames. For region 2 and region 3, we only analyze frame 40 to 100 since obvious motion appears in the previous 39 frames, the results are shown in Figure 8(b) and (c). Overall, our proposed method keeps the best temporal consistency compared with other methods in static regions. The statistical data collection shown in Table 2 tells us that our method provides a better quality, which ensures the temporal consistency in static regions.

In Table 2, we analyze quantitatively both the rendering quality and the temporal consistency in three different static regions as presented in Figure 7. It is shows that our method obtains much better results than those of the original method or TDE.

3.2 The evaluation across the entire image

Although RTDE is not deployed to dynamic regions in this paper, experimental results indicate that our method achieves the best trade-off between the rendering quality and the temporal consistency across the entire image among synthesized results generated



(a) Reference Image

(b) Synthesized image using original depth

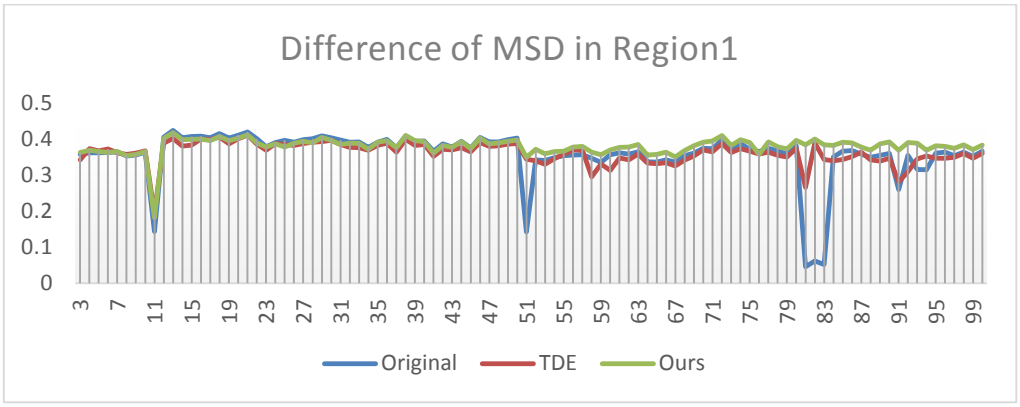
(c) Synthesized image using enhanced depth by TDE

(d) Synthesized image using enhanced depth by our proposed method

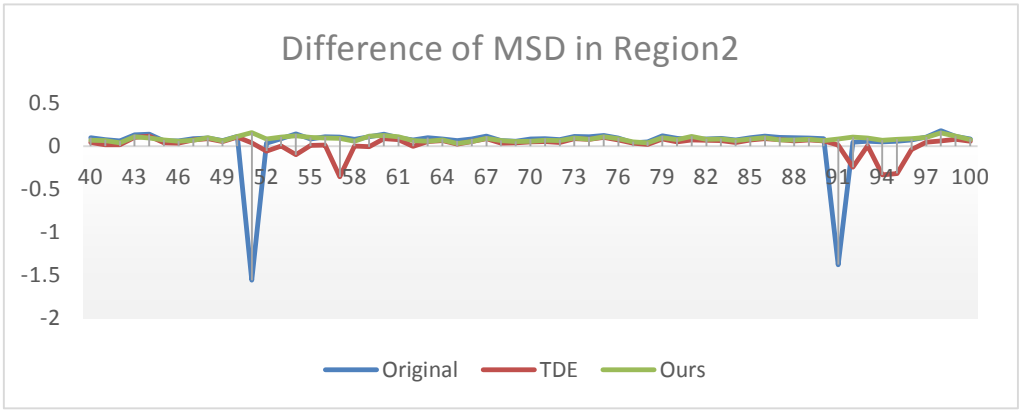
Figure 7: Qualitative analysis of synthesis quality in the static regions. (a) represents the reference image of Frame 79. (b) is the synthesized result using original depth. (c) indicates the synthesized image using TDE. (d) is the result of our method. The 2th, 4th and 6th columns show the synthesized results of local static regions. The error heatmaps between synthesized results and reference images are displayed in the 3th, 5th and 7th columns . Warm colors mean large errors, and cold colors mean small errors.

by original depth and enhanced depth from TDE (see Table 3).

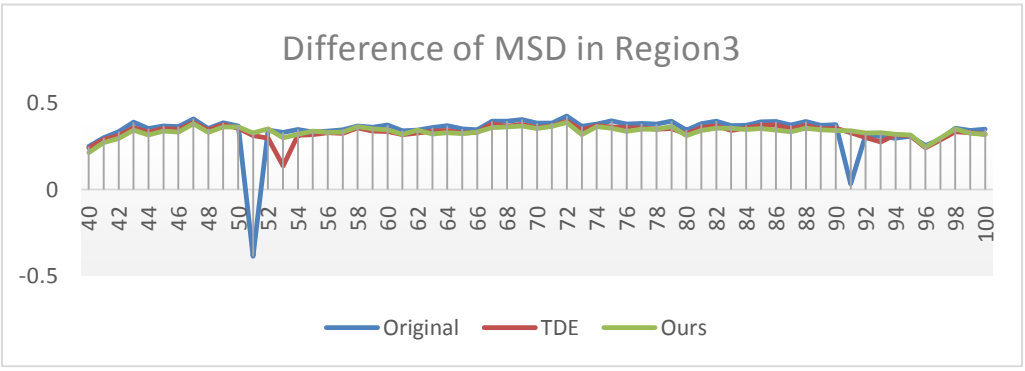
In Table 3, the temporal consistency of our proposed method in Bookarrival and Carpark datasets are a little lower than TDE but higher than the original one. This is because RTDE focuses merely on static regions, while TDE concentrates on the entire image so the temporal consistency in motion regions will also be improved. However, TDE provides the worst synthesis quality in Bookarrival and Carpark datasets, because the inaccuracy of motion detection and misalignment problem will degrades depth quality in motion regions or near the boundaries of moving objects. On the contrary, our RTDE could provide the best synthesis quality compared with the other two methods across the entire image of three datasets.



(a)



(b)



(c)

Figure 8: Temporal consistency evaluation in the static regions

Bookarrival	Region1(1-100)			Region2(40-100)			Region3(40-100)		
	Original	TDE	Ours	Original	TDE	Ours	Original	TDE	Ours
PSNR	49.08307	49.82602	**50.20146**	30.57667	30.74217	**31.55368**	34.54613	34.34882	**35.09781**
SSIM	0.946841	0.947205	**0.947711**	0.886019	0.900981	**0.936692**	0.920664	0.923001	**0.925322**
TII	0.068823	0.031144	**0.025312**	0.281997	0.098828	**0.025767**	0.109639	0.040748	**0.029024**

Table 2: Quantitative evaluation of synthesis quality and temporal consistency in the static regions. The PSNR and SSIM data are the average values of all frames. It should be noted that lower TII values represent higher temporal consistency and the bold figures indicate the best results among three methods.

 ISBN 978-80-86943-65-7

	Bookarrival			Pantomime			Carpark		
	Original	TDE	Ours	Original	TDE	Ours	Original	TDE	Ours
PSNR	35.76804	35.76522	**35.92959**	39.48054	39.53301	**39.76341**	30.94377	30.89608	**30.96458**
SSIM	0.923868	0.923806	**0.923939**	0.960488	0.960489	**0.963027**	0.890607	0.889084	**0.890761**
TII	0.053621	**0.029822**	0.035902	0.122023	0.101601	**0.095132**	0.297778	**0.058146**	0.128696

Table 3: Quantitative evaluation of synthesis quality and temporal consistency across the entire image

Moreover, for the dataset of Pantomime, our method keeps the best rendering quality as well as the temporal consistency. Because, there exists tremendous depth fluctuation in static regions of Pantomime. TDE only reduces the range of depth fluctuation, while our method could suppress them to a great extent.

4. CONCLUSIONS

In this paper, we proposed a robust temporal depth enhancement method for dynamic virtual view synthesis, which includes an effective and efficient motion block detection and a robust temporal depth filter aided by depth reliability check. Experimental results prove the robustness of our method that temporal consistency and synthesis quality can be both improved in static regions. Moreover, we proposed a comprehensive objective evaluation metric which is efficient and reasonable in assessing the temporal consistency of synthesized video sequences. However, as described in previous sections, in this paper we concentrate our method merely in static regions and only temporal filtering is conducted. In the future, we will extend our idea and method to motion regions and a spatiotemporal filter will be utilized to enhance depth sequences. Additionally, the accuracy of depth reliability check will also be considered.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (Grant No.61204023, Grant No.61203251 and Grant No. 61404028).

REFERENCES

[1] A. Redert, M.O. de Beeck, C. Fehn, W. Ijsselsteijn, M. Pollefeys, L. Van Gool, E. Ofek, I. Sexton, and P. Surman, Advanced three-dimensional television system technologies, 3D Data Processing Visualization and Transmission, 2002. Proceedings. First International Symposium on, 2002, pp. 313-319.

[2] C. Zhu, Y. Zhao, L. Yu, and M. Tanimoto, 3D-TV System with Depth-Image-Based Rendering, Springer New York, 2013.

[3] T. Fujii, and M. Tanimoto, Free viewpoint TV system based on ray-space representation, ITCom 2002: The Convergence of Information Technologies and Communications, International Society for Optics and Photonics, 2002, pp. 175-189.

[4] M. Tanimoto, Overview of free viewpoint television. Signal Processing: Image Communication 21 (2006) 454-461.

[5] Y. Mori, N. Fukushima, T. Yendo, T. Fujii, and M. Tanimoto, View generation with 3D warping using depth information for FTV. Signal Processing: Image Communication 24 (2009) 65-72.

[6] C. Fehn, A 3D-TV approach using depth-image-based rendering (DIBR), Proc. of VIIP, 2003.

[7] C. Fehn, Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV, Electronic Imaging 2004, International Society for Optics and Photonics, 2004, pp. 93-104.

[8] D. Scharstein, and R. Szeliski, A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International journal of computer vision 47 (2002) 7-42.

[9] X. Mei, X. Sun, M. Zhou, H. Wang, and X. Zhang, On building an accurate stereo matching system on graphics hardware, Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on, IEEE, 2011, pp. 467-474.

[10] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz, Fast cost-volume filtering for visual correspondence and beyond, Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 3017-3024.

[11] Q. Yang, A non-local cost aggregation method for stereo matching, Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 1402-1409.

[12] http://vision.middlebury.edu/stereo/eval/

[13] R. Khoshabeh, S.H. Chan, and T.Q. Nguyen, Spatio-temporal consistency in video disparity estimation, Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, IEEE, 2011, pp. 885-888.

[14] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N.A. Dodgson, Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid, Computer Vision–ECCV 2010, Springer, 2010, pp. 510-523.

[15] C. Cheng, S. Lin, and S. Lai, Spatio-temporally consistent novel view synthesis algorithm from video-plus-depth sequences for autostereoscopic displays. Broadcasting, IEEE Transactions on 57 (2011) 523-532.

[16] C. Richardt, C. Stoll, N.A. Dodgson, H.P. Seidel,

and C. Theobalt, Coherent spatiotemporal filtering, upsampling and rendering of RGBZ videos, Computer Graphics Forum, Wiley Online Library, 2012, pp. 247-256.

[17] D. Min, J. Lu, and M.N. Do, Depth video enhancement based on weighted mode filtering. Image Processing, IEEE Transactions on 21 (2012) 1176-1190.

[18] D. Fu, Y. Zhao, and L. Yu, Temporal consistency enhancement on depth sequences, Picture Coding Symposium (PCS), 2010, IEEE, 2010, pp. 342-345.

[19] M. Tanimoto, T. Fujii, and K. Suzuki. View synthesis algorithm in view synthesis reference software 3.5 (VSRS3. 5) Document M16090, ISO/IEC JTC1/SC29/WG11 (MPEG). 2009.

[20] S. Lu, N.N. King, L. Chern-Loon, and L. Songnan, Online Temporally Consistent Indoor Depth Video Enhancement via Static Structure. Image Processing, IEEE Transactions on 24 (2015) 2197-2211.

[21] L.M. Garth, and H.V. Poor, Detection of non-Gaussian signals: A paradigm for modern statistical signal processing [and prolog]. Proceedings of the IEEE 82 (1994) 1061-1095.

[22] M. Solh, G. AlRegib, and J.M. Bauza, 3VQM: A vision-based quality measure for DIBR-based 3D videos, Multimedia and Expo (ICME), 2011 IEEE International Conference on, Barcelona, Spain, 2011, pp. 1-6.

[23] S.D. Cochran, and G. Medioni, 3-D surface description from binocular stereo. IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (1992) 981-994.

[24] P. Fua, A parallel stereo algorithm that produces dense depth maps and preserves image features. Machine vision and applications 6 (1993) 35-49.

[25] S. Lee, and Y. Ho, Temporally consistent depth map estimation using motion estimation for 3DTV, International Workshop on Advanced Image Technology, 2010.

[26] X. Xu, L. Po, K. Cheung, L. Feng, and C. Cheung, Watershed based depth map misalignment correction and foreground biased dilation for DIBR view synthesis, Image Processing (ICIP), 2013 20th IEEE International Conference on, IEEE, 2013, pp. 3152-3156.

[27] I. Feldmann, *et al.* HHI Test Material for 3D Video. ISO/IEC JTC1/SC29/WG11, M15413, April 2008.

[28] ftp://multimedia.edu.pl

[29] http://www.fujii.nuee.nagoya-u.ac.jp/multiview-data/mpeg/mpeg_ftv.html

[30] M. Tanimoto, T. Fujii, M. P. Tehrani, M. Wildeboer Depth Estimation Reference Software (DERS) 4.0, ISO/IEC JTC1/SC29/WG11, MPEG 2008/M16605, 2009.

[31] M. Tanimoto, T. Fujii, and K. Suzuki. Reference software of depth estimation and view synthesis for FTV/3DV. ISO/IEC JTC1/SC29/WG11 M 15836, 2008.

[32] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity. Image Processing, IEEE Transactions on 13 (2004) 600-612.

[33] C. Georgoulas, G.C. Sirakoulis, and I. Andreadis, Real-time Stereo Vision Applications, INTECH Open Access Publisher, 2010.

Corrosion Rendering: Fusing Simulation and Photo-texturing

Nisha Jain
IIT Delhi
Hauz Khas, Delhi
110016, India
nisha@cse.iitd.ac.in

Prem Kalra
IIT Delhi
Hauz Khas, Delhi
110016, India
pkalra@cse.iitd.ac.in

Subodh Kumar
IIT Delhi
Hauz Khas, Delhi
110016, India
subodh@cse.iitd.ac.in

Abstract

We present a technique for realistic rendering of corroded objects. We employ a physio-chemically based stochastic model to determine the deterioration level of different points on an object, given its material characteristics and the vigor of the environment. Guided by values from the ISO standard, our model predicts shape degradation. This shape degradation is then applied to the object in the form of surface displacements and weathered appearance. The appearance degradation is hard to physically model accurately due to its dependence on a large number of unknown parameters as well as its high sensitivity to errors in modeling them. Hence, we instead sample from photographs of real objects to generate similar appearance for the rendered surface, but consistent with the simulated corrosion levels. We demonstrate our technique using several simulation results as well as different input photographs. We also evaluate the fidelity of the generated output to the simulation as well as to the sample texture patterns and validate our work with the help of data published in the corrosion literature. Our framework is generic and can be extended to a variety of corrosion scenarios. Ours is an important step towards predictive analysis of material loss and weathering phenomena for real objects.

Keywords

Corrosion rendering

1 INTRODUCTION

Corrosion is a stochastic process impacted by non-linear combination of factors like material property, environment, exposure time, etc. The complexity of corrosion makes it difficult to accurately predict the complete state of weathering objects, or even to measure all the causal factors. Simulation and rendering of weathered 3D objects is necessary for many application, games, movies, aesthetic design, and even unweathering of already weathered objects. Predictive models of weathering can also be used to estimate structural damage. We extend the model presented in [20] to general weathering and demonstrate its effectiveness in estimating corrosion. Such estimation of the physio-chemical state of an object undergoing corrosion has applications across diverse fields such as arts, engineering, aerospace, etc. Predictive analysis for an object undergoing corrosion is an important part of taking protective measures against both appearance and shape damages.

Corrosion can be decomposed into two basic building blocks including acceleration process due to an auto-catalytic chemical reaction followed by deceleration due to physical formation of semi-passive porous layer. Corrosion as a process results in both state change resulting in aesthetic degradation as well as material loss resulting in structural damage of the exposed layer. We model corrosion as a stochastic process influenced by the probabilities dependent on material and environmental conditions. The simulation works on a voxelization of the input object undergoing corrosion. In addition to our model being agnostic to the object size, shape intricacies and voxel resolution, the in-built continuous functions ensure that the model structure itself is generic and independent of specific material type or environment corrosivity, allowing them to be provided as inputs. The model allows each voxel to progressively reach higher corrosion states before being ultimately removed, exposing inner layers. To replicate actual corrosion process we also ensure the additive effect of highly corroded neighborhood and decelerating impact caused by creation of semi-porous layer of inert oxides.

Because of applications of corrosion study across domains, there has been significant effort to measure and simulate corrosion reported in the literature [14]. How-

 ISBN 978-80-86943-65-7

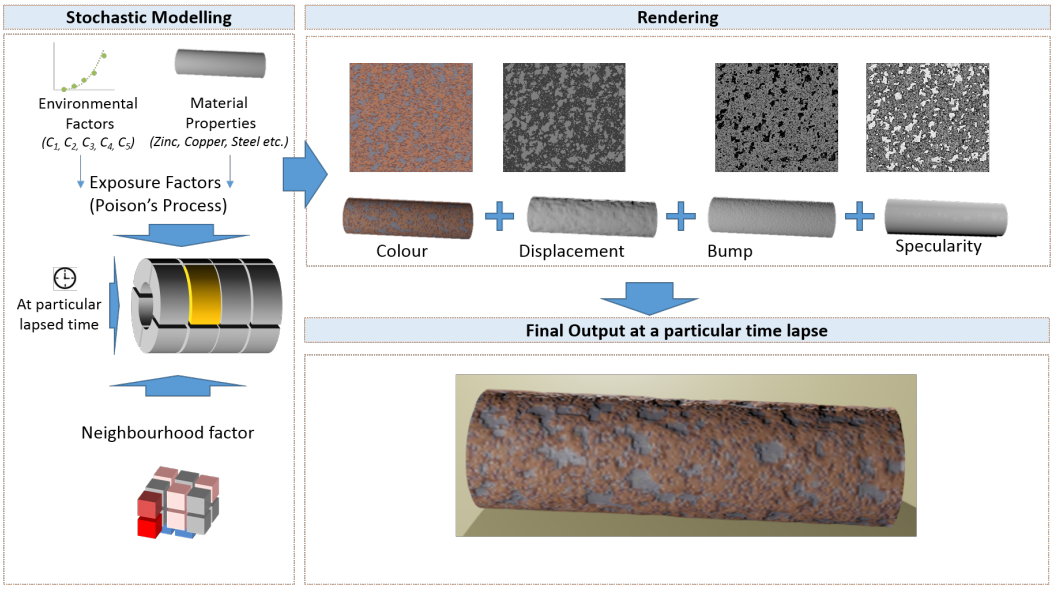


Figure 1: Simulation and Rendering Pipeline : The first block is the stochastic model, which includes voxelization and the corrosion simulation. The second block is the rendering pipeline, which generates different maps and combines them to create the final rendered corroded object.

ever, an accurate prediction of the shape of a corroded object has remained elusive. On the other hand, effects like failure time, pit depth, material loss etc., over time have been measured and simulated [21]. We extend the pitting corrosion model of Jain et al. [20] to compute the shape of the object at any given time. The simulation is driven by measured material loss included in ISO standard [14]. We demonstrate that data from the ISO standard can be used to effectively guide the stochastic parameters of the model to ensure that the predicted material loss closely follows the expected material loss. At the same time, our robust rendering framework ensures that the resulting shapes are realistic. The simulation additionally produces the degree of corrosion (normalized between 0 and 1) for every point on the surface at a given time interval. In order to create a realistic rendering, the reflectance properties of each corroded point needs to be derived.

Optical properties of the surface can vary in largely unpredictable fashion due to corrosion. Mostly in game development tools the textures are hand modeled and do not incorporate the actual corrosion process. Also, coexistence of multiple dynamic states due to the material's interaction with the environment results in complex and dynamic appearance, which a real photograph can best capture. Hence, instead of simulating it physio-chemically, we sample the appearance from parts of real photographs of other corroded objects, assuming that they exhibit the appearance of a variety of differently – some slight, others significant – corroded areas. Using these areas as examples then, we generate the appearance of the simulated object. We first normalize the photographs and subtract the effect of lighting to derive the underlying albedo [5]. Similar to [24], we then allow a user to mark corrosion degrees for a few pixels in the albedo map. The appearance manifold technique of [24] is then employed to derive the corrosion degree of all the un-marked pixels. This albedo map is then used to create reflectance values for the corroded material. This novel integration of physical simulation and appearance synthesis produces plausible results.

Ours is a more holistic approach than, say, predicting only material thickness or material loss, given a particular combination of factors (corrosion level and time). It predicts actual shape changes as well as the degree of corrosion, before imposing the reflective properties of the corroded material that is consistent with the corrosion.

2 RELATED WORK

Corroded objects have compound construction and their structure dynamically changes under the effect of varied physio-chemical conditions. These physio-chemical conditions are influenced by the interaction of changing environment and the deteriorating object. This results in making corrosion a complicated stochastic process where even a given material may deteriorate differently given its state and environmental conditions.

Given its relevance in varied fields including civil engineering, aerospace, heavy industries and art, extensive work has been done to understand corrosion phenomena [10] [16]. Though these work try to predict physical loss of material, not much work has been done on es-

timating structural in conjunction with aesthetic degradation.

Corrosion damages can be broadly categorized into localized and uniform corrosion [23]. Pitting corrosion is a severe form of localized corrosion that causes small cavities on the surface. Jain et al. present a model for pitting corrosion [20], which we generalize in this paper. Their work is limited to pitting corrosion with no notion of real time, resulting in lack of comprehensive predictive modeling. Our basic model uses there model as core but is tightly coupled to real time, thus opening the possibility of predictive analysis of corrosion for major materials including steel, copper and zinc.

We simulate uniform corrosion to validate our results given the lack of weight loss data for other forms of corrosion. Uniform corrosion is a generic form of corrosion where the attack proceeds evenly throughout the surface. There are a few models for simulation of general corrosion. Guessasma et al. [11] simulate the corrosion phenomena under potentiostaic conditions to the behavior of the process. They generate the simulation on a 3D grid and study the current density and exposed area. They simulate the corrosion process but do not produce real shape degradations.

There have been other studies on corrosion of different materials [9] [22]. It is observed that different materials behave differently and the subsequent surface degradation is discrete. Mérillou et al. [19] predict the changes due to corrosion with time. They best fit the ISO standard [14] [15] experimental data for steel and use the weight loss information to remove material from a three dimensional corrosion map grid. They do not simulate the actual process on a three dimensional object. Also they modify the BRDF model to affect the porosity and roughness of the surface in the final shading model. This is not based on any physical evidence. Our model gathers all the degradation effects from the actual stochastic simulation of the corrosion phenomena.

General corrosion analysis consists of predicting the extent of attack with time, under specific atmospheric conditions. ISO standards [15] list the weight loss in grams per meter square for specific materials under certain atmospheric conditions over time. They categorize environment corrosivity levels and catalog material weight loss under different conditions. We validate our results by generating similar weight loss from objects in our simulation.

Realistic rendering of corroded objects is vital in the field of computer graphics [7] [6][8]. Wang et al. [24] produce the optical appearance of a weathered surface by generating an "appearance manifold". They apply appearance sampling, but only approximately follow the general trend of corrosion variance across the surface. Also they use a 7D feature vector which is cumbersome to gather. They apply stochastic models to deteriorate color but are unrelated to the actual corrosion process. Bandeira et al. [1] create weathering effects based on chroma and luminance values. They introduce the concept of appearance maps similar to [24] but they do not address rough geometry variations. Their method works only on images and they do not present any three dimensional procedure. Hwang et al. [13] propose a method for creating a weathering gallery based on time dependent appearance manifolds(TDAMs). They create these TDAMs from sample videos clips. The major difficulty in their proposed method is collection of sample videos which covers presents a good weathering phenomena with least change in camera position and illumination characteristic.

Textures have also been used [26] to capture realistic surface effects in this domain, γ-ton tracing [3], is a particle based surface-centric model of corrosion. It does not handle severe geometry degradation that follows corrosion statistics. Clément et al. [4] generate aging textures by taking as input a target aging mask from the user. They employ elimination and reproduction techniques for producing the final texture. Their process is simple but requires a lot of user intervention and thus the user needs significant domain knowledge to achieve realistic results. Wojtan et al. [25] generate weathering by increasing or decreasing the surface of the object but the results produced are quite synthetic and do not appear real. Kamata et al. [17] describes a model for texture regeneration of peeling of preservative coatings of surfaces. They show the effects of peeling using geometric and environmental maps. Their model produces this effect by striking the surface with water drops and calculating the amount of accumulated droplets on the surface which leads to peeling. Neither do they incorporate the actual corrosion phenomena nor do they have a real time assessment of the corrosion process.

We propose a time variant voxel-based model to simulate and render generic corrosion on different materials. Our model produces geometric results based on the stochastic simulation and color changes based on texture synthesis, onto the object to produce photo-realistic effects. We validate our results based on ISO standard [15]. The validation is strengthened by observing the weight loss from different materials at different time-steps, which is similar to the standard. We demonstrate surface degradation with respect to time for different materials.

3 MODELING OF CORROSION

Corrosion is the deterioration of an object with specific material properties under dynamic environmental condition. Due to the dynamic change in intensity as well as the number of factors influencing the shape and aes-

thetics of an object undergoing corrosion, it is hard to accurately predict object state with elapsed time.

Jain et al. [20] simulate *pitting* corrosion as a two-step stochastic process. The first step nucleates pits and the second step grows them. The division of simulation into these two steps is based on the physio-chemical nature of the pitting corrosion process. Their model is qualitative in nature and does not account for the real weight loss in the object after a given time elapses. There is no real-time scale. Their stochastic model of corrosion growth, however, is inspired by the physio-chemistry of corrosion process. We improve their model in this paper and apply it to general corrosion with real time scale.

In order to introduce the notion of real time-scale, we take recourse to the standardized observations of the corrosion phenomena [15] to predict the actual weight loss in a given time for a particular material in a type of environment. We employ the stochastic aging process to obtain a single step corrosion model and modulate it to produce the observed weight losses in a given time period as cataloged in the ISO standard.

3.1 Corrosion Rate Categorization

Atmospheric condition such as rural, urban, industrial, marine, chemical, etc., varies and is one of the major contributors to corrosion. The complex inter-play of various factors such as metallic properties, environmental factors and operating conditions make accurate prediction of detailed corrosion behavior of different materials nearly impossible to track. The ISO standard [15] classifies the character of corrosion attack into five "corrosivity" levels as shown in Table 1. Corrosivity is a measure of the ability of the atmosphere to cause corrosion in a given corrosion system. For different materials (e.g., Steel, Copper, Aluminium, etc.) exposed to each class, ISO lists the average weight lost per unit surface area in each year of exposure, We call this the "corrosion rate."

Corrosivity	1 year	5 years	10 years
Very Low (C1)	10	23	33
Low (C2)	200	464	668
Medium (C3)	400	928	1334
High (C4)	650	1508	2167
Very High (C5)	1500	3480	5001

Table 1: Corrosion rate for steel for different standardized corrosivities [15] for different exposure times

Table 1 shows the corrosion rate in grams per m^2 of steel when exposed to different standardized corrosivities [15] for different exposure times. By fitting a function to the table entries, we can compute the amount of material lost in different environment types (namely, corrosivity categories C1, C2, C3, C4, C5 and intermediate levels) during different time periods. The corrosion standard has similar corrosion rate tabulation for different materials. We best fit the values in these tables to generate a function that gives weight loss per unit area for a particular material, given the atmospheric corrosivity, for a given elapsed time t. The following form fits all materials and conditions:

Material	k	n	m
Steel	13.84	2.9621	0.5257
Copper	0.97416	2.4492	0.67018
Zinc	0.7	2.7355	0.8169

Table 2: Constants for different materials for Equation 1

$$\Omega = k * C^n * t^m \tag{1}$$

where, Ω is the corroded weight per unit area, C is the corrosivity level and k, n, and m are constants whose values are given in Table 2 (with different values for different materials). Our stochastic corrosion model is now guided by Equation 1.

3.2 Model Building Blocks

We start with a voxelization of the corroding object as in [20]. As a result of this parametric voxelization, the voxels in the object space are warped cubes [2]. Voxels account for the geometric structure of a solid object. In this paper, we incorporate physical properties of the voxel, i.e., each voxel is assigned a mass based on its volume and material density.

We follow the framework of [20] as the core of our model, which we describe first. Each voxel has six face neighbors. A voxel with at least one face exposed is a boundary voxel. The simulation processes only the boundary voxels. Each voxel maintains a corrosion level (ψ), which accounts for the actual decay state of the voxel. It ranges from 0 to 1. At level 0, an exposed voxel is fresh material. At 1, the voxel is fully corroded and removed from the simulation, exposing voxels behind it. At each step of the simulation, the corrosion level of each exposed voxel is incremented by a constant factor δ based on a probability distribution function and the average level of corrosion around it.

We borrow the following intrinsic parameters of the simulation employed by [20].

- ν: It controls the influence of the immediate neighborhood. This aggravates the corrosion due to the flow of anions in the medium.
- ω: A constant value that accounts for material strength resistance to get corroded. When ψ at a point exceeds ω, the voxel is removed.
- δ: This is the factor by which ψ of a voxel is increased at a simulation step, it is selected for upgrade by the probability distribution function.

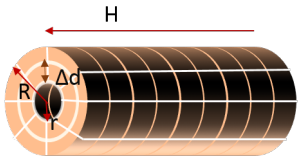


Figure 2: Basic cylindrical pipe

Fig. 1 describes our simulation and rendering pipeline. Corrosion is a generic phenomena, which broadly depends on the material properties of the object and the environmental conditions it is exposed to. The input parameters of the model are as follows:

- M : material type (steel, copper, zinc etc.)
- C : corrosivity level of the environment (1–5)
- t : elapsed time for which corrosion the process is to be simulated

We describe the simulation scheme using cylindrical pipes. Let us consider a pipe with interior radius r, exterior radius R and height H, as shown in Fig. 2. Our simulation is agnostic to the voxel size. Let the thickness of the exposed layer (voxel width) be Δd. ρ is the density of the metal and 'A' is the total area exposed ($2\Pi RH$ assuming the outer surface of the pipe is exposed). So the volume exposed is $A * \Delta d$. The weight of the exposed volume (α) is given by $A * \Delta d * \rho$. Equation 1 gives us the value of Ω, the expected weight loss (in g/m^2) in the given elapsed time. The expected weight loss in grams (ξ) is thus:

$$\xi = \Omega * A \tag{2}$$

The probability of a voxel to be removed can now be calculated as the ratio of the expected weight loss to the weight of the exposed volume. It is given by:

$$\upsilon = \frac{\xi}{\alpha} = \frac{\Omega * A}{A * \Delta d * \rho} = \frac{k * C^n * t^m}{\Delta d * \rho} \tag{3}$$

υ plays a vital role in the propagation of the corrosion phenomena. This is explained in detail in section 3.3. The material weight loss computation is agnostic to the area exposed and shape dependencies (Equation 3). The proposed model is independent of the the voxel resolution (as seen in Equation 3). The probability of removal of a voxel is also independent of the total exposed area.

3.3 Model Simulation

At every simulation step, the state of each voxel is primarily influenced by three main factors: the time of exposure, the the level of corrosion in its vicinity and the exposure of the voxel.

Corrosion rate has been measured to be an exponentially decreasing function of time [12]. We model the corrosion probability to decay with time according to Equation 4.

$$\chi = \lambda_0 * exp(-\lambda_1 * t) \tag{4}$$

with some constants λ_0 and λ_1. To achieve the expected Ω, we assign $\lambda_0 = \upsilon$ from Equation 3 to calculate χ for updating ψ. This controls the aggregate material loss in the simulation. Our simulation has fixed $\omega = 1$. The pace of corrosion is hence controlled by λ_1. This can increase or decrease the probability of material loss. [20] does not have any such control over the speed of the complete simulation.

For each exposed voxel, the average neighborhood (β) is the average of ψ values of its neighbors. Now, the corrosion simulation steps are listed below for each exposed voxel 'a':

1. Choose a random number r uniformly distributed between 0 and 1.
2. If $r < \chi$, $\psi(a)$ += δ
3. If $\beta(a) > \nu$, $\psi(a)$ += δ
4. If $\psi(a) > \omega$, remove the voxel. Distribute $\psi(a)$ equally among the now exposed voxels in its neighborhood.
5. For all exposed voxels, repeat steps 1–3 until the total material loss reaches Ω for a given elapsed time t given in Equation 3 .

If a voxel has been exposed longer, the chances of its corrosion is high. Closeness to a highly corroded region also accelerates the corrosion process of a voxel.

4 RENDERING

Corroded objects exhibit a wide spectrum of appearances, which may be nearly impossible to replicate using conventional rendering techniques. The dynamic conversion of one state to another depending on the interaction between the environment and the material further complicates rendering. To best represent the actual state of the object, we propose a general rendering framework, which imparts the final color to the corroded object derived from actual high resolution photographs of real objects. Our simulation results in the set of surface voxels whose exposure is primarily governed by material-environment interaction over a certain period of time. We estimate their appearance by that of similarly corroded points in the input photograph. We start with a parameterization of the original surface to be simulated. This allows us to map the model voxels back to that parameterization at any stage of the simulation.

 ISBN 978-80-86943-65-7

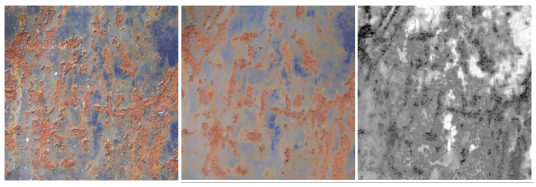


Figure 3: Real photograph sample (a), Albedo Map (b) and its corresponding weathering degree map (c)

Then, instead of re-surfacing the model voxels, we directly render the original surface with the removed voxels encoded in a displacement map, following [20]. The displacement map value is directly equal to the depth of the exposed surface from the original surface in the direction normal to that surface. The simulation also generates the current level of corrosion for every exposed voxel in a "weathering degree" map (W_2). We use this weathering degree map also, to compute a bump-map.

Finally, we compute the color by example of a corroded material of the same type. We first eliminate the illumination from the photograph [5], getting a map of albedo values for each pixel, I_1. Employing the user-in-the-loop technique of Wang et al. [24], we then construct a degree map of the photograph: the estimated corrosion level of the material at each pixel. Fig. 3 shows an example of a photograph, its albedo map and the final weathering degree map for the real photograph.

We then generate the final albedo map I_2 of the simulated object using these three maps:

1. Example weathering degree map W_1 constructed from the photograph
2. Simulated weathering degree map W_2 generated from our simulation
3. Example albedo map I_1 constructed from the photograph

Let I_1 and W_1 each be of size M $\times$ N and let I_2 and W_2 be of size P $\times$ Q.

To compute $I_2(x,y)$, one possibility is to individually locate the value (similar to) $W_2(x,y)$ in W_1 and assign the albedo from I_1 found at that pixel. This provides unnatural and random looking results. Further, it is possible to find $W_2(x,y)$ at multiple locations in W_1, with large variance in color values. Hence a coherent look-up provides a more natural look. However, instead of matching pixels based on their neighborhoods as in [24], we match entire tiles. Unfortunately, this generates discontinuity at tile boundaries. Hence, we employ overlapping tiles and then combine color from multiple tiles at each pixel. Further, we do not require a seven dimensional appearance manifold to generate realistic color. 7D input is cumbersome to acquire. We instead use

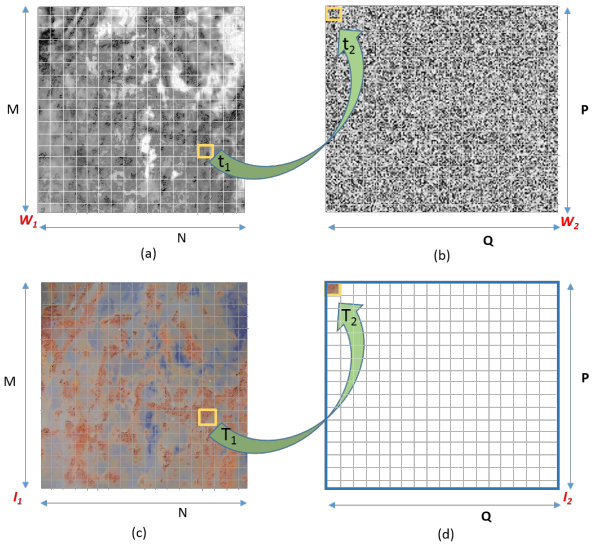


Figure 4: Procedure to generate the final diffuse color map based on least chi-squared distance between histogram distribution of degree values. Each tile (highlighted as yellow) is m*m sized.

standard RGB photographs and obtain improvements over the technique in [24] as demonstrated later. We describe our algorithm next.

We consider $m \times m$ tiles in each map. As I_1 and W_1 are of the same size, a tile T_1 at (x,y) location in I_1 corresponds to the same (x,y) location in W_1. (See Fig. 4(a) and (c), t_1 and T_1, respectively.) We hence use the same symbol to denote corresponding tiles in the two maps. Similarly, we use a common symbol for tiles in W_2 and I_2 (See Fig. 4(b) and (d), t_2 and T_2, respectively).

As mentioned, we employ a sliding window approach. We start with an $m \times m$ tile in one corner. The next tile is offset by one pixel in one of the dimensions, and so the window slides. The entire map is overlaid with such overlapping tiles (see Figure 4). For each tile in W_2, we compute the best matching tile in W_1 using its *histogram distribution of the degree values* as the 'feature vector' for matching.

- Calculate histogram of each tile in W_1 and W_2 using their degree values.
- For each tile T_2 in W_2, find the best matching tile T_1 in W_1, based on least chi-squared distance [18] between their respective histograms.
- Impart color from the matching tile T_1 in I_1 to T_2 in I_2.
- Every pixel $p \in I_2$ lies in multiple (sliding) tiles (see tiles marked in blue and green in Fig. 5) and has corresponding color contributions.
- The final color for pixel p is generated by the weighted average of these color contributions.

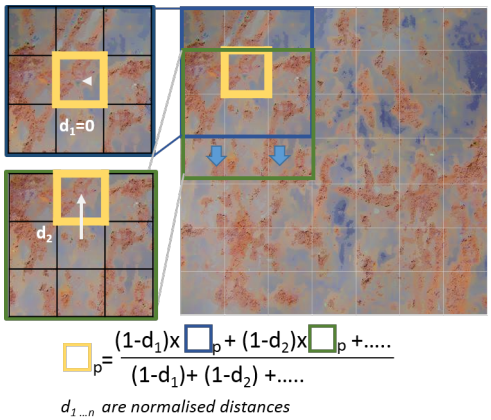


Figure 5: Sliding window procedure to impart color to a pixel p in I_2. The distribution of colors is dependent on the spatial distance of pixel p in I_2 from the center of the tile (T_2 in I_2). Yellow highlighted part is the pixel for which color contribution calculation is shown. Blue and green are the tiles pixel p is part of. For the blue tile p lies at the center, but for green tile p is at distance d_2 from the center. The color contribution from the blue tile at distance d_1 would be more than the RGB value from green tile at d_2.

The color maps generated using the RGB values employing the pixel matching technique of [24] produces blockiness. As can be observed in Fig. 6, our results produce sharper results. Moreover, our colors have a higher fidelity to the computed weather map in W_2.

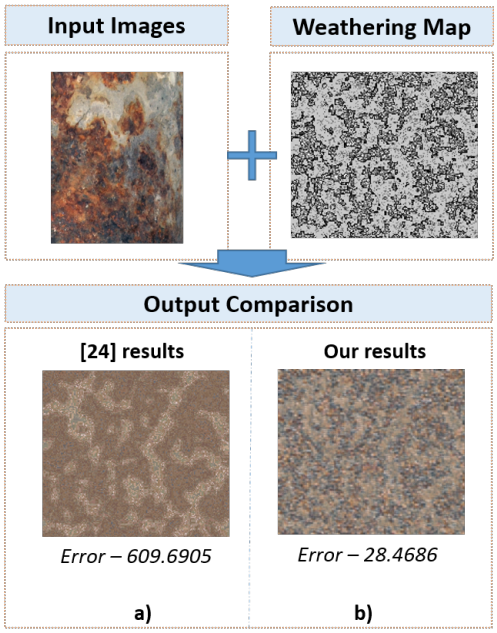


Figure 6: Comparison between our diffuse map generation and texture generated using [24], our results have much less blockiness and much better spread of colors. Also L^2 norm error is much less.

We perform an error analysis of results from our color maps and those generated by [24]. This is done by calculating how accurately a color has been assigned for a particular corrosion level. For each pixel in I_2, we compute the difference in color from the "nearest supporting color" in I_1. We then compare the L^2 norm error for both the set of outputs. To compute the nearest supporting color for $I_2(x,y)$, we locate all points on the appearance manifold [24] on I_1 and W_1, which have a corrosion level of $W_2(x,y)$. Among these, $I_1(x,y)$ that is closest (in L^2 sense) to $I_2(x,y)$ is considered the nearest supporting color. Intuitively, the corroboration of a similar color with the same corrosion level in the input maps testifies to it being the right choice.

Also, we generate a diffuse texture map, a normal map and a displacement map, which incorporate the overall deterioration in the three dimensional object, which is difficult in their proposed strategy.

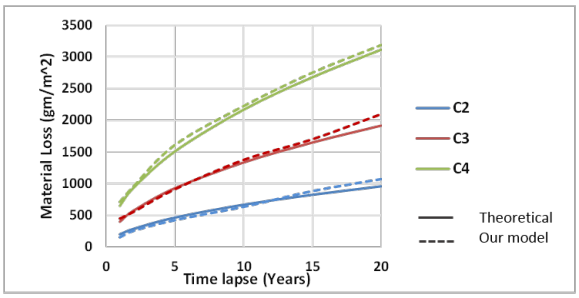
Finally, after generating the diffuse texture map, the normal map and the displacement map, we produce the final rendering.

5 RESULTS AND VALIDATION

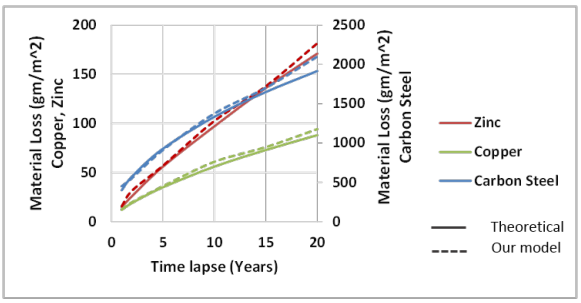
Our model results closely follow the weight loss paradigm of materials including Iron, Copper and Zinc under environmental conditions ranging from gentle to highly hostile. We validate our results exhibiting a comparison of the weight loss from our simulation to the predicted weight from the ISO standard. Our major focus is on realistic rendering of corroded objects and less on the performance statistics. Our model simulation takes 2-4 minutes (for Figs. 8, 9 and 10). Texture generation is a two step process. The first step creates the appearance manifold (which is computed just once) and the second step generates the final color, which takes 20–30 seconds for for a texture size of 256×256.

Our model simulations are agnostic to voxel size. The surface layer of depth is specified by the user. The size of voxels, in real units, is also specified by the user. For the results shown in Figs. 8, 9 and 10, the thickness of the pipe is 0.5cm. This depth is divided into sixteen layers and the average voxel size is 0.00491 cm^3. In Fig. 7(a), we compare time variant simulated results for carbon steel under various environmental conditions (C1 - C5) with predicted weight loss in grams per unit area from the ISO standard. As is apparent from the graph, our model complies with the ISO standard from various atmospheric conditions at different elapsed times. The graph clearly depicts that the total weight loss closely matches the predicted value. Being agnostic to the type of material, our framework is able predict the state of other materials like Copper and Zinc also once the respective material property is provided as input. We have validated our model with the actual weight loss in materials such as Steel, Copper and Zinc. In Fig. 7(a), we compare our results for various materials including steel, copper and zinc for cor-

 ISBN 978-80-86943-65-7



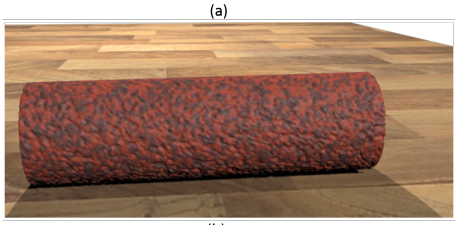
(a)



(b)

Figure 7: The two charts compare weight loss in grams per unit area from our simulation at different elapsed times with predictive weight losses from [15]. The first chart compares the weight losses for carbon steel at different corrosivity levels. (b) shows the weight losses for different metals at corrosivity level C3. The compliance of our model with standard results is evident.

(a)



(b)

Figure 8: Visual comparison of material loss in steel (a) and copper (b) pipe after one year

rosivity level C3 and observe that the predicted weight losses are achieved by our simulation effectively.

Our results are validated both with respect to the expected structural changes as well as color similarity to input. For example, Fig. 8 showcases the rendered results within the weight loss band in steel and copper pipe for the first year as suggested by ISO [15]. For steel pipe, the weight loss is much more than for copper after one year. Our model generates a time variant corrosion simulation. Fig. 9 exhibits the state of a copper pipe at different time steps impacted by an environment of corrosivity C3. The color and surface changes for the copper pipe shows the deterioration of the pipe at multiple time steps. Fig. 10 shows the impact of different corrosivity levels on steel pipe. C5 is much more aggressive and hostile environment and the surface deterioration of the pipe clearly depicts that.

6 CONCLUSION

We propose a physio-chemically based stochastic model to predict shape degradation of objects undergoing corrosion. Our proposed approach is generic across different materials and handles different object sizes and shape intricacies. We allow for material type and environmental conditions to be given as inputs making the framework simple yet comprehensive. In order to generate rendered output we propose a holistic rendering approach involving mapping of surface with color seen in actual corroded objects. We have presented a model that estimates structural damage due to material loss for uniform corrosion of various materials. Extending the model to other forms of corrosion including pitting and crevice corrosion would be an important avenue for further research. Our current rendering scheme focuses primarily on surfaces of revolution. We aim to generate simulation for general objects in future. Also, further work can be done to understand the physical and optical properties of corrosion residue.

7 REFERENCES

[1] Bandeira, D., Walter, M.: Synthesis and transfer of time-variant material appearance on images. In: Computer Graphics and Image Processing (SIBGRAPI), 2009 XXII Brazilian Symposium on, pp. 32–39. IEEE (2009)

[2] Chen, Y., Cohen, J., Kumar, S.: Visualization of time-varying curvilinear grids using a 3D warp texture. In: Vision, Modeling, and Visualization (2005)

[3] Chen, Y., Xia, L., tsin Wong, T., Tong, X., Bao, H., Guo, B., yeung Shum, H.: Visual simulation of weathering by γ-ton tracing. In: ACM SIGGRAPH, pp. 1127–1133 (2005)

[4] Clément, O., Benoit, J., Paquette, E.: Efficient editing of aged object textures. In: Proceedings of the 5th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa, pp. 151–158. ACM (2007)

[5] Dong, Y., Tong, X., Pellacini, F., Guo, B.: AppGen: Interactive material modeling from a single image. ACM Trans. Graph. **30**(6), 146:1–146:10 (2011). DOI 10.1145/2070781.2024180

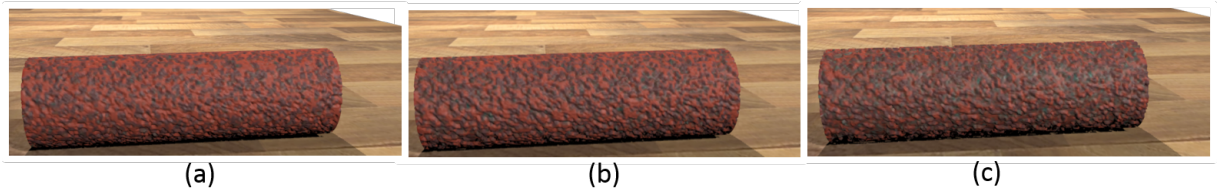


Figure 9: Depiction of time variant decay of a copper pipe at 1 (a), 2 (b) and 5 (c) years respectively. The surface degradation of the copper pipe in the 5^{th} year compared to the 1^{st} year is shown.

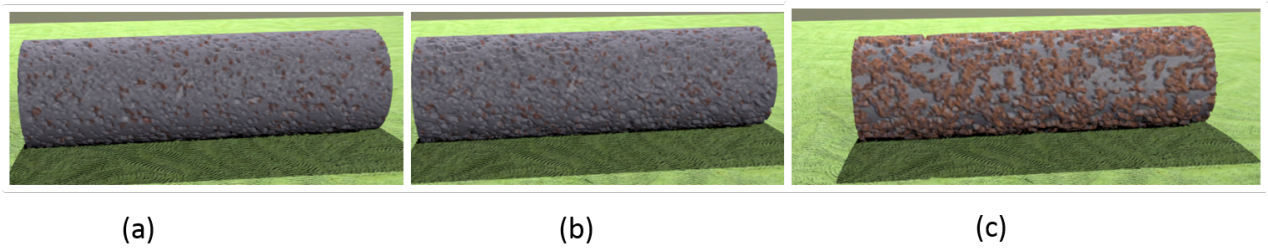


Figure 10: Corroded steel pipe after one year under C2 (a), C3 (b) and C5 (c) corrosivity levels, respectively. The pipe decays at a higher pace under C5 atmospheric conditions.

[6] Dorsey, J., Edelman, A., Jensen, H.W., Legakis, J., Pedersen, H.K.: Modeling and rendering of weathered stone. In: ACM SIGGRAPH 2006 Courses, p. 4. ACM (2006)

[7] Dorsey, J., Hanrahan, P.: Modeling and rendering of metallic patinas. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp. 387–396. ACM (1996)

[8] Dorsey, J., Pedersen, H.K., Hanrahan, P.: Flow and changes in appearance. In: ACM SIGGRAPH 2005 Courses, p. 3. ACM (2005)

[9] El Aal, E.A., El Wanees, S.A., Diab, A., El Haleem, S.A.: Environmental factors affecting the corrosion behavior of reinforcing steel III. measurement of pitting corrosion currents of steel in Ca $(OH)_2$ solutions under natural corrosion conditions. Corrosion Science **51**(8), 1611–1618 (2009)

[10] Genel, K., Demirkol, M., Gülmez, T.: Corrosion fatigue behaviour of ion nitrided aisi 4140 steel. Materials Science and Engineering: A **288**(1), 91–100 (2000)

[11] Guessasma, S., Elkedim, O., Nardin, P., Hamzaoui, R., Grosdidier, T.: Monte carlo simulation of uniform corrosion process under potentiostatic conditions. Corrosion science **49**(7), 2880–2904 (2007)

[12] Henshall, G., Halsey, W., Clarke, W., McCright, R.: Modeling pitting corrosion damage of high-level radioactive-waste containers, with emphasis on the stochastic approach. Tech. rep., Lawrence Livermore National Lab., CA (United States) (1993)

[13] Hwang, G., Yoon, S.H., Park, S.: Video-based weathering gallery. Multimedia Tools and Applications pp. 1–17 (2015)

[14] Corrosion of metals and alloys – corrosivity of atmospheres – classification, determination and estimation (2012)

[15] Corrosion of metals and alloys – corrosivity of atmospheres – guiding values for the corrosivity categories (2012)

[16] Kadry, S.: Corrosion analysis of stainless steel. Eur. J. Sci. Res **22**(4), 508–516 (2008)

[17] Kamata, Y., Manabe, Y., Yata, N.: Simulation of aging metal with preservative coating. In: The 2013 International Conference on Computer Graphics, Visualization, Computer Vision, and Game Technology. Atlantis Press (2013)

[18] McCune, B., Grace, L., Urban, D.: Analysis of Ecological Communities. MjM Software Design, Gleneden Beach, OR (2002)

[19] Mérillou, S., Dischler, J.M., Ghazanfarpour, D.: Corrosion: simulating and rendering. In: Graphics Interface, vol. 2001, pp. 167–174 (2001)

[20] Nisha, J., Prem, K., Kumar, S.: Simulation and rendering of pitting corrosion. In: Computer Vision, Graphics & Image Processing, 2014. ICVGIP'14 (2014)

[21] Rivas, D., Caleyo, F., Valor, A., Hallen, J.: Extreme value analysis applied to pitting corrosion experiments in low carbon steel: Comparison of block maxima and peak over threshold approaches. Corrosion Science **50**(11), 3193–3204 (2008)

[22] Scarf, P.A., Laycock, P.J.: Estimation of extremes in corrosion engineering. Journal of applied statistics **23**(6), 621–644 (1996)

[23] Shreir, L.L., et al.: Corrosion. Vol. I. Metal/environment reactions. Ed. 2. Butterworth & Co.(Publishers) Ltd. (1976)

[24] Wang, J., Tong, X., Lin, S., Pan, M., Wang, C., Bao, H., Guo, B., Shum, H.Y.: Appearance manifolds for modeling time-variant appearance of materials. ACM Transactions on Graphics (TOG) **25**(3), 754–761 (2006)

[25] Wojtan, C., Carlson, M., Mucha, P.J., Turk, G.: Animating corrosion and erosion. In: NPH, pp. 15–22. Citeseer (2007)

[26] Xuey, S., Wang, J., Tong, X., Dai, Q., Guo, B.: Image-based material weathering. Computer Graphics Forum **27**(2), 617–626 (2008)

An Efficient Reduction of IMU Drift for Registration Error Free Augmented Reality Maintenance Application

Lakshmiprabha N. S.[1,2] ns.lakshmiprabha@cern.ch
Alexander Santos[1] a.alvsantos@cern.ch
Olga Beltramello[1] olga.beltramello@cern.ch
[1]European Organization for Nuclear Research, CERN, Switzerland
[2]University of Rome Tor Vergata, Italy

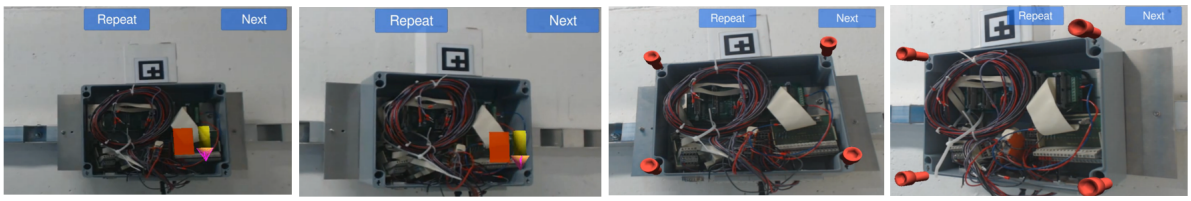


Figure 1: Augmented reality application showing user perceptive view of two different maintenance procedures.

ABSTRACT

Augmented reality (AR) is a technology that overlays virtual 3D content in the real world to enhance a user's perception. This AR virtual content must be registered properly with less jitter, drift or lag to create a more immersive feeling for the user. The object pose can be determined using different pose estimation techniques using the data from sensors cameras and inertial measurement units (IMUs). Camera based vision algorithms detect the features in a given environment to calculate the relative pose of an object with respect to the camera. However, these algorithms often take a longer time to calculate the pose and can only operate at lower rates. On the other hand, an IMU can provide fast data rates from which an absolute pose can be determined with fewer calculations. This pose is usually subjected to drift which leads to registration errors. The IMU drift can be substantially reduced by fusing periodic pose updates from a vision algorithm. This work investigates various factors that affect the rendering registration error and to find the trade-off between the vision algorithm pose update rate and the IMU drift to efficiently reduce this registration error. The experimental evaluation details the impact of IMU drift with different vision algorithm pose update rates. The results show that the careful selection of vision algorithm pose updates not only reduces IMU drift but also reduces the registration error. Furthermore, this reduces the computation required for processing the vision algorithm.

Keywords

Augmented reality; Pose estimation; Inertial measurement unit; Marker tracking; Sensor fusion; Registration error

1 INTRODUCTION

In past decades, virtual reality (VR) and augmented reality (AR) have seen a wide range of applications starting from entertainment (games) to medical fields. Many companies like Google (Google glass[1]), Microsoft (Hololens[2]), Epson (Moverio[3]) for example brought this AR technology from research experiment projects to daily use commercial products. The AR technology enables one to perceive reality in a more informative dimension. This supplementary information can be useful in many applications, one such is a maintenance application [Feiner11]. This paper concentrates on maintenance operation in ATLAS[4] particle physics detector (in Large Hadron Collider (LHC)) as a use case environment. The operators and technicians in this high energy physics radiation environment have to finish the maintenance job rapidly to reduce the exposure time. The AR technology can help in replacing paper manuals and actually display each maintenance procedure in the operator's field of view. Figure 1 shows the user perceptive view of

[1] https://support.google.com/glass/
[2] https://www.microsoft.com/microsoft-hololens/
[3] http://www.epson.com/cgi-bin/Store/jsp/Landing/moverio-bt-200-smart-glasses.do
[4] http://atlas.ch/

 ISBN 978-80-86943-65-7

two maintenance procedures for repairing a FPIAA (Finding people in ATLAS environment) system.

AR technology relays on the pose information provided by the sensors to overlay the virtual content. Amongst many available sensors, camera and Inertial measurement units (IMUs) are complete sensors, as they provide position and orientation in both indoor and outdoor applications. Also, these sensors can be compared closely with the humans vision (camera) and vestibular (IMU) system [Corke07]. The vision and vestibular systems provide key information about spatial orientation, body posture, equilibrium, reflex behaviours such as eye movement coordination, and navigation. This vestibular system that does inertial sensing is protected in the inner ear. The proper fusion of the camera and the IMU data can help us to mimic the human way of perceiving the environment. In this work, pose estimation is carried out by fusing camera and IMU data.

Camera based vision algorithms are very well suited for the AR application. However, based on the available processing power, the computation time for a vision based algorithm (other than fiducial marker tracking) can vary from several milliseconds to seconds. It is well known that the IMU provides faster pose updates than camera image based pose estimation. The next obvious measure to know about is the accuracy in terms of rendering an AR virtual content. The major disadvantage of IMU is that they typically suffer from drift. One well known method to overcome this drift is to use other low drift data (for example pose computed from the vision images) and then fuse the two data together. The fusion of vision relative pose and IMU absolute pose data will help to predict the object location assuming that the camera and IMU are very well calibrated. There have been different methods proposed in AR community for camera and IMU related sensor fusion [Azuma94, Chai99, Davison03, Hol06]. Nevertheless, one of the aspects that significantly affects the registration error for AR applications is the pose update rate from vision algorithm that will efficiently minimize the IMU drift. The focus of this paper is to define the different parameters that affect the rendering registration error and also to determine the pose update rate from the vision algorithm required to reduce the IMU drift and the registration error for an AR application. The experimental results show the impact of pose updates from the vision algorithm on IMU drift and registration error.

2 RELATED WORK

The usage of camera and IMU sensors is very well known in augmented reality applications for pose estimations. Starting from [Azuma94], there have been different work that discussed about the fusion of the pose from the camera and the IMU to reduce the registration error [Chai99, Davison03, Hol06]. IMU data has also been used in boosting vision feature matching and these features are used as an individual measurement as opposed to the more traditional approaches where camera pose estimates are first extracted by means of feature tracking and then used as measurement updates in a filter framework as in [Oskiper12]. Recently, Kriti et al [Kumar14] highlighted the usage of IMU data for the occlusion problem in vision algorithm by defining the fusion techniques. Another interesting application of augmented reality is binoculars [Oskiper13] using stereo camera, GPS and IMU for pose estimation. Their results show that the vision tracking algorithm computation takes 30 milliseconds which is more acceptable even without the need of an IMU. One different application of an IMU is used in sensing the user movement to correct the registration errors [Lo10].

In this work, the application of our interest is a maintenance operation in a complex and an extreme environment. A most recent work on a similar application is [Zhu14], which presented a whole system from tracking to rendering with the delay measurements at different modules. In this work IMU data was fused with vision algorithm pose using Extended Kalman Filter but there was limited highlight on the vision pose updates used for fusion and its effect on the registration error. The goal of our work is to find a trade-off between vision algorithm pose updates and the IMU drift to efficiently reduce the registration error. This will help to use only the required number of pose updates from the vision algorithm to have a good visualization of AR content at the correct location. Hence this optimizes the use of processing power that is required for vision algorithm computation.

In this paper, the different parameters affecting registration of AR visualization is discussed in section 3. In the subsequent sections, pose estimation using a camera and an IMU is elaborated. In section 6, the sensor fusion using extended kalman filter is explained. The experimental evaluation of determining the correct pose update rate from a vision algorithm required to reduce the drift and further the registration error is illustrated in section 7. The findings from this work and future direction is summarized in conclusion.

3 AR VIRTUAL CONTENT REGISTRATION

Augmented reality (AR) is user centric, where the user evaluates the system based on what he/she sees in the display device (either hand-held or head mounted display). Most common problem in rendering the AR virtual content is the registration between a real and a virtual object in the scene. The misalignment of the AR virtual content with their desired real world object is referred to as registration error. Figure 2 shows the registration error between real screws (dark gray) and the

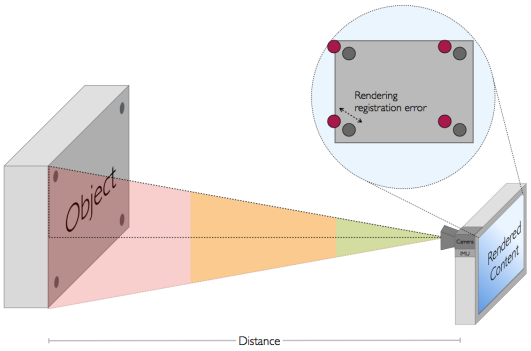


Figure 2: Augmented reality video display showing registration error in rendering the virtual object (red) on real object (dark gray) is shown.

virtual screws (red). This registration error must be kept quite minimal even during fast movements. Otherwise the system will not provide any help to the user. Assuming that the camera, IMU and the display device are calibrated to perfection, the registration error can be expressed as,

$$E = f(P_{cam}, D_{IMU}, D, S) \quad (1)$$

where, P_{cam} is the accuracy of vision algorithm pose estimation, D_{IMU} is the drift developed in the IMU over time, D is the distance between the system (camera+IMU) and the target object (refer Figure 2) and S is the size of the object of interest.

If the vision algorithm detects object A as object B (for example, blue cable detected as red in Figure 1) from the captured images, this will result in highlighting a different object. If the pose update from the vision algorithm is at every 1s, then between t_0 to $t_0 + 1s$ the drift accumulated in the IMU (D_{IMU}) can influence the registration error. If the distance (D) is increasing the error is also increasing, this can result in pointing A object (say red cable) as B (say blue cable) to the user. If the object of interest and the AR content is large (virtual box lid over the FPIAA box in Figure 1), a small registration error can be acceptable by the user. In other case, if the object of interest and virtual content is small, say highlighting a cable where there are identical cables next to each other, a small registration error can point a different cable to the user.

The main goal of this paper is to study the IMU drift related registration error developed over time and efficiently reduce that error using vision pose update. In order to have an accurate pose measurement from the vision algorithm, this work uses marker tracking instead of real object pose estimation. This will also help us to use marker pose as the ground truth and evaluate the error from other parameters (i.e. from the IMU drift).

As a foundation to the above discussed point, it is required to have pose estimation from a camera, IMU and sensor fusion to carry out the experiments. In the next sections, pose estimation using a camera and an IMU is discussed.

4 POSE ESTIMATION USING CAMERA IMAGES

In order to have accurate pose estimation from the vision algorithm, a prepared environment with fiducial markers are used. There are several libraries mainly used to resolve the marker tracking issues in the given scene. Most well known marker tracking libraries are ARToolKit [Kato99], osgART [Looser06], DWART [Bauer01], ARTag [Fiala04], Ubitrack [Ubitrack04] etc. In this work, marker tracking from Ubitrack[5] library is used. The precise marker tracking is possible when it is provided with the camera parameters (intrinsic and extrinsic) calculated using camera calibration[6] [Salvi02].

5 POSE ESTIMATION USING IMU

An inertial measurement unit has three accelerometers, three gyroscopes to measure acceleration and angular velocity along X, Y and Z axis. Some of the IMU also include three magnetometers to calibrate against earth's magnetic field. The cost and size of inertial sensors increase with the accuracy and reduced drift range. Following subsections detail the calculation of orientation and position from angular velocity and acceleration data from IMU.

5.1 Orientation calculation

The orientation in terms of quaternions was calculated using explicit complimentary filter (ECF) discussed by M. Euston et al in [Mahony08]. Please refer [Mahony08] for more details on orientation calculation. In the next section, position calculation from accelerometer data is explained.

5.2 Position calculation

The process of integrating acceleration data twice to calculate position is not so direct, since the accelerometer data contains the body acceleration and gravity component. As a first step, the gravity component was removed and the resulting linear acceleration was used for further processing. The absolute position is then calculated by double integrating this linear acceleration. The relative 6Dof pose from camera and absolute 6Dof from IMU is fused using the Extended Kalman Filter detailed in sensor fusion section.

[5] http://campar.in.tum.de/UbiTrack/WebHome
[6] http://www.vision.caltech.edu/bouguetj/calib_doc/

6 SENSOR FUSION

The fusion of pose data is performed with an Extended Kalman filter (EKF) [Bishop01] approach, a technique widely used in state estimation problems such as pose estimation in robotics, aviation and augmented reality [Azuma94, Chai99, Davison03, Hol06]. The kalman filter can be either used at the output of a pre-built sensor [Azuma94, Chai99] or as an integral part of the vision algorithm for pose estimation [Davison03]. The difference is based on the orientation representation, as in [Azuma94, Davison03] it is quaternion and [Chai99] uses euler representation. Most importantly the filter should cope up with unsynchronized pose data coming from the different sources particularly IMU and camera in our work. In this paper, marker tracking is used as vision algorithm since it provides accurate pose estimation which can serve as ground truth to measure the registration error developed by an IMU drift. Further for other applications this marker tracking can be replaced by any real object pose estimation algorithms within the same sensor fusion frame work detailed below.

6.1 Extended Kalman Filter (EKF)

A nonlinear version of the Kalman filter that linearizes an estimate of the current mean and covariance is referred as an Extended Kalman filter (EKF). This filtering technique allows us to estimate the parameters from multiple measurements without completely discarding information from previous sensor readings.

Let us assume that the process has a state vector $x \in R^n$, and is governed by the non-linear stochastic difference equation

$$x_k = f(x_{k-1}, u_k, w_{k-1}) \tag{2}$$

with a measurement $z \in R^m$ that is

$$z_k = h(x_k, v_k) \tag{3}$$

where the random variables w_k and v_k represent the process and measurement noise. In this case the non-linear function f in the difference equation 2 relates the state at the previous time step $k-1$ to the state at the current time step k. It includes driving function u_k and the zero-mean process noise w_k as parameters. The non-linear function h in the measurement equation 3 relates the state to the measurement z_k. In this case, the state vector consists of pose from camera and pose from IMU transformed to the camera coordinate frame using the camera-IMU calibration.

To estimate a process with non-linear difference and measurement relationships, we begin by writing new governing equations that linearise an estimate about equation 2 and equation 3

$$x_k \approx \tilde{x}_k + A(x_{k-1} - \hat{x}_{k-1}) + W w_{k-1} \tag{4}$$

$$z_k \approx \tilde{z}_k + H(x_k - \tilde{x}_k) + V v_k \tag{5}$$

where $\hat{x}_k$ is a posteriori estimate of the state at step k. A is the Jacobian matrix of partial derivatives of f with respect to x, that is

$$A_{[i,j]} = \frac{df_{[i]}}{dx_{[j]}}(\hat{x}_{k-1}, u_k, 0) \tag{6}$$

W is the Jacobian matrix of partial derivatives of f with respect to w,

$$W_{[i,j]} = \frac{df_{[i]}}{dw_{[j]}}(\hat{x}_{k-1}, u_k, 0) \tag{7}$$

H is the Jacobian matrix of partial derivatives of h with respect to x,

$$H_{[i,j]} = \frac{dh_{[i]}}{dx_{[j]}}(\tilde{x}_k, 0) \tag{8}$$

V is the Jacobian matrix of partial derivatives of h with respect to v,

$$V_{[i,j]} = \frac{dh_{[i]}}{dv_{[j]}}(\tilde{x}_k, 0) \tag{9}$$

The complete set of EKF equations is given in equations below,

EKF Time update

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_k, 0) \tag{10}$$

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T \tag{11}$$

the time update equations project the state and covariance estimates from the previous time step $k-1$ to the current time step k. A_k and W_k are the process Jacobians at step k (see equation 6 and 7) and Q_k is the process noise covariance equation at step k. Careful selection of process noise is a must to have good performance. The high value causes old measurements to decay quickly and new measurements are given a higher weighting.

EKF measurement update

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1} \tag{12}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0)) \tag{13}$$

$$P_k = (1 - K_k H_k) P_k^- \tag{14}$$

The measurement update equations (12 to 14) corrects the state and covariance estimates with the measurement z_k. H_k and V_k are the measurement Jacobians at step k (see equation 8 and 9). R_k is the measurement noise covariance at step k.

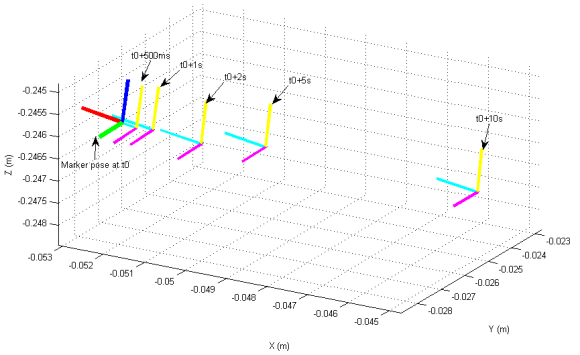


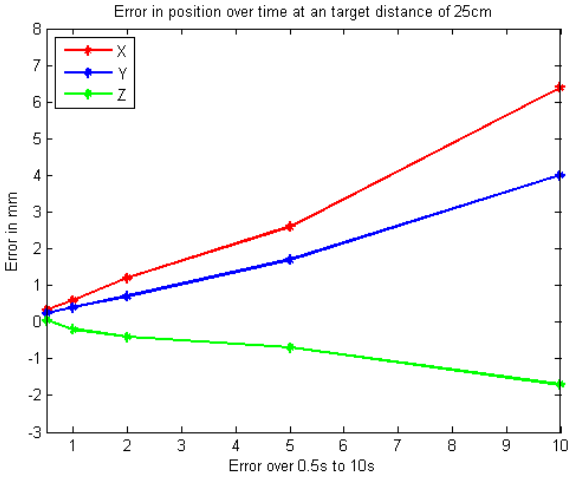
Figure 3: Error calculation over time at a target distance of 25cm. RGB is the marker pose (ground truth) and CMY is the sensor fusion pose after 500ms, 1s, 2s, 5s and 10s.

7 EXPERIMENTAL EVALUATION

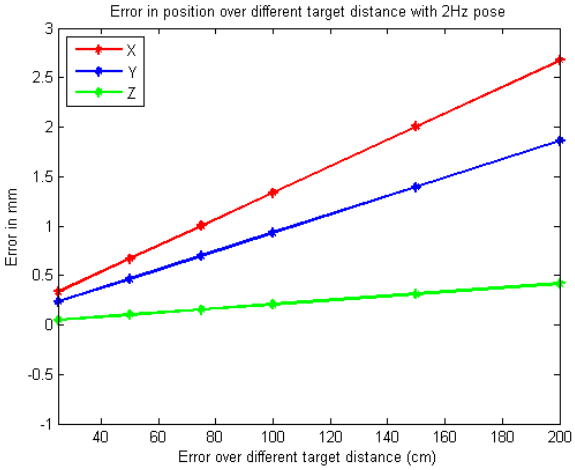
The framework is implemented in C and C++ and executed in Advantech MIO-5271 with Intel corei5-4300U CPU and 8GB of RAM. The sensors consist of a Logitech C920 camera and an Xsens MTi-100 IMU. Camera images are captured at 640x480 pixels resolution at 30fps and the IMU data at 100Hz frequency. The AR virtual content rendering is implemented using Unity3D[7] game engine.

This work concentrates on AR for a maintenance application as shown in Figure 1, the operating distance between system and target object can vary from 0.25 meter to 1.5 meter. The size of the object over which the AR content is rendered vary from 2cm to 8cm for the considered maintenance use case. The IMU drift related registration error was studied by varying one parameter and keeping all other constant. For example, the pose update rate from the vision algorithm to sensor fusion was varied to see the registration error produced due to the IMU drift under static conditions and at a fixed distance. The pose obtained from the marker remains as the ground truth pose that can be used to measure the drift from the IMU. The marker pose from camera images is taken at a fixed rate for fusing with the IMU data. The IMU pose through EKF is used to know the object location between the intermediate marker pose updates. The IMU drift developed in this duration is corrected using the upcoming marker pose from the camera. This drift over time was measured by comparing the marker pose with the sensor fusion pose. Since the camera and IMU are in a static condition (at 25cm away from the marker), the deviation between the marker pose and the sensor fusion pose is caused by the IMU drift. In the experiments, the marker pose is taken at 0.5s, 1s, 2s, 5s and 10s update rates. The drift between these updates are shown in the Figure 3 where the

[7] http://unity3d.com/



(a) Error in position over time at a target distance of 25cm.



(b) Error in position at different target distance with the 2Hz marker pose update.

Figure 4: Position error over time and at different target location.

marker pose is shown in RGB and sensor fusion pose is shown in CMY. The drift increases with the decrease in marker pose update rate, i.e. the drift is higher for every 5s update than the 2s marker pose update. As it can be seen in Figure 3, the deviation in orientation between marker pose (RGB) and the sensor fusion pose (CMY) is considerably lower than the position (XYZ) values. The closer look for the deviation in position (XYZ) values for different marker pose update rate is shown in figure 4 (a). This deviation in position is measured at a target marker placed 25cm away from the camera and the IMU. The registration error in terms of percentage along the three axis is given in Table 1.

 ISBN 978-80-86943-65-7

As mentioned in equation 1, the registration error is also related with the distance between the system and the target object. The influence of the different target distance and the related registration error is shown in Figure 4 (b). As the distance increases the registration error along all the three axis also increases steadily. These results are measured with the vision algorithm pose update of 0.5s to the sensor fusion since 2Hz pose update is possible in most real object pose estimation algorithms. Thus the careful selection of a vision algorithm is required to efficiently reduce the IMU drift and this in turn reduces the AR content registration errors. Further, this reduces the computation required for vision based algorithms to process several images per second to the number that is sufficiently enough to correct this registration error.

Marker pose update (s)	X(%)	Y(%)	Z(%)
0.5	0.64	0.82	0.02
1	1.23	1.54	0.067
2	2.4	2.69	0.147
5	5.9	6.99	0.29
10	12.2	14.33	0.686

Table 1: Registration error in percentage along X, Y, Z axis for different pose updates from marker.

The drift cannot be the same for different IMU. To test this statement, the above mentioned experiment was repeated again with another Xsens IMU and the measured registration error over time is shown in Figure 5. The result shows that the registration error developed over time due to IMU drift is purely random. The direction along which the error was happening is opposite in both the cases. The registration error was happening more in north east direction using the first IMU and it was moving in south west for the second case. Having said this, a method that reduces the registration error in the first case may not really serve as a solution for the other case. Thus, it is wiser to use other sources of information (vision algorithm in our work) to eliminate this error.

The maintenance application of our interest has a working distance of 0.5m to 1.5m. For this working distance, the IMU drift is 0.3mm to 2mm along the X axis with 0.5s marker pose updates and slightly less in other two axis. These measurements are taken under static conditions, but in real situation the system will be moved dynamically by the user. The size of the object on which the virtual rendering has to be shown is 2cm (screw in Figure 1). Considering these facts, we select the marker pose updates of 4Hz to be fused with the IMU data. The marker tracking pose at 4Hz frequency along with the sensor fusion pose is shown in Figure 6 . There is a slow and smooth transition of the sensor fusion pose between the two marker poses as shown in Figure 7 . The application of this pose data for a maintenance operation in ATLAS environment is shown in Figure 1. First two images show a cable operation and in next two images the box closure is highlighted by pointing the screw locations.

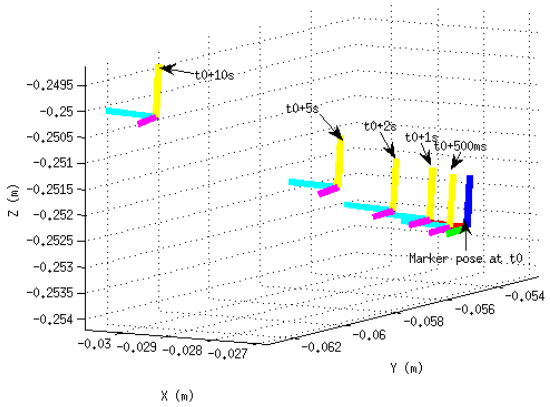
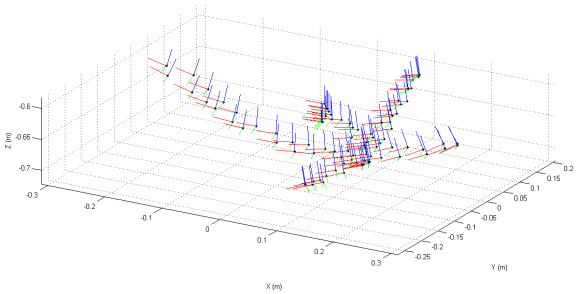


Figure 5: Error calculation over time at a target distance of 25cm. RGP is the marker pose (ground truth) and CMY is the sensor fusion pose after 500ms, 1s, 2s, 5s and 10s. Experiment performed with different IMU from Figure 3.

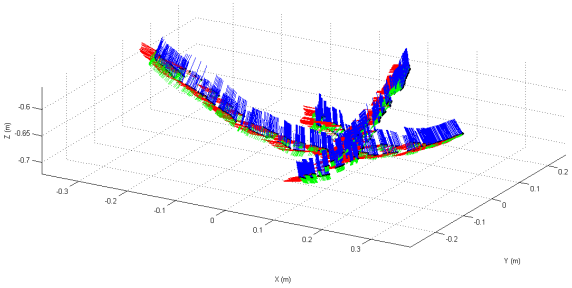
The camera and IMU together with the processor board mentioned above and a handheld video display showing the AR content was demonstrated to the users. They were asked to try with two similar setups, first one with only the marker tracking pose on all images (30Hz) and the second one with the marker pose at 4Hz and the IMU fused pose. The feedback was equal for both the setups with an advantage observed in fused pose setup during fast movements. This is because the motion blur in camera image due to the fast movements affected marker detection and the AR content was lagging behind or swimming around the corresponding real objects. Thus the efficient selection of marker pose updates rate for the fusion with the IMU data is performing better even in fast movements.

8 CONCLUSION

In this paper, the factors affecting the registration error in an augmented reality (AR) application was discussed. The real object pose estimation using camera and IMU was detailed with the formulation of sensor fusion using Extended Kalman filter (EKF). The advantage of having IMU data was to have fast pose updates. This data was rather affected by drift and needs to be corrected by the pose from a vision algorithm. Between this periodic correction of poses from a vision algorithm, the virtual object rendering was based on IMU data and the drift causes the registration error. This error in rendering can be defined as a function of pose accuracy from the vision algorithm, drift from IMU over



(c) Marker tracking pose at 4Hz.



(d) Sensor fusion with 4Hz marker pose and IMU data.

Figure 6: Marker tracking and sensor fusion results.

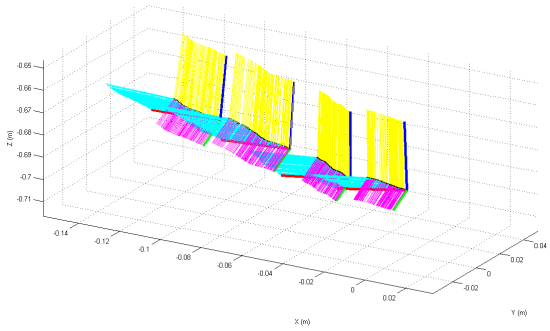


Figure 7: Close comparison of marker pose 4Hz (RGB) pose updates with the Sensor fusion pose (CMY).

time, size of the real object on which virtual content is overlaid and the distance between the target object and the system. Experimental evaluation detailed the effect of IMU drift on registration error over different pose update rates from the vision algorithm. The results also showed the effect of registration error over the distance and size factors. Based on the studies, a vision algorithm pose update rate that efficiently reduces the IMU drift was selected for the AR maintenance application. As a result, the registration error was minimized and secondarily that also optimized the processing power required for the vision algorithm computation.

The AR maintenance application was demonstrated to the end-user for their feedback with two setups, one with only marker tracking and the other using IMU fusion with fixed pose updates from the marker. Both the setup's performance was satisfactory during slow movements with the IMU fused marker pose setup showing a clear advantage during dynamic movements. In future work, the IMU drift during dynamic movements and its effect in registration error is in the pipeline to be studied by performing different experiments.

ACKNOWLEDGEMENTS

The authors wish to thank all other members of the EDUSAFE consortium. This research has been supported by a Marie Curie Initial Training Network Fellowship of the European Commission FP7 Programme under contract number PITN-GA-2012-316919-EDUSAFE.

9 REFERENCES

[Oskiper12] T. Oskiper, S. Samarasekera and R. Kumar, Multi-Sensor Navigation Algorithm Using Monocular Camera, IMU and GPS for Large Scale Augmented Reality, IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2012.

[Kumar14] K. Kumar et al, An Improved Tracking using IMU and vision fusion for mobile augmented reality application, The International Journal of Multimedia & Its Applications (IJMA) Vol.6, No.5, October 2014.

[Lo10] C.-C. A. Lo, T.-C. Lin, Y.-C. Wang, Y.-C. Tseng, L.-C. Ko, and L.-C. Kuo, Using intelligent mobile devices for indoor wireless location tracking, navigation, and mobile augmented reality, in IEEE VTS Asia Pacific Wireless Commun. Symposium (APWCS), 2010.

[Oskiper13] Oskiper, M. Sizintsev, V. Branzoi, S. Samarasekera, and R. Kumar, Augmented reality binoculars, IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2013.

[Zhu14] Z. Zhu et al, AR-Mentor: Augmented Reality Based Mentoring System, IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2014.

[Feiner11] S. Henderson and S. Feiner. Exploring the benefits of augmented reality documentation for maintenance and repair, IEEE Transactions on Visualization and Computer Graphics, 2011.

[Ubitrack04] Newman, Joseph and Wagner, Martin and et al. Ubiquitous tracking for augmented reality, IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp.192-201, 2004.

[Mahony08] Euston, Mark and Coote, Paul and et al. A complementary filter for attitude estimation of a fixed-wing UAV, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008.

[Hol06] Hol, J.D. and Schon, T.B. and et al. Sensor Fusion for Augmented Reality, 9th International Conference on Information Fusion, pp.1-6, 2006.

[Bishop01] Bishop, Gary and Welch, Greg. An introduction to the kalman filter, Proceeding of SIGGRAPH, Course 8, 2001.

[Azuma94] R. Azuma and G. Bishop, Improving Static and Dynamic Registration in an Optical See-through HMD, in Proceeding of Siggraph, Orlando, July 1994, pp. 194-204.

[Chai99] L. Chai, K. N Guyen, B. Hoff, and T. Vincent, An Adaptive Estimator for Registration in Augmented Reality, International Workshop on Augmented Reality (IWAR), October 1999.

[Davison03] A. Davison, W. Mayol, and D. Murray, Real-Time Localisation and Mapping with Wearable Active Vision, IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2003.

[Kato99] H. Kato and M. Billinghurst, Marker tracking and hmd calibration for a video-based augmented reality conferencing system, International Workshop on Augmented Reality (IWAR 99), pages 85-94, 1999.

[Looser06] J. Looser, R. Grasset, H. Seichter, and M. Billinghurst, OSGART - A pragmatic approach to MR, IEEE International Symposium of Mixed and Augmented Reality (ISMAR), 2006.

[Bauer01] M. Bauer, B. Bruegge, G. Klinker, A. MacWilliams, T. Reicher, S. Riss, C. Sandor, and M. Wagner, Design of a component-based augmented reality framework, IEEE International Symposium on Augmented Reality (ISAR), 2001.

[Fiala04] Fiala, M, ARTag Revision 1, A fiducial marker system using digital Techniques, NRC Technical Report (NRC 47419), National Research Council of Canada, 2004.

[Salvi02] J. Salvi and X. ArmanguÃ© and J. Batlle, A comparative review of camera calibrating methods with accuracy evaluation, Pattern Recognition, 35(7), pp. 1617-1635, 2002.

[Corke07] P. Corke and J. Lobo and J. Dias, "An Introduction to inertial and visual sensing", The International Journal of Robotics, pp. 519-535, 2007.

Adaptive Depth Bias for Soft Shadows

Alexander Ehm
Munich University of Applied Sciences
Lothstrasse 64
80335 Munich, Germany
ehm@hm.edu

Alexander Ederer
Munich University of Applied Sciences
Lothstrasse 64
80335 Munich, Germany
aederer@hm.edu

Andreas Klein
MBDA Deutschland GmbH
Hagenauer Forst 27
86529 Schrobenhausen, Germany
andreas-herbert.klein@mbda-systems.de

Alfred Nischwitz
Munich University of Applied Sciences
Lothstrasse 64
80335 Munich, Germany
nischwitz@cs.hm.edu

Abstract

With shadow mapping the need of a suitable biasing technique due to shadow aliasing is indisputable. Dou et al. [Dou14] introduced a biasing technique that computes the optimal bias adaptively for each fragment. In this paper, we propose enhancements for this algorithm. First, we extend the algorithm for soft shadows, such as percentage closer filtering (PCF) and percentage closer soft shadows (PCSS). Second, we minimize the projective aliasing by introducing a scale factor depending on the ratio between surface and light direction. We show that our enhancements increase the shadow quality and introduce only a small overhead.

Keywords

shadow mapping, bias, adaptive bias, projective aliasing, automatic bias adjustment

1 INTRODUCTION

The most common ways to achieve interactive shadows is shadow mapping. One of the major drawbacks of shadow mapping is surface acne, which is erroneous self-shadowing. This is commonly addressed by a depth bias. There are different biasing techniques and research has been (and is still) done in this area. Most of these biasing methods suffer from two problems: first, they require hand tweaking of different parameters for different scenes, for them to work well, and second, they do not use the minimal bias and therefore cause shadow detachment.

In [Dou14], Dou et al. proposed an approach, which adaptively computes the optimal bias for each fragment. But in the original paper, the adaptive bias is only used with hard shadow mapping, and to be of common interest today, the technique should be suitable for use with soft shadowing algorithms.

For that reason we introduce an enhancement to extend the "Adaptive Depth Bias for Shadow Maps" to the soft shadowing techniques PCF and PCSS. We start with an explanation of the first approach of the extension, which turned out to be slow, and then introduce the optimized extension, which only introduces reasonable overhead. Furthermore we found that the adaptive bias still suffers heavily from projective aliasing, and therefore introduce an enhancement of the original algorithm to be more robust against this kind of aliasing. We show the detailed modifications of the algorithm for all enhancements. The main contributions of this paper are:

- Extending the adaptive bias to the soft shadowing techniques PCF and PCSS;
- Enhancing the original algorithm to be more robust against projective aliasing.

2 STATE OF THE ART

A complete overview of biasing algorithms is out of scope for this paper. In [Eis11a] a nice assembly of shadowing and biasing algorithms can be found. Although biasing methods use different approaches to obtain the bias, they all have in common that they apply an offset, the bias, to the sampled depth values to remove false self-shadowing.

The OpenGL function glPolygonOffset, is (more general speaking) the combination of a constant bias and a slope-scaled bias. Constant bias means that the same offset is used for every fragment. A slope-scaled bias, in contrast, scales the bias up the higher the slope of the surface is compared to the shadow map plane.

With PCF filtering and large filter kernels, the assumption of a single depth value for comparison across the whole filter kernel does not suffice anymore. Isidoro [Isi06a] presents a method to use an individual depth value for each sample, by using a bias based on the depth changes in the receiver plane.

Dual depth layer bias as originally proposed in [Woo94a], uses the depth difference between the closest and the second closest surface as bias. To remove the surface-acne which was left at silhouette lines, this method was improved by [Wei03a] by the addition of a constant bound to this bias. This method introduces significant extra costs, since it needs an extra rendering pass.

3 ADAPTIVE BIAS AS BASIS

We will now shortly go through the Adaptive Bias introduced by Dou et al. [Dou14], which forms the basis to this paper. The "Adaptive Bias for Shadow Mapping" is yet another biasing technique, and functions as a drop-in replacement for other biasing techniques.

The basic idea behind the adaptive bias is to calculate the optimal bias for each fragment, since the minimal amount of bias needed to remove false shadows differs for each fragment. To achieve this, firstly the potential occluder that may cause false shadow, is computed. Afterwards, a small adaptive epsilon is added to shift the current, already biased, fragment just a little closer to the light source, just above its potential occluder. For an overview of the whole algorithm see [Dou14].

3.1 Optimal Depth Bias

Shadow mapping suffers from aliasing artifacts, such as false shadows, due to the discretization of a scene into a 2d texture of depth values. If the current fragment F_c lies in the region of this shadow map texel but not exactly where the depth value was sampled, its depth value may be bigger than the depth of fragment F_o stored in the shadow map, which will cause erroneous shadow on this fragment. F_o is the potential occluder of the current texel F_c. See Figure 1 for a basic illustration.

The optimal bias is the depth difference between the current fragment F_c, and the fragment sampled for the shadow map F_o, since this bias is the minimal bias to move the fragment F_c to its occluder F_o. In order to obtain the optimal bias, the potential occluder F_o is located under the assumption that the underlying geometry is a planar surface. This can be done by computing the intersection of the light ray $\overrightarrow{R}$, which is the ray traced from the light source through the center of the corresponding shadow map texel, and the plane P, which is the tangent plane defined by the current fragment F_c and its normal N.

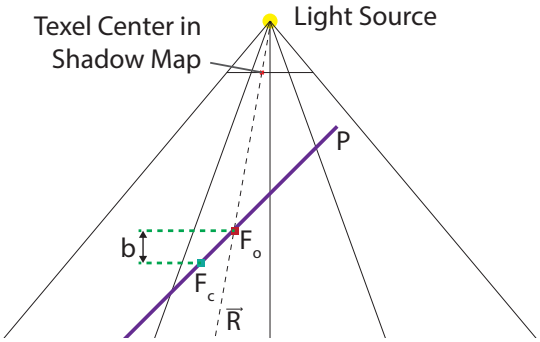


Figure 1: Illustration of the adaptive bias computation. F_c represents the current fragment, F_o is the potential occluder where this shadow map texel has been sampled. b is the optimal bias for this case.

3.2 Adaptive Epsilon

The optimal bias shifts the current fragment to its potential occluder. In order to shift it just above its potential occluder, which is desired to have a robust shadow test, a small epsilon value is needed. A constant epsilon value is not a good choice, since depth values are stored non-linearly in perspective projection.

Therefore, Dou et al. propose to use a constant epsilon but transform it based on the standard OpenGL depth compression function for perspective projections. The standard OpenGL depth compression function is the function that maps the depth values in between the near and the far clipping plane non-linear into the interval $[0,1]$.
For the constant component they recommend a value computed from *sceneScale*, which is defined as the length of the scene's bounding box diagonal and an empiric constant K. Thus the formula proposed by Dou et al. to obtain the adaptive epsilon is:

$$\varepsilon = \frac{(lf - depth \times (lf - ln))^2}{lf \times ln \times (lf - ln)} \times sceneScale \times K \quad (1)$$

With ln being the distance to the light's near plane, lf the distance to the light's far plane and depth the normalized depth value of the current fragment.

Dou et al. state, that they used $K = 0.0001$ for all their experiments. We could not prove that $K = 0.0001$ would be a good choice for different scenes. As a matter of fact, we had to modify the value of K for each scene, in order to get good results. This problem can be seen in Figure 11.

3.3 Performance

Dou et al. explain that their method is not very much slower, to be precise, around 20 %, than a constant bias. It is therefore much faster than the dual depth layer method, which they used as reference method for quality comparison, but can achieve results which are

of equal quality as those from dual depth layers. See [Dou14] for seeing the original quality comparisons. In our experiments we could prove that the adaptive bias performs well, it was 12 % slower than biasing with the glPolygonOffset function.

4 ADAPTIVE BIAS FOR SOFT SHADOWS

To be of use for today, a biasing technique needs to work with soft shadows. We will now firstly go through the combination with PCF [Ree87a] and then go through the combination of the adaptive bias with PCSS [Fer05a].

4.1 PCF

To obtain good results, it is not sufficient for PCF shadowing to use the adaptive bias of the current fragment for all shadow map texels in the filter kernel. This would lead to erroneous self-shadowing for large filter kernels. It would be best if we could compute the optimal bias for every texel in the PFC filter kernel. This is possible with some modifications to the original algorithm.

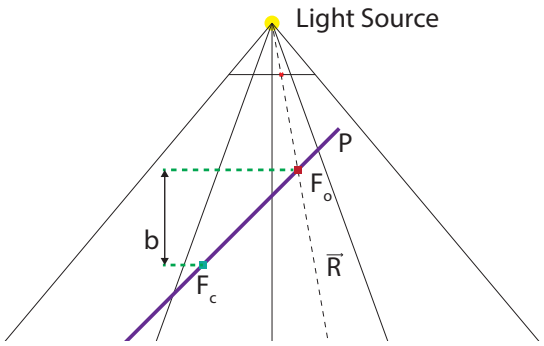


Figure 2: Illustration of the adaptive bias for PCF computation. F_c represents the current fragment, F_o is the potential occluder for the shadow map / filter kernel texel offset by 1 texel "to the right". b is then the optimal bias for this texel in the filter kernel.

Still making the simplification of taking the tangent plane defined by the current fragment and its normal as the underling geometry, we can compute the optimal bias for any texel in the PCF filter kernel. We therefore define the light ray $\overrightarrow{R}$ (which is intersected with the tangent plane P), by the light source and the texel center of the corresponding shadow map texel in the filter kernel instead of the texel center of the shadow map texel corresponding to the current fragment. See Figure 2 to see a basic illustration.

But for the PCF filtering this means, that the adaptive bias computation needs to be done for any texel in the filter kernel. This leads to a significant performance impact (For an 11x11 filter kernel, rendering time was about 5x as high as without adaptive bias. Since the computation is done for any texel in the filter kernel, it gets worse as the filter kernels get bigger.). So some optimization was needed, to be able to use the adaptive bias with PCF.

Suppose that the light direction is the same for any texel in the filter kernel. This is exactly true for an orthogonally rendered shadow map, when a directional light is used, but also holds nearly unchanged for shadow maps rendered using a perspective projection, since filter kernels do not get too large compared to the whole scene. Then the depth difference between the potential occluder for any texel in the filter kernel and the potential occluder for one texel further in x- or y-direction is the same as for any other texel in the filter kernel. And therefore, the depth difference for a texel which is offset n texels in x- or y-direction, it is the same corresponding value times n. So the bias for a texel in the filter kernel can be obtained as in equation 2.

$$bias = bias(F_c) + n \times \Delta bias_X + m \times \Delta bias_Y \quad (2)$$

If we know the depth differences, we will not have to compute the potential occluder again for each texel in the filter kernel. We could simply compute the potential occluder for the original texel, and than add the according value for the x- and y-direction multiplied by the offset - measured in texels - in x- and y-direction. See Figure 3 for clarification.

These values, from now on called $\Delta bias_X$ and $\Delta bias_Y$, can be obtained by computing the potential occluder for the original texel, and for the texels offset in x-direction and y-direction by one shadow map texel. With these three potential occluders, we can obtain the $\Delta bias$ for x- and y-direction. When PCF filtering, the potential occluder for each texel in the filter kernel is the already computed potential occluder for the original texel plus $\Delta bias_X$ times the offset in x-direction plus $\Delta bias_Y$ times the offset in y-direction. With this approach, the potential occluder needs to be calculated three times for a filter kernel, regardless of the size of the filter kernel, instead of for each texel in the kernel. This leads to the complete algorithm looking like Algorithm 1, and reduces the performance loss to reasonable 27 %.

4.2 PCSS

The whole PCSS algorithm has many similarities with PCF, since it is an extension to the PCF algorithm. On the one hand the final filtering is a PCF filtering, so the whole adaptive bias enhancement for PCF can be reused on this stage, and on the other hand the initial blocker search step does not differ from PCF in any step that is of importance for the adaptive bias enhancement, so this is no problem as well.

The blocker search step is identical to a PCF filtering, since, in a given filter kernel, the depth of the fragment

 ISBN 978-80-86943-65-7

Algorithm 1 optimized PCF with Adaptive Bias

$SM \leftarrow generateShadowMap(LightPosition)$
for *each fragment F with normal N* **do**
  $isLit \leftarrow 0$, $n_{shadowTests} \leftarrow 0$
  $F_{o_{original}} \leftarrow calculatePotentialOccluder(F)$
  $F_{o_{X+1}} \leftarrow calculatePotentialOccluder(F + (1\ texel\ in\ SM\ x-direction))$
  $F_{o_{Y+1}} \leftarrow calculatePotentialOccluder(F + (1\ texel\ in\ SM\ y-direction))$
  $\Delta bias_X \leftarrow z(F_{o_{X+1}}) - z(F_{o_{original}})$
  $\Delta bias_Y \leftarrow z(F_{o_{Y+1}}) - z(F_{o_{original}})$
  for *each texel T in filterkernel* **do**
    $F_o \leftarrow F_{o_{original}} + \Delta bias_X \times xOffset + \Delta bias_Y \times yOffset$
    $\varepsilon \leftarrow calculateAdaptiveEpsilon(F_o)$
    $isLit \leftarrow isLit + shadowTest(SM, F_o, \varepsilon)$
    $n_{shadowTests} \leftarrow n_{shadowTests} + 1$
  end for
  $isLit \leftarrow isLit / n_{shadowTests}$
  $fragColor \leftarrow isLit \times shadeFrag(F)$
end for

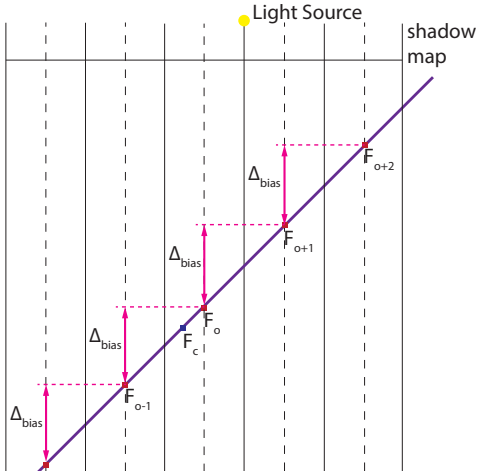


Figure 3: Illustration of the delta bias. Assuming the light direction is the same for any texel in the filter kernel, the depth difference between the potential occluders of neighboring texels is the same for any texel. (But different in x- and y-direction.)

is compared to the depth saved in the corresponding shadow map texel. There are two differences: firstly, the filter kernel for the blocker search differs from the PCF filter kernel. And secondly, not the results of the shadow tests are saved and then averaged as with PCF, but for all texels which are shaded according to the shadow test, the depth values of the blockers are collected and averaged. Since this difference has nothing to do with the biasing, the same modifications as for PCF with adaptive bias are suitable here.

Of course, the initial idea for a combination without the optimization already made with PCF, brings an even bigger performance penalty than with PCF shadows, since two PCF filters are applied for any texel in the final image. The actual PCF filtering for creating a soft shadow and the blocker search, which - in terms of PCF - depending on the light's position and size and the objects, often uses very large filter kernels.

But with the same $\Delta bias$ optimization as with PCF (and Poisson Disc sampling in the blocker search), the performance impact can be put into reasonable bounds. The $\Delta bias$ only needs to be computed once in the beginning, and can then be used for both the blocker search, and the final filtering. This does not introduce any performance loss in the blocker search, since, for standard PCSS, a receiver plane depth bias [Isi06a] should already be used in this stage to obtain clean results. And taken the $\Delta bias$ as given here, the computational effort to obtain the bias is very similar to the one that obtains the receiver plane depth bias. The final PCF filtering performs as the optimized enhancement to PCF , which it actually is, and is therefore of adequate speed. The overall performance impact is, since there is none in the blocker search, as with PCF 27%.

5 MAKING THE ADAPTIVE BIAS MORE ROBUST AGAINST PROJECTIVE ALIASING

As the sampling density of the shadow map is not adapted, the adaptive bias algorithm still suffers from projective aliasing. By increasing the value for K, the aliasing is reduced, however light leaking occurs (Figure 11a-d).

Projective aliasing appears in areas where the surface becomes parallel to the light direction. These areas can be detected with a scalar product between the light direction and the fragment normal vector. We use these scalar product to scale the adaptive bias in order to reduce the projective aliasing. This leads to the following enhanced formula for the adaptive epsilon:

$$\varepsilon = \frac{(lf-depth \times (lf-ln))^2}{lf \times ln \times (lf-ln)} \times sceneScale \times K \times scaleFactor \quad (3)$$

$$scaleFactor = min\left(\frac{1}{(lightDirection \cdot normal)^2}, 100\right) \quad (4)$$

So by using this scale factor, the epsilon gets scaled up as the surface and the light direction become more parallel. The square of the scalar product simply comes from the fact, that if we only use the scalar product, the epsilon does not scale fast enough. And introducing another constant that may not be suitable for different scenes is not a good idea. The value of 100 that limits the scaling is thus not a constant in the sense that it

needs to be hand tweaked for different scenes, it simply causes the adaptive epsilon to not scale further when the angle between light direction and surface normal is over 84.26° (this value is computed from the threshold of 100, it is not a meaningful value itself). Figure 9 shows what happens with projective aliasing artifacts for different thresholds. For too small threshold values, artifacts due to projective aliasing remain present, while overlarge thresholds produce light leaking problems. Also the range of threshold values producing good results for this scene is very large, it may be significantly smaller for other scenes. We have chosen 100 as threshold, since it firstly is, compared to the whole range of threshold values producing good results, a rather small value, so no unnecessary large scaling factors and therefore unnecessary large biases are used. Secondly 100 has proven itself as a good value for all scenes we used for testing (any image in this paper is rendered with 100 as threshold), also for the scenes for which the range of thresholds giving good results is much smaller than in the above example. Even though the value of 100 worked for all our scenes, there is no guarantee it would work for any scene, so it still needs to be hand tweaked. If we did not limit the scaling factor (or if a overlarge threshold is used), it would get very large, leading to visible strips of light in these areas, since we would use a very large bias due to the large factor, which would additionally stack with the light leaking problem which already comes with the original bias. Note that this does not make the epsilon completely adaptive. The value of K has still to be adapted for each scene. However, the range of scenes where a given value of K performs well is extended compared to the original definition in [Dou14].

6 RESULTS AND DISCUSSION

We will now show results and performance of our methods. The implementation was done with OpenGL and GLSL shaders. The scenes were rendered on an Intel Core i7 with 4 GHz and a NVIDIA GTX770 graphics card. All images where rendered with a resolution of 1024x768. We will follow the same order as before, and go firstly through the results of the combination of the adaptive bias and PCF, then the combination with PCSS and finally show the results from the enhancement against projective aliasing.

6.1 PCF

Since Dou et al. explicate in [Dou14] how the adaptive bias preserves more shadow detail than other biasing methods and therefore increases the quality of the shadows, we solely focus on the quality of the PCF shadow, and therefore use a suitable scene.

A naive PCF implementation is used, meaning any texel in the filter kernel is sampled. Figure 4 shows the comparison between PCF with receiver plane depth bias, the adaptive bias and our optimized adaptive bias for PCF. The difference image in Figure 4 shows that there are small differences in the self shadow, however no differences in the shadows cast on the plane. In Figure 5 a comparison between a PCF shadow - emerging from being on the backside (from the light's point of view) of an object - with receiver plane depth bias and the optimized adaptive bias can be seen. In this case the optimized adaptive bias gives a significantly better result. Furthermore Figure 8c demonstrates that also with a nonplanar shadow receiver, our combination of PCF and the adaptive bias produces good results. In addition in figure 10 you can see a comparison of PCF with receiver plane depth bias and PCF with our adaptive bias on the complex sponza scene. Due to the left artifacts and the less preserved shadow detail, PCF with our adaptive bias clearly outperforms PCF with receiver plane depth bias.

Table 1 shows the corresponding performance for the scene in the above mentioned Figure 4. As displayed, the adaptive bias is extremely slow compared to the receiver plane depth bias. But the optimized adaptive bias brings the performance back into reasonable boundaries, since the optimized adaptive bias does not cost more than 27% more overall rendering time compared to the receiver plane depth bias, depending on the filter kernel size.

For the performance comparison with different shadow map resolutions, see Figure 7b. As you can see the rendering time of both, the adaptive bias, and of course the more interesting optimized one, comparatively do not increase faster, instead, the rendering times of the different biasing methods converge for higher shadow map resolutions.

6.2 PCSS

The PCSS implementation without adaptive bias uses a receiver plane depth bias, for both the blocker search and the final PCF filtering. The PCSS with adaptive bias uses the optimized adaptive bias on both stages. The not optimized adaptive bias is excluded in this stage, since the PCF section proves that the optimized one produces equally good results, and PCSS with the unoptimized version is extremely slow. Both implementations use Poisson Disc sampling with 25 samples in the blocker search, and a naive PCF implementation for the final filtering.

Figure 6 shows a comparison of PCSS with and without the adaptive bias. And again, you can clearly see, that all results are equally good, which proves that not only the PCF enhancement, but also the enhancement of the adaptive bias for PCSS, works. The difference image shows some minor differences at the shadow boundaries, resulting from the blocker search, and which are not even visible with the naked eye. Table 2 shows the

corresponding performance measures. The optimized adaptive bias is of reasonable performance, as it is again about 27% slower than the receiver plane depth bias version.

Figure 7a shows the performance under different shadow map resolutions. As you can see, the performance of PCSS with the optimized adaptive bias does not lag a lot behind the performance of PCSS with receiver plane depth bias and scales equally well. Actually, for higher shadow map resolutions, the computational extra cost reduces, as for a shadow map resolution of 8192^2 the optimized adaptive bias is only about 12% more expensive.

6.3 Making the adaptive bias more robust against projective aliasing

Figure 8 shows a simple scene, that demonstrates where projective aliasing causes problems with the original [Dou14] algorithm, and that the enhancement against projective aliasing is able to remove these artifacts. In Figure 11 the complex island scene is pictured with and without the enhancement against projective aliasing and rendered by a ray tracer as reference, and it is clearly visible that it looks a lot better with the enhancement. There are falsely lit points, but as you can see in the picture with the original adaptive epsilon, they mostly come from the light leaking problem that already comes with the adaptive bias algorithm. Some very few additional light points are introduced by the enhanced adaptive epsilon. These are so scarce, that, assuming that the light leaking problem that comes with the original algorithm is not a big issue as stated in [Dou14], we claim that this is still no problem. Additionally Figure 10 shows the sponza scene, as another complex scene. As in Figure 11, the result of rendering with the enhancement against projective aliasing looks much cleaner.

In the comparison of the different views from the island scene in Figure 11, you can see that now, while the "backside" of the scene where projective aliasing still was a huge problem looks a lot better, we have no problem with shadow detachment on the "frontside". This was not possible without the enhanced adaptive epsilon, since, with the original adaptive epsilon, a constant value of K that was large enough to remove the projective aliasing already caused shadow detachment in other parts of the scene, and the other way around, a value that did not cause any shadow detachment left projective aliasing.

7 CONCLUSION

In this paper, we made the adaptive bias algorithm of Dou et al. more robust against projective aliasing and presented a strategy for incorporating it into soft shadow algorithms such as PCF and PCSS. Our idea is to calculate the potential occluder only once for each texel and interpolate it for the kernel offsets of a PCF filter. This results only in a small performance penalty compared to the receiver plane depth bias.

Furthermore, we extended the adaptive bias algorithm with a light dependent factor in order to make it more robust against projective aliasing. However, the sampling density of the shadow map is not increase and therefore, projective aliasing is still present. In order to reduce the projective aliasing further, adaptive partitioning approaches, such as queried virtual shadow maps [Gie07a], are required.

In future work, we wish to replace scene dependent parameters, such as the scene scale, with scene independent parameters in order to avoid parameter tweaking for multiple scenes. Furthermore, we wish to increase the performance of the technique when used in soft shadow algorithms.

8 REFERENCES

[Dou14] Dou, H., Yan, Y., Kerzner, E., Dai, Z., and Wyman, C. Adaptive depth bias for shadow maps. In Conf.proc ACM Symposium on Interactive 3D Graphics and Games. 2014

[Eis11a] Eisemann E., Schwarz M., Assarsson U., Wimmer M. Real-Time Shadows, Taylor & Francis, 2011.

[Fer05a] Fernando R. Percentage-Closer Soft Shadows. ACM SIGGRAPH 2005 Sketches, 2005.

[Gie07a] Giegl, M., and Wimmer, M Queried virtual shadow maps. In Conf.proc ACM Symposium on Interactive 3D Graphics and Games, 2007.

[Isi06a] Isidoro J. R. Shadow Mapping GPU-based Tips and Techniques. GDC 2006, 2006.

[Ree87a] Reeves W. T., Salesin D. H., Cook R. L. Rendering antialiased shadows with depth maps. In Conf.proc SIGGRAPH '87, ACM, 283-291, 1987.

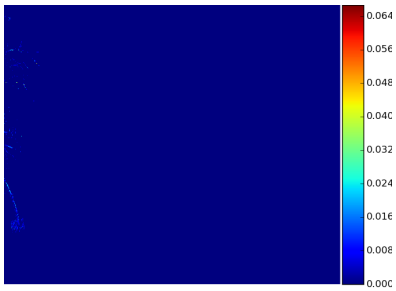
[Wei03a] Weiskopf, D., and Ertl, T. Shadow mapping based on dual depth layers. In Conf.Proc. Eurographics, ACM, 173-180, 2003.

[Woo94a] Woo, A. The shadow depth map revisited. In Graphics Gems III, Morgan Kaufmann, 338–342, 1994.

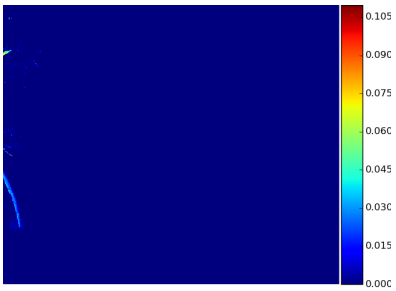
(a) receiver plane depth bias

(b) adaptive bias

(c) optimized adaptive bias

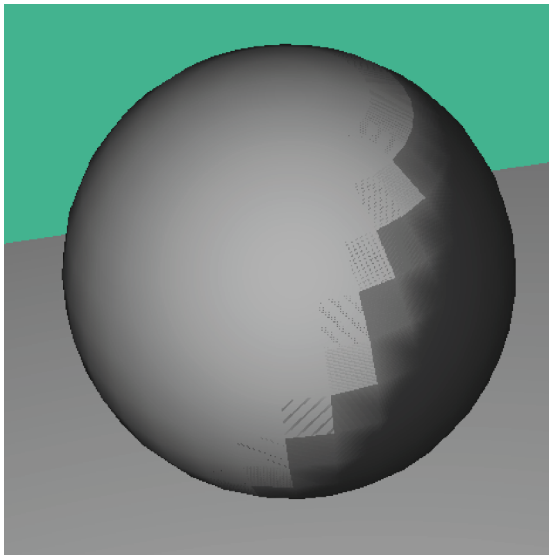


(d) difference image between (a) and (b)

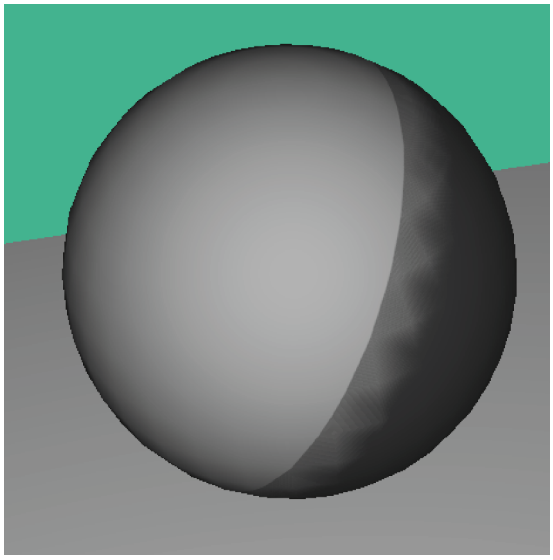


(e) difference image between (a) and (c)

Figure 4: Comparison between a PCF shadow (which is actually cast) with an 11x11 filter kernel and with (a) receiver plane depth bias, (b) adaptive bias and (c) the optimized adaptive bias. The difference image between the receiver plane depth bias and the adaptive bias is shown in (d) or the optimized adaptive bias in (e). There is no difference, which means the quality of our approach is at least as good as these approaches.



(a) receiver plane depth bias



(b) optimized adaptive bias

Figure 5: Comparison between a PCF shadow (emerging from being on the side turned away from the light) and with (a) receiver plane depth bias and (b) the optimized adaptive bias. The optimized adaptive bias gives a significantly better result.

(a) receiver plane depth bias (b) optimized adaptive bias (c) optimized adaptive bias

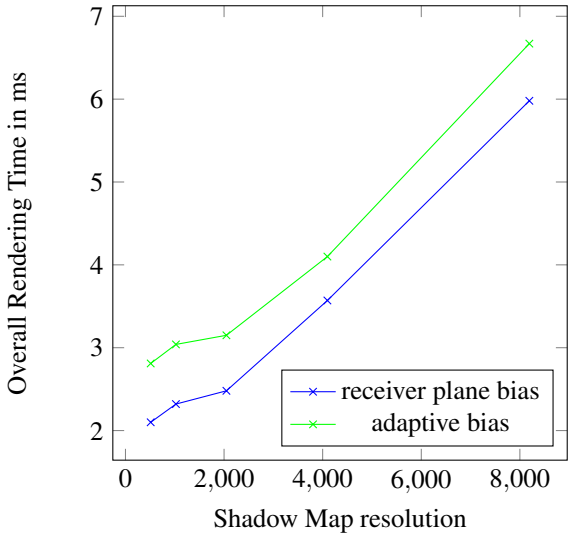
Figure 6: Comparison between PCSS with (a) receiver plane depth bias and (b) optimized adaptive bias. In (c) the difference image can be seen. There are very few differences at the shadow outlines, still resulting in a shadow of equal quality.

biasing method	kernel	Shadow Map	Final Shading	Overall
receiver plane depth bias	5x5	0.44ms	0.74ms	1.18ms
adaptive bias		0.44ms	1.38ms	1.82ms
optimized adaptive bias		0.44ms	0.86ms	1.30ms
receiver plane depth bias	7x7	0.44ms	1.05ms	1.49ms
adaptive bias		0.44ms	2.49ms	2.93ms
optimized adaptive bias		0.44ms	1.38ms	1.82ms
receiver plane depth bias	11x11	0.44ms	2.27ms	2.71ms
adaptive bias		0.44ms	5.75ms	6.19ms
optimized adaptive bias		0.44ms	3.01ms	3.45ms

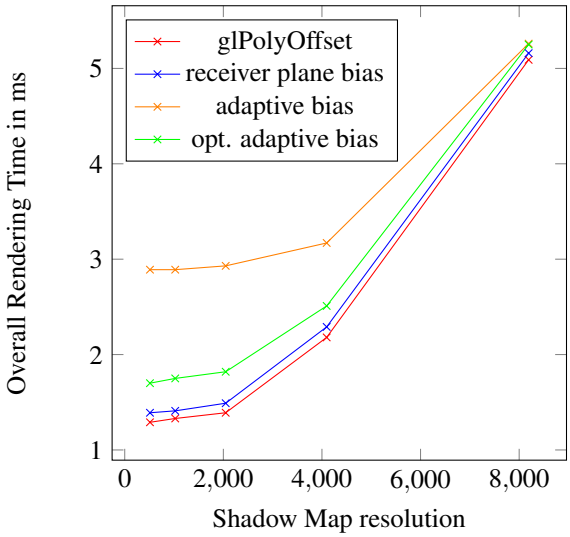
Table 1: Performance measurement of PCF with different filter kernel sizes. The shadow map resolution was constantly 2048^2.

biasing method	light size	Shadow Map	Final Shading	Overall
receiver plane depth bias	small (0.05)	0.44ms	3.61ms	4.05ms
adaptive bias		0.44ms	5.11ms	5.55ms
plane bias & poisson		0.44ms	1.92ms	2.36ms
adaptive bias & poisson		0.44ms	2.57ms	3.01ms
receiver plane depth bias	medium (0.10)	0.44ms	13.65	14.09ms
adaptive bias		0.44ms	19.10ms	19.54ms
plane bias & poisson		0.44ms	2.04ms	2.48ms
adaptive bias & poisson		0.44ms	2.71ms	3.15ms
receiver plane depth bias	large (0.16)	0.44ms	34.74ms	35.18ms
adaptive bias		0.44ms	48.51	48.95ms
plane bias & poisson		0.44ms	2.17ms	2.61ms
adaptive bias & poisson		0.44ms	2.86ms	3.30ms

Table 2: Performance measurement of PCSS with different light sizes in the tree scene. The shadow map resolution was 2048^2 for all measurements. Receiver plane depth bias according to [Isi06a]. If not specifically mentioned naive sampling is used, meaning that any texel in the filter kernel was sampled, poisson means Poisson disk sampling was used.

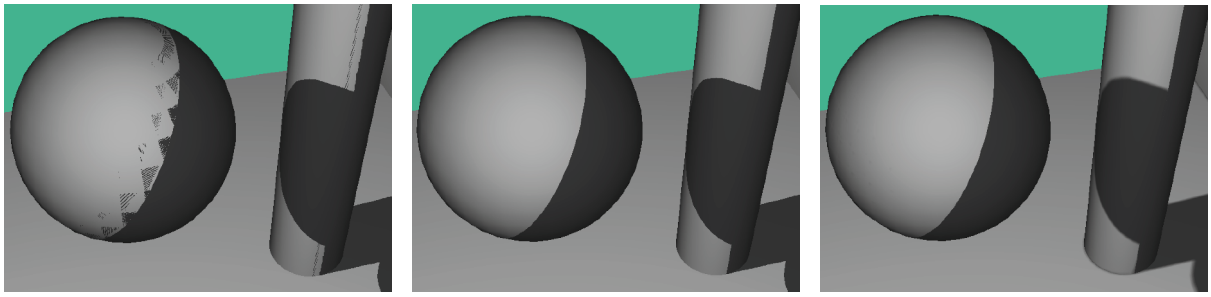


(a) Performance of PCSS with optimized adaptive bias and receiver plane depth bias with Poisson Disc sampling under different shadow map resolutions.



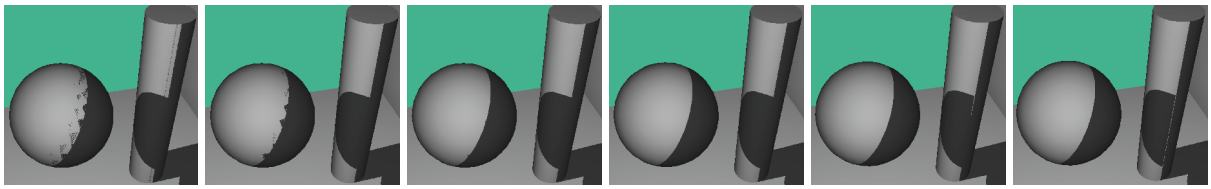
(b) PCF Performance with different biasing and varying shadow map resolutions.

 ISBN 978-80-86943-65-7



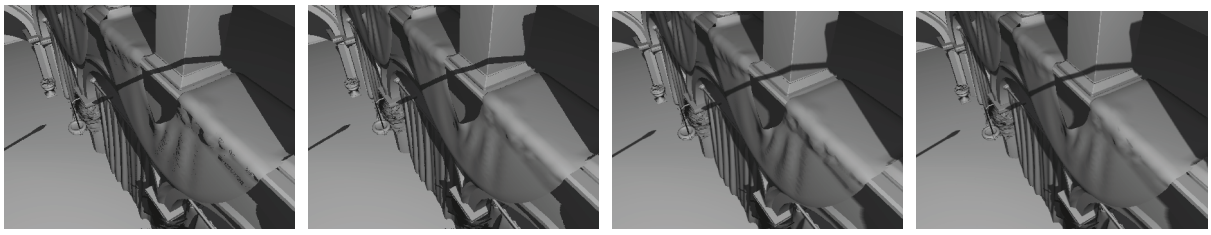
(a) original [Dou14] algorithm (b) enhanced adaptive epsilon (c) enhanced adaptive epsilon with PCF (19x19 filter kernel)

Figure 8: Casted shadow on a nonplanar surface. While the original [Dou14] algorithm (a) still has artifacts, the enhanced adaptive epsilon (b) has no visible arifacts, and generates good results on the nonplanar shadow-receiver, even with PCF filtering with large filter kernels (c).



(a) threshold of 1 (b) threshold of 10 (c) threshold of 50 (d) threshold of 100 (e) threshold of 1700 (f) threshold of 2000

Figure 9: Different threshold values for the scale factor, from very low values to very high values. It is clearly visible, that a too small threshold results in remaining artifacts, while overlarge values results in lightleaking. These values cover a very large range in which good results are produced for this scene, but this range might be significantly smaller for other scenes.

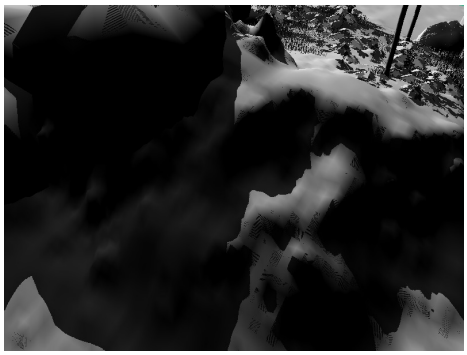


(a) original [Dou14] algorithm (b) enhanced adpative epsilon (c) PCF with receiver plane depth bias (d) PCF with our adaptive bias

Figure 10: Complex sponza scene for comparison. Comparison of the original [Dou14] algorithm (a) and the enhanced adaptive epsilon (b). The original algorithm still suffers from projective aliasing, while the enhanced adaptive epsilon creates a satisfying result. Comparison of PCF with receiver plane depth bias (c) and PCF with our adaptive bias (d). With the receiver plane depth bias there are still artifacts left, while less shadow detail is preserved.

(a) original adaptive epsilon, $K = 0.0001$

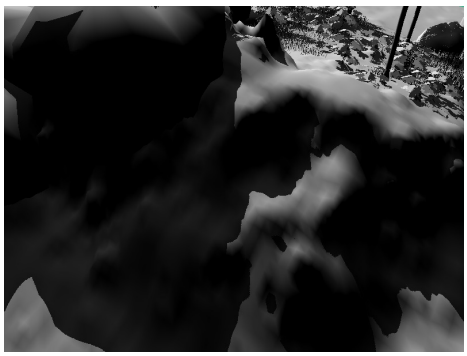
(b) tree trunk



(c) original adaptive epsilon, $K = 0.0001$

(d) original adaptive epsilon, $K = 0.01$

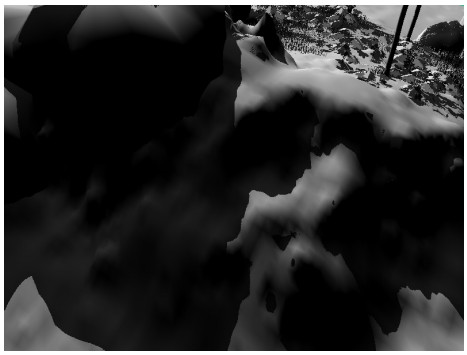
(e) tree trunk



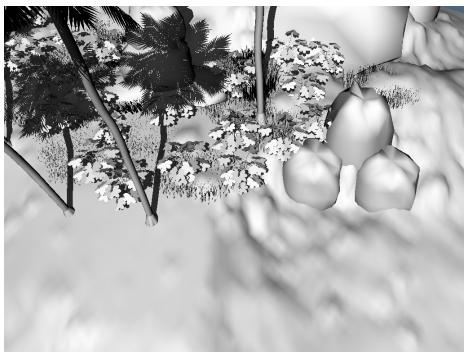
(f) original adaptive epsilon, $K = 0.01$

(g) enhanced adaptive epsilon, $K = 0.0001$

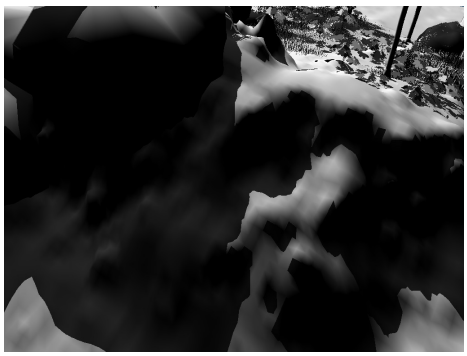
(h) tree trunk



(i) enhanced adaptive epsilon, $K = 0.0001$



(j) ray tracing result



(k) ray tracing result

Figure 11: The island scene, with in (a) - (c) the original epsilon producing good results in (a) but with projective aliasing on (c) due to the grazing angle of the light. In (d) - (f) the original formula is still used, but with $K = 0.01$ which reduces the projective aliasing in (f), but causes shadow detachment (tree trunk shadows) in (d). In (g) - (i) the enhanced adaptive epsilon is used, causing good results for the same value of $K = 0.0001$. In (j) and (k) are rendered with a ray tracer as a reference implementation.

 ISBN 978-80-86943-65-7

Hybrid client-server and P2P network for web-based collaborative 3D design

Caroline Desprat
University of Toulouse
118 Route de Narbonne
Toulouse, France
desprat@irit.fr

Hervé Luga
University of Toulouse
118 Route de Narbonne
Toulouse, France
luga@irit.fr

Jean-Pierre Jessel
University of Toulouse
118 Route de Narbonne
Toulouse, France
jessel@irit.fr

ABSTRACT

Our proposed research project is to enable 3D distributed visualization and manipulation involving collaborative effort through the use of web-based technologies. Our project resulted from a wide collaborative application research fields: Computer Aided Design (CAD), Building Information Modeling (BIM) or Product Life Cycle Management (PLM) where design tasks are often performed in teams and need a fluent communication system. The system allows distributed remote assembling in 3D scenes with real-time updates for the users. This paper covers this feature using hybrid networking solution: a client-server architecture (REST) for 3D rendering (WebGL) and data persistence (NoSQL) associated to an automatically built peer-to-peer (P2P) mesh for real-time communication between the clients (WebRTC). The approach is demonstrated through the development of a web-platform prototype focusing on the easy manipulation, fine rendering and light update messages for all participating users. We provide an architecture and a prototype to enable users to design in 3D together in real time with the benefits of web based online collaboration.

Keywords

WebRTC, WebGL, collaborative, Peer-to-peer, Applications, Web

1 INTRODUCTION

As Ortiz [OJ10] questioned if "3D is finally ready for the web?", the Internet responds with a large amount of creation, transmission, storage and access solutions for 3D contents [ERB14]. 3D CVEs (Collaborative Virtual Environments) are representative of this increasing popularity in industry and makers communities ; due to the competition and the mobility nowadays, people are turning faster toward the optimal resources. A good example of this trend is the emerging market of 3D collaborative modelers, server-based such as GradCAD, ThinkerCAD, Sunglass.io, Clara.io [HLL+13], Verold Studio or cloud-based like AutoCAD360. The collaborative aspect in 3D modeling CVEs shows the need of an efficient cooperation over the network between the users. Even if they are geographically far and have different points of interest, they have the same shared goal: the manufacture of the product.

This research is led by the desire of working collaboratively and sharing 3D scenes across the network. Large scenes or complex models are very likely to be constructed and reviewed by more than one person, especially in the context of 3D design, PLM (Product Lifecycle Management) or BIM (Building Information Modeling) solutions. The new usages and the increasing mobility of workers are pointing to web based solutions. Moreover, in small designing teams, we can observe that the design process is conducted with direct communication channels. The mimic of direct communication in computing is peer-to-peer (P2P): why to pass through a proxy when the team members are so close? We can also observe that the need of persistent communications is mandatory with this running. Since the network speed can be a limiting factor in collaborative design, one of the main criteria for our system is to spread and display only relevant information between the users without overloading the server.
The contributions of this work are multi folds:

- consider the solutions for plug-in-less visualization of 3D scenes on the web,
- use efficiently the local client resources for visualization and storage,
- allow small and asynchronous message system,
- overcome the difficulties related to interactive collaboration across the network with shared access to 3D models with bandwidth limits of the actual connections.

The reminder of this paper is organized as follows: the related work in distributed collaborative modeling is in section 2, our model architecture is described in section 3, the implementation of the web editor, the server architecture with the storage mechanisms and the P2P

 ISBN 978-80-86943-65-7

communication layer with the synchronization are explained in section 4. Then we introduce some examples of collaboration on 3D scenes with our model including a discussion about the network and display optimizations of the system regarding the user experience (field-speciality, pieces that "matters" to the user, client resources, device used...) in section 5. Finally, conclusion and future works are given in section 6.

2 RELATED WORK

CAD is an essential tool for 3D models production in industry. During the last years, considerable time and resources were spent on CAD as well as in the improvement of the computers power. These two features associated with a increasing need of team working and professional mobility, enabled the development of several Internet based collaboration tools.

2.1 Web-based visualization and collaboration

A wide range of standards and technologies have emerged the last decade for web-based and mobile 3D visualization. With HTML5 and more powerful clients, solutions that do not require the installation of a software are now well admitted on the web (unlike Flash, Unity3D[1]). Two predominant pluginless approaches exist: imperative with WebGL[2] supported by the W3C[3] and declarative [SKR+10] with X3D [JBDW12]. Mouton and al. [MSG11] sum up a coverful analysis of current systems and trends in CVEs [Fle12] arguing that web applications have major benefits over desktop applications because they are available for all major platform guaranteeing a cross-platform compatibility (including mobile devices) and do not require any software or libraries (except the web browser). They highlighted the need for new applications to reduce their bandwidth consumption by using local client resources to increase performances (interactivity). Web-based collaboration is particularly present in scientific visualization [JFM+08][GGCP11][CSK+11], cultural heritage [DBPM+14] and CAE (Computer-Aided Engineering) applications [CCW06]. This last one offers many online collaborative and distributed modelers like GradCAD, ThinkerCAD, Sunglass.io, Clara.io [HLL+13] or Verold Studio. However, most of these popular systems are client-server based and rely on full transfer of large 3D data for each client.

2.2 Web-based networking

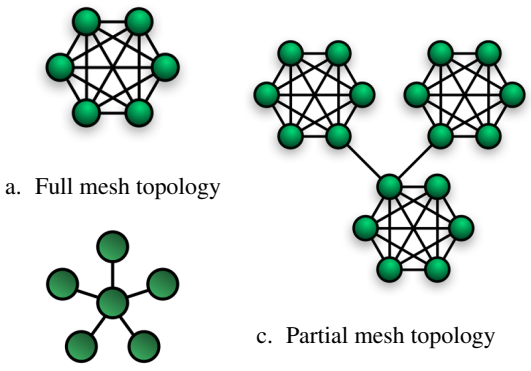
2.2.1 *Client-Server*

The client-server network topology puts the different clients in relation via the server that manages, transforms and stores the modifications and the data using a persistent database. This type of network offers security and easy management of information.

In 2011, Gutwin and al. [GLG11] has exposed the increasing role of web browsers as *"a platform for delivering rich interactive applications"* and details the different web-based networking approaches. They are all client-server types: HTTP-based communication (sending/getting server requests), AJAX with XHR (requesting without page reloading) and WebSockets [Rak14] (keeping an open connection with the server and communicate through messages). The works of Marion [MJ12] and Grasberger and al. [GSWG13] are based on the WebSocket protocol [FM11]. [MJ12]'s proposition transfers scientific data to and between clients to visualize with WebGL. The users can work on the data concurrently but they cannot edit it unlike in [GSWG13] where a BlobTree functional representation method requiring small memory footprint messages is also used to store, transfer, visualize and edit data.

2.2.2 *Peer-to-peer*

The P2P topology network allows each peer to be client and server simultaneously and to communicate directly with each other. The P2P network topology offers better resilience in case of a system crash or incongruous network disconnection using the autonomy of the peer (and data replication in the mesh). This induces higher efficiency in communication between team members (direct communication like in the real world).



a. Full mesh topology

b. Star Topology

c. Partial mesh topology

Figure 1: Peer-to-peer topologies

Full mesh topology (Figure 1.a) connects each nodes to every other one. It requires that every time a new peer join the network, the other peers establish a connection with the new peer. The increase in the number of connection is exponential and it does not scale well, saturating the bandwidth.

Star topology (Figure 1.b) uses a star node that distributes the data to the others. That node is a very

[1] http://unity3d.com/5

[2] http://www.khronos.org/webgl/

[3] World Wide Web Consortium – http://www.w3.org/

high bandwidth consumer and could be a dedicated server. This client-server like topology removes the advantage of P2P distribution but keeps the benefit of having reliable messages.

Partial mesh topology (Figure 1.c) connects the nodes "indirectly" to each others: one device maintains multiple connections to others without being fully meshed. Partial mesh topology provides redundancy by having several alternative routes and needs good recovery mechanism to maintain the data transmission in the mesh.

The development of P2P communication between browsers arrived in 2011 with the drafts of WebRTC (Web Real-Time Communication) API [BBJN12] of the W3C and IETF®[4]. Many projects with WebRTC technology [BBJN12] are interested in the MediaStream API (audio and video streaming) but only a few are using the DataChannel part. Services like PubNub[5], very popular to set up real-time applications with WebSockets, are just starting to support WebRTC. [WV14] presents the WebRTC architecture foundations for decentralized content-publishing facility between browsers with concerns about security and privacy.

ROCCAD [CT07] is a prototype providing a 2D/3D graphics interchanges in real-time during a development process for synchronous design collaboration. It offers distributed mechanisms to handle data transmission, data access policy and conflict resolutions, users management based on Tree First P2P overlay network over TCP/IP. The communication architecture exposed in [KVaD14], is very powerful and scalable using a client manager to abstract and synchronize the different devices communicating in P2P, where the servers are supporting the LODs (Level Of Details) management.

Chen and al.[CH14] presented an asynchronous online collaboration for BIM generation using hybrid client-server and P2P network based on a hierarchical topology: a peer team appointed a local server to transmit data to the global server. This architecture offers a good scalability in collaboration with parallel modeling (intra-disciplinary) to achieve a single multi-disciplinary task. Moreover, design team members can share their work in modeling and cooperate while working concurrently. The critical points are the servers: the local server could be overloaded (it is the only proxy to reach the global server) and if the global server suddenly goes down, the inter-collaboration is broken due to conflict generation between sub-models avoiding (teams) to communicate.

[4] The Internet Engineering Task Force – https://www.ietf.org/

[5] http://www.pubnub.com/

3 HYBRID ARCHITECTURE FOR 3D MODELING COLLABORATION

Our collaborative design environment (CoDE) requires an appropriate network model. The two main types of communication networks on the web are client-server and P2P. Even if client-server is more common, P2P is coming more and more attractive because of its characteristics of decentralized control and self-organisation although its web standards are still on progress.

To set up our CoDE, we developed a full web-based communication architecture for a 3D modeling platform. This work is lead in the context of a small amount of users (small teams max 7/8 people) which means full mesh topology is adapted (direct reachability) and its exponential growth is negligible. Indeed, with more users a partial mesh topology should fit better to relieve the network congestion. In such a virtual workspace, the contributions of each user is directly transmitted to others and they can observe the doings of others in real-time. The network model is mixing conventional client-server architecture, mostly used for persistence, and a full mesh P2P network for the real-time data transmission between the clients.The users are working together on a scene where they can add, remove and update 3D models.

3.1 Web-based 3D editor

The 3D rendering framework is based on WebGL which is pluginless. The framework is able to handle 3D data stored locally and on an external server for persistence and synchronization. The 3D viewport editor allows users to view and interact with the model. The interactions offered to the user are:

Viewing, navigating and using transformations tools The user can lean on commons commands from known CAD programs to interact with the view and the camera. It uses classic handles for object translation, rotation and scaling, and conventional CAD navigation.

Uploading 3D models, textures The editor handles the most used of open 3D file formats in CAD [Bou12] (OBJ, PLY, DAE, JSON...) via user friendly interaction: drag and drop importation.

Referential modification The modification of the reference coordinate system from local to global for transformation can be helpful for designer.

Grip snapping The definition of the grid can be modified by the user to get a specific resolution. It is also possible to use it to align models on the grid points.

Switching point of view The user can switch from his camera to other's users point of view.

 ISBN 978-80-86943-65-7

Assembled 3D models (textured or not) are available to every users (viewers, collaborators, editors). The interface is designed to be easy and clear to provide a better user experience (non-repulsive) particularly for the non-experts users for enhancing the accessibility of the application.

A content access policy for the objects is necessary to avoid modification conflicts during the collaboration. We use a lock/unlock mechanism with a visual feedback associated to represent that the object is in use (selection state) by a user to prevent concurrent edition of the same object.

3.2 Server: RESTful architecture

The client-server part is based on a REST (Representational State Transfer) architecture [TV10] that benefits from distributed hypermedia systems such as our. The responsibilities are separated between the client (user interface) and the server (data storage interface). Each request from client to server contains all the necessary information to let the server understand the request without context dependency stored on the server. With its uniform interface, each resource is unitarily identified with defined representations and auto descriptive messages. Also, the caching exempts many client-server interactions.

Table 1: REST architecture summary

Pros	Cons
Easier to maintain; No need to keep an open connection permanently; Web context: HTTP protocol, URI as resource representative, caching.	Bandwidth increasing and latency: client needs to keep locally all the necessary data to send the request.

Table 1 resumes the advantages and the drawbacks of a RESTful system. It fits well for web distributed systems even if mobile devices should have limited performance due to back and forth energy consuming requests.

NoSQL database

The rise of the web as a platform encourages the change in data storage for new needs like supporting large volumes of data (such as 3D data). NoSQL database provides dynamic schema and a rich query language API for data manipulation. Therefore the records can add new information on-the-fly facilitating the enrichment of the (3D) objects. In our application the NoSQL database is mainly used to maintain persistence of the state of the scenes while a user is absent. When the user returns, he/she receives the entire scene document. It provides robustness to the system and better experience to the user.

3.3 P2P communication

3.3.1 Topology

We propose to automatically connect users on a scene with a WebRTC connection. As each user send their ID to the database at their arrival, they also retrieve those which where already present on the scene. We are able to create a full mesh topology network in order to make them communicate the updates.

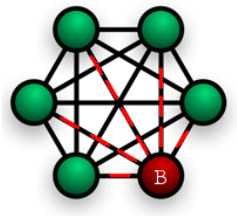


Figure 2: P2P topology of our model: a star node for message broadcasting inside a full mesh network.

Even if we have a full mesh topology, the P2P message layer is more similar the star topology. The Figure 2 shows the path of a sent message operation on the connection and it is only sent to the one-degree neighbors of the original broadcaster (“B” node on the Figure 2).

3.3.2 WebRTC and DataChannels

Web Real-Time-Communication (WebRTC) is a collection of standards, protocols (Figure 3) and JavaScript APIs specifying media transport and data sharing between browsers (peers)[Gri13]. P2P communication with *“WebRTC still needs servers for signaling”* and *“to cope with NAT and firewall”* [Ris14]. The signaling mechanism (Figure 4) allows peers to send control messages to each other in order to establish the communication protocol, the canal and the connection API.

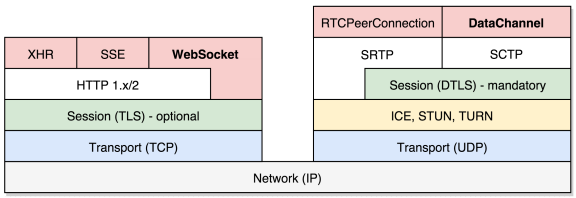


Figure 3: WebRTC protocol stack [Gri13]

We use the DataChannel protocol through the RTCDataChannel API to exchange arbitrary data between peers with customizable delivery properties (reliable or partially reliable, in-order or out-of-order) of the underlying transport [Gri13]. We choose to keep reliable and in order delivery for now. The RTCDataChannel API supports many data types (strings, binary types: Blob, ArrayBuffer...). These types are helpful in a 3D multi user environment to broadcast messages including the objects and their transformations. We tried to limit the amount of sent data with granularity choices (see Section 4) to prevent channel overfeeding.

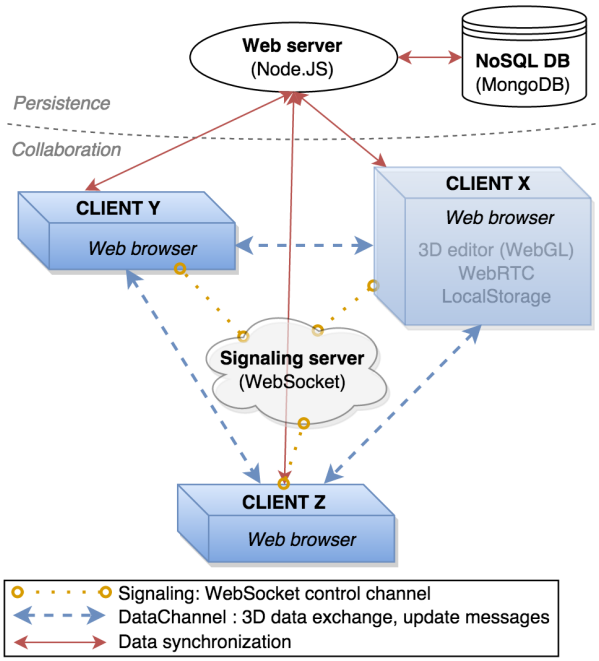


Figure 4: System overview

Some issues remain in RTCDataChannel API: the compatibility and interoperability is still not complete between browsers[6], some browsers (like Chrome) impose a send limit (about 6MB) for the data transmitting through DataConnections and the security of the communication is still vulnerable[7]. The system overview (Figure 4) illustrates the communication architecture topology between the peer clients, the web server and database (plus signaling server).

4 IMPLEMENTATION

As illustrated in the Figure 5, when a user arrives on the workspace editor the clients retrieves the scene from the server database. Each object is added to the Three.JS scene graph and rendered in the viewport. The connection to the P2P network with WebRTC is initiated by the assignation of an ID to the peer client which corresponds to the signaling mechanism. With this ID, the server automatically builds the full mesh topology by creating bi-directional connections between the new peer and the others. This action updates the list of connected users and their relations. Now that the scene is loaded and the P2P mesh is built, the user can freely interact with the scene with CRUD (Create, Read, Update, Delete) operations and synchronize the updates with the server and the other peers. For each operation, the type and the data are stored in a message according to the granularity of the transmission defined as follow:

- on import: all meshes and materials (such as textures) are sent.
- on transformation (translation, rotation, scale): the id of the transformed object and matrix of the transformation are sent.
- on delete: the id of the object to delete is sent.
- on lock/unlock: the id of the object is sent.

Once the message sent through the P2P mesh, the other peers can update their scene graph with the new values. It is also sent to the server through XMLHttpRequest for database persistence. When the client leaves the workspace editor, a flag is raised on the WebSocket server therefore it can broadcast the peers to update their list of connected peers to avoid useless sendings.

The implementation has a strong dependence on our architecture and the browser-based rendering constraints. WebGL is a JavaScript API provided by the Khronos Group. It is completely integrated into all the web standards allowing the browsers to use the GPU accelerated usage of image processing and effects as part of the web page. The choice of the 3D rendering framework has been oriented on an imperative solution: **Three.JS** [cab10]. It is a cross-platform solution that has already been widely adopted by the 3D community [MR10].

Our application is event-driven because of the nature of the manipulations of the users in a 3D environment. The event model is characterized by the event loop, event handlers and asynchronous processes. We based the message layer for user interface on a custom event/messaging system library called ***js-signals***[8]. Each signal has its own controller, which allows easy control of the event broadcaster and subscriber, avoiding the wrong objects reacts to the event. When a *Signal* instance is defined, procedures can be added to it. The signal will be intercepted anywhere in the scope of the application, the associated procedures will be triggered as well. A very interesting property of the event loop model is that JavaScript (unlike a lot of other languages) never blocks. A *Signal* is typically performed via events and callbacks, therefore when the application is waiting for a WebRTC message or an asynchronous XHR request to return, it can still process things like clicks.

As an asynchronous server-side run time environment, Node.JS replicates this model by using continuations: it keeps a stack of functions waiting to be run when the right event comes along. It is ideal for a data intensive real-time application that runs across distributed machines or a fast file upload client. Furthermore, with Node.JS we have JavaScript both client and server side, simplifying the understanding and the maintenance of

[6] WebRTC compatibility between web browsers from `http://iswebrtcreadyyet.com/`

[7] `https://github.com/diafygi/webrtc-ips`

[8] `http://millermedeiros.github.io/js-signals/`

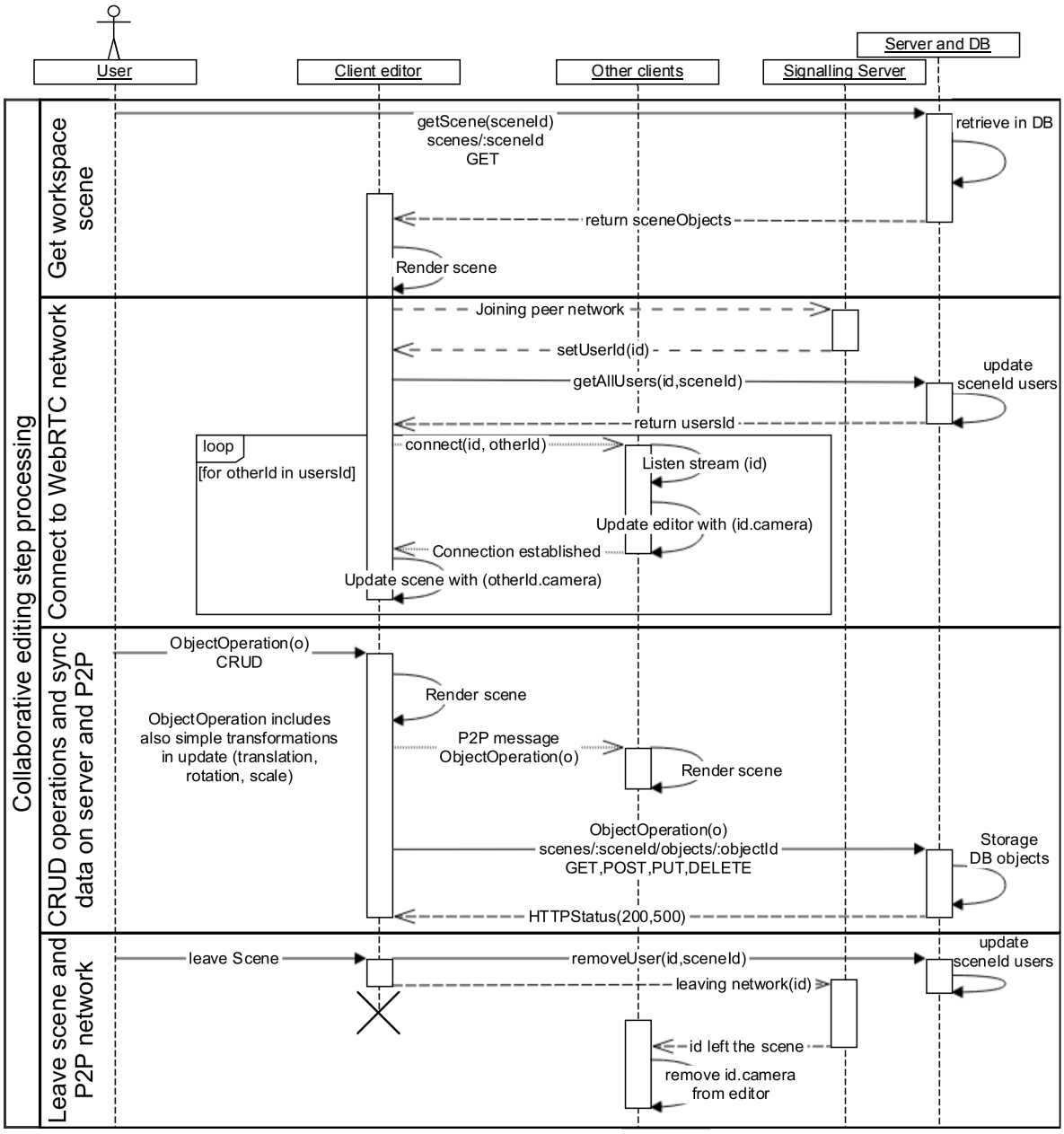


Figure 5: Sequence diagram of collaborative communication process

the environment. We use the micro-framework **ExpressJS**[9] to build the Node.JS web application.

To ensure a persistent track of the world state of the user's scene modifications, we choose the NoSQL (Not Only SQL) database **MongoDB**[10] over a conventional relational database. MongoDB is based on the *document database* technology[11] that stores and retrieves *documents* (relevant data stored together). A *document* is self-describing and can nest values in a hierarchical tree data structure. A collection is a grouping of documents, the equivalent of a relational database table. Our database contains two collections: *scenes* and *objects*. The *scenes*'s collection contains the scenes descriptions with their IDs and their metadata of the virtual workspace including a label and the connected users. This last information is crucial for the creation of the P2P mesh described in the Section 3.3. In the *objects* collection, the database stores the object with the ID of the scene it is attached to, its own ID and the full 3D object export in JSON format. The query parameters for the fundamental database operations (CRUD) on collections are fully supplied by the REST request.

[9] http://expressjs.com/
[10] http://www.mongodb.org/
[11] http://www.mongodb.com/nosql-explained

Because of the youth of WebRTC, browser's compatibility is still incomplete and some features are not yet implemented. That is why our application is only compatible with Chrome(v42+) and Firefox (v39+). **PeerJS** [MB13] is an open source library that wraps the browser's WebRTC implementation to provide a peer-to-peer connection API. The peer client, equipped with a clientID by the signaling server, can connect to a remote peer. In any case to establish a WebRTC session, a signaling protocol is needed such as WebSocket Protocol [LPR12]. We use the **PeerJSServer** implementation, based on a WebSocket server, provided by PeerJS to help broker connection between *PeerJS* clients.

Table 2: Summary of implementation choices

	Platform/Service	Library (version)
Client	WebGL Rendering	Three.JS (r69)
	Event manager	signal-js (v1.0.0)
	WebRTC	PeerJS (v0.3.9)
Server	Node.JS (v0.10.32)	ExpressJS (v4.9.0)
	WebSocket	PeerJSServer (N/A)
	NoSQL database	MongoDB (v2.6.8)

We use many different technologies in our model, as reflected in the technical choices for implementation resumed in Table 2. The modules of our implementation are communicating through APIs which is a benefit for modularity and maintenance. These choices were done with scalability perspectives.

As a result of this implementation we proposed a web platform for users where they can access to the list of the scenes' links and then the scene editor where the 3D collaboration starts.

3D Editor interface

The scene editing is done with the editor presented in Figure 6. The user can access to the list of the scenes from the menu and the "Back to scenes" link. The scene editor is titled with the scene's name, and the users connected to the scenes are shown by their ID (the bold one is the active user's). The editing part contains the tools, the viewport and the relative info. The tools are represented by the actions buttons *translate*, *rotate* and *scale* that trigger the associated helper. The *grid integer input* is for changing the resolution of the grid helper; the checkboxes: *snap* is for snapping the selected object to the grid; *local* is for changing the referential from world to local; *show grid* is for showing/hiding the grid helper. The key *s* allows to switch to the other users' cameras. The user can see the point of view shown by the camera helper representing the other's one. The viewport is where the 3D scene is represented. We can see that the user has selected the wheel of the plane and intents to translate it on the X,Y axis. The viewport info contains

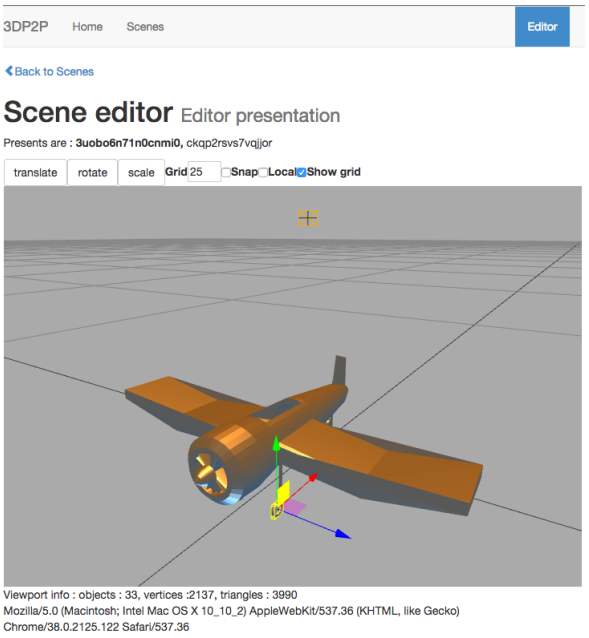


Figure 6: Editor interface

the information relative to the scene such as the number of objects (including light, cameras. . .), the number of vertices and triangles. Finally, the client information are displayed.

5 EXPERIMENTAL SETUP

We developed a prototype of the the web-based multi-user collaborative modeling to demonstrate the feasibility of our model architecture focusing on the user experience.

Table 3: Model descriptions for the experiments

Experiment	objects	size	users
Wind turbine	6	1.0 MB	2
Pick up	8	1.3 MB	4
Castle from *server*	35	1.3 MB	4
Castle from *peer*	35	1.3 MB	4

In one hand, the visualization was based on the WebGL technology using Three.JS to visualize 3D models online and offline without plugins. On the other hand, the real-time interactive collaboration relied on the hybrid architecture model exposed in the previous section. The Node.JS server platform allowed us to run a WebSocket server that handled the signaling mechanism for the WebRTC user connections creating the P2P mesh. The resources of the client were used in terms of graphics, storage, WebRTC capabilities in order to share the scene information between the users. We propose four experiments with three different models experiments (Table 3) to evaluate our system in the following criteria: user-friendly interface and the quality of the collaboration mechanisms (feedbacks, robustness and latency). A the end of each experiment, qual-

Table 4: Form distributed for each experiment and global results for 14 forms

	Questions	Answers* (results in %)			
General	Do you understand the goal?	No (0%)	**Yes (100%)**		
	Did you reach the goal?**	0 (0%)	1 (0%)	2 (14%)	**3 (86%)**
	Collaboration satisfaction ?	0 (0%)	1(14%)	**2 (72%)**	3 (14%)
	Type(s) of communication?***	None (0%)	**Oral (100%)**	Virtual (15%)	
User interface quality	3D interface expertise	0 (7%)	1 (14%)	**2 (50%)**	3 (29%)
	Tools usage	0 (0%)	1 (0%)	**2 (71%)**	3 (29%)
	Object manipulation	0 (14%)	1 (14%)	**2 (57%)**	3 (14%)
	Global quality	0 (7%)	1 (21%)	2 (64%)	3 (7%)
	The collaboration is:	Interactive (21%)	**Real-time** (79%)		
Open questions	Practice (define your practice: difficulties or frustration).				
	Collaborative rendering (define :latency, consistency, recovery)?				
	What improvements should you suggest?				

*Rates: 0 (bad), 1 (poor), 2 (good), 3 (very good). **Not asked for Castle experiments. ***Could use more than one channel.
Results have been rounded to the unit.

itative feedback was asked to the users via a form (see Table 4). For the experiments, the users were on the same local network as the server.

The *Wind Turbine* and *Pick Up* experiments had the same goal: assemble a model with multiple pieces apart. We showed a picture of the final assembly to the users, showing them the different parts of it. We distributed the pieces arbitrarily between the users and they had to import them in the viewport scene. Using the editor tools, real time information from others (application updates or any real communication), they had to manipulate the pieces (select, translate, rotate, scale) collaboratively to match the final assembly in a coherent way. The difference between the *Wind Turbine* and the *Pick Up* was the number of simultaneous users connected to the scene.

In the *Castle from **server***, the goal was slightly different: a castle kit (towers, walls, stairs...) was uploaded first on the server. At connection, the users retrieved (automatically from the server) the objects and they had about 10 minutes to creatively, but still collaboratively, build a castle. A variant, *Castle from **peer*** was introduced with the importation of another castle kit by a peer into the scene, broadcasting the new objects.

5.1 Results

Each experiment lasts about 5/10 minutes. We compiled the qualitative feedbacks of the users (see Table 4) and our observations and deductions. The users where not all very familiar with 3D interfaces but very familiar with computer: we observed mutual aid between expert users and beginners. Users were globally satisfied of the collaborative and visual results of the experiments (see Figure 7) because the goals were achieved: they succeeded in the assembly of the proposed 3D models without (too much) frustration. They even had fun on the castle experiment because they were free to create and they wanted to stay longer on the scene.

We noticed during the experiments that, on the lock system, we forgot to indicate which object was used by which user. Consequently to this lack of visual feedback, the external collaboration channel was mostly oral to exchange about object prehension: what they were doing on which piece.

The user interface was well appreciated, but maybe too simple for expert users. The manipulation of objects had a good evaluation except for the reception of a new imported model caused by the size of the message and the processing. A window frozen once during the *Castle from **peer*** on Chrome browser. The user had to quit and come back to the scene to refresh the viewport. This was a robustness test because the user appreciated to retrieve all the data and the other peers connections from the server at his/her return. The same remark was done for the fluidity of the application on the collaboration aspect.

The users did not feel latencies due to transformation operations during the experiments so they qualify the quality of the collaboration as real-time more than interactive. The variation of the number of users between experiments has not altered the rendering and networking quality of the user experience in terms of latency.

6 CONCLUSION

This paper proposed web-based 3D modeling collaboration based on a hybrid communication architecture client-server and P2P network.

The client is responsible for 3D rendering and handling the user interactions on a scene. It also hosts the peer connection to be able to communicate updates to other peers. The server is used to link the client with the NoSQL database in order to store the modifications, and manage the users presence on a scene and automatically create a P2P full mesh topology network between them. The P2P connection relies on a WebRTC communication that transmits information directly between

Figure 7: Results of 3D editor's scenes during the experiments: Wind Turbine, Pick Up and Castle kit

browser with update messages, broadcasting according to the P2P star topology, using a signaling server to establish the communication between two peers.

The qualitative evaluations of the experiments were conclusive overall even if some points should be improved. On model import, the broadcast causes latency issues on client peers receivers (camera freeze). To reduce latency with larger scenes we consider using progressive rendering and making a better use of peer-to-peer mesh to stream the model relying on seed peers like in the partial mesh topology. An improvement of interface features and visual feedback of collaborative manipulations was asked by the users. To start we have set focus on user experience.

The evaluation will be supplemented in future works with a quantitative evaluation to compare our hybrid architecture to others by selecting metrics (server logs, FPS in client, throughput, bandwidth requirements, number of connections supported...). We are now investigating experimental WebRTC tools[12] which provides statistics and graphs on the data exchanged between peers' browsers and automation tools (web automation like SeleniumHQ[13]), to evaluate on a set of scenarii the global performance and scalability of the system.

7 REFERENCES

[BBJN12] A Bergkvist, D Burnett, C Jennings, and A Narayanan. Webrtc 1.0: Real-time communication between browsers. w3c working draft. *World Wide Web Consortium*, 2012.

[Bou12] Rozenn Bouville. *Interopérabilité des environnements virtuels 3D: modèle de réconciliation des contenus et des composants logiciels*. PhD thesis, INSA de Rennes, 2012.

[cab10] Three.js - javascript 3d library, 2010.

[CCW06] Chih-Hsing Chu, Ching-Yi Cheng, and Che-Wen Wu. Applications of the web-based collaborative visualization in distributed product development. *Computers in Industry*, 57(3):272–282, 2006.

[CH14] Hung-Ming Chen and Chuan-Chien Hou. Asynchronous online collaboration in BIM generation using hybrid client-server and P2P network. *Automation in Construction*, 45:72–85, 2014.

[CSK+11] John Congote, Alvaro Segura, Luis Kabongo, Aitor Moreno, Jorge Posada, and Oscar Ruiz. Interactive visualization of volumetric data with webgl in real-time. In *Proceedings of the 16th International Conference on 3D Web Technology*, pages 137–146. ACM, 2011.

[CT07] Hung-Ming Chen and Hung-Chun Tien. Synchronous design collaboration in a peer-to-peer network. 2007.

[DBPM+14] Marco Di Benedetto, Federico Ponchio, Luigi Malomo, Marco Callieri, Matteo Dellepiane, Paolo Cignoni, and Roberto Scopigno. Web and mobile visualization for cultural heritage. In *3D Research Challenges in Cultural Heritage*, pages 18–35. Springer, 2014.

[ERB14] Alun Evans, Marco Romeo, and Arash Bahrehmand. 3D Graphics on the Web: a Survey. *Computers & Graphics*, 2014.

[Fle12] Cédric Fleury. *Modèles de conception pour la collaboration distante en environnements virtuels distribués: de l'architecture aux métaphores*. PhD thesis, INSA de Rennes, 2012.

[FM11] Ian Fette and Alexey Melnikov. The websocket protocol. 2011.

[GGCP11] Daniel Ginsburg, Stephan Gerhard, John Edgar Congote, and Rudolph Pienaar. Realtime visualization of the connectome in the browser using webgl. *Frontiers in Neuroinformatics*, 95, 2011.

[GLG11] Carl A Gutwin, Michael Lippold, and TC Graham. Real-time groupware in the browser: testing the performance of web-

[12] Chrome: `chrome://webrtc-internal`; Firefox: `about:webrtc`

[13] `http://www.seleniumhq.org/`

based networking. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, pages 167–176. ACM, 2011.

[Gri13] Ilya Grigorik. *High Performance Browser Networking: What every web developer should know about networking and web performance*. " O'Reilly Media, Inc.", 2013.

[GSWG13] Herbert Grasberger, Pourya Shirazian, Brian Wyvill, and Saul Greenberg. A data-efficient collaborative modelling method using websockets and the blobtree for over-the air networks. In *Proceedings of the 18th International Conference on 3D Web Technology*, pages 29–37. ACM, 2013.

[HLL+13] Ben Houston, Wayne Larsen, Bryan Larsen, Jack Caron, Nima Nikfetrat, Catherine Leung, Jesse Silver, Hasan Kamal-Al-Deen, Peter Callaghan, Roy Chen, et al. Clara. io: full-featured 3d content creation for the web and cloud era. In *ACM SIGGRAPH 2013 Studio Talks*, page 8. ACM, 2013.

[JBDW12] Yvonne Jung, Johannes Behr, Timm Drevensek, and Sebastian Wagner. Declarative 3d approaches for distributed web-based scientific visualization services. In *Dec3D*, 2012.

[JFM+08] Sebastien Jourdain, Julien Forest, Christophe Mouton, Bernard Nouailhas, Gerard Moniot, Franck Kolb, Sophie Chabridon, Michel Simatic, Zied Abid, and Laurent Mallet. Sharex3d, a scientific collaborative 3d viewer over http. In *Proceedings of the 13th International Symposium on 3D Web Technology*, Web3D '08, pages 35–41, New York, NY, USA, 2008. ACM.

[KVaD14] Timo Koskela, Jarkko Vatjus-anttila, and Toni Dahl. Communication architecture for a p2p-enhanced virtual environment client in a web browser. pages 1–5, 2014.

[LPR12] Salvatore Loreto and Simon Pietro Romano. Real-time communications in the web: Issues, achievements, and ongoing standardization efforts. *IEEE Internet Computing*, 16(5):68–73, 2012.

[MB13] Eric Zhang Michelle Bu. The peerjs library, 2013.

[MJ12] Charles Marion and Julien Jomier. Real-time collaborative scientific webgl visualization with websocket. In *Proceedings of the 17th international conference on 3D web technology*, pages 47–50. ACM, 2012.

[MR10] Anna Maria Manferdini and Fabio Remondino. Reality-based 3d modeling, segmentation and web-based visualization. In *Digital Heritage*, pages 110–124. Springer, 2010.

[MSG11] Christophe Mouton, Kristian Sons, and Ian Grimstead. Collaborative visualization: current systems and future trends. In *Proceedings of the 16th International Conference on 3D Web Technology*, pages 101–110. ACM, 2011.

[OJ10] Sixto Ortiz Jr. Is 3 d finally ready for the web? *Computer*, 43(1):14–16, 2010.

[Rak14] Shruti M Rakhunde. Real time data communication over full duplex network using websocket. 2014.

[Ris14] Dan Ristic. Webrtc data channels for high performance data exchange, 02 2014.

[SKR+10] Kristian Sons, Felix Klein, Dmitri Rubinstein, Sergiy Byelozyorov, and Philipp Slusallek. Xml3d: Interactive 3d graphics for the web. In *Proceedings of the 15th International Conference on Web 3D Technology*, Web3D '10, pages 175–184, New York, NY, USA, 2010. ACM.

[TV10] Stefan Tilkov and Steve Vinoski. Node.js: Using javascript to build high-performance network programs. *IEEE Internet Computing*, 14(6):0080–83, 2010.

[WV14] Max Jonas Werner and Christian Vogt. Implementation of a browser-based p2p network using webrtc. *Hamburg University of Applied Sciences, Technical Report, January*, 2014.

Online 3D Signature Verification by using Stereo Camera & Tablet

Jay Dave
Indian Institute of Technology
Dept. of Electrical Engineering
India (208016) , Kanpur
jay.dave.1988@gmail.com

KS Venkatesh
Indian Institute of Technology
Dept. of Electrical Engineering
India (208016) , Kanpur
venkats@iitk.ac.in

Garima Jain
Indian Institute of Technology
Dept. of Electrical Engineering
India (208016) , Kanpur
gari1217@gmail.com

Abstract

The signature of a person is an important biometric attribute which can be used to authenticate human identity. Conventional online approaches to signature verification only use either a single camera to track the pen tip position or a tablet to extract the dynamic features of the signature, hence the signature has only two spatial dimensions. In this paper we combine data inputs from a pressure sensitive device (tablet digitizer) and stereo vision to record signatures in 3D. Stereo vision from a pair of low cost SONY Eyecam cameras is used to track the pen tip position in x, y, & in z when the pen is off the surface as well as the pen angle with respect to the surface at all times. The digitizing tablet on the other hand, tracks x, y as well as pressure magnitude (which we denote as $-z$) when the pen contacts the surface. In all, we record the following parameters as functions of time through the duration of the signature: x, y, z, θ, ϕ, where all the linear paramaters are bipolar, with the particular case of z representing motion with positive values and pressure level with negative values. The angular values are two dimensional. The distance between the input signature's features recorded as a 5-variate parameter time sequence and the template signature's features whichwere collected during the training phase is computed using Dynamic Time Warping (DTW), and is thresholded to take a decision. While better learning techniques and more intensive experimentation will help suggest improvements, even as of the present, we have a fully working prototype of the system.

Keywords

stereo triangulation; feature vector; dynamic programming matching; stereo camera; pressure digitizing

1 INTRODUCTION

Signature verification is one of the behavioural biometrics which is commonly used to identify human beings. Signatures are very useful for identification purpose as they are very unique, especially if we consider the dynamic features of the signature in addition to the its static features.

Online signature verification techniques can be classified into two methods: function based and parameter based [yasuda2010]. In the function based approach, the features of the signatures are extracted as a function of time. For example, x position with respect to time $x(t)$, y position with respect to time $y(t)$, pressure with respect to time $p(t)$, etc. In the parameter based approach, the signature is represented as a vector of elements, each one carrying the value of the feature. The parameter based approach is further classified into global parameters (total signature time duration, number of pen ups/downs, etc.) and local parameters (speed at certain bending points, the pen direction when the signature finishes, etc.). In general, the function based approach results in better verification performance compared to the parameter based approach, but is more time consuming due to the rigorous matching procedure.

Nalwa [nalwa1997] proposed an algorithm based on the shape of the signature rather depending on the pen dynamics. The author proposed that pen dynamics have very high intra-class variability which makes the use of the features, extracted from the pen, impractical. He used global features such as aspect ratio of the signature, jitter and local features such as spatial torque, coordinates relative to the center of mass, etc. Pippin [pippin2004] proposed a new method by applying separate filters to the global features and the local features. Feng [feng2003] proposed a new warping method for verification. He used the functional approach and used

extreme points warping (EPW) for verification. EPW warps only a set of important points, and hence the time complexity is less as compared to Dynamic Time Warping (DTW).

Munich [munich2003] proposed a visual system for signature verification. An ordinary camera tracking the spatial position of the pen tip in each frame was used as an input device to the system. Dynamic Time Warping (DTW) and Continuous Dynamic Time Warping (CDTW) techniques were used to match the signature.

Kumiko Yasuda [yasuda2010] also proposed a visual system for signature verification. He used seven webcams as input devices to the system and a sequential Monte Carlo (SMC) method to track the pen tip in each frame.

Nidal S. Kamel [kamel2008] proposed a glove based signature verification method. He used the glove as an input device to the system. He proposed to use Singular Value Decomposition (SVD) as a numerical tool for matching signatures.

In this paper, we demonstrate a novel approach for signature verification, in which we use a stereo camera setup along with a pressure digitizer to make and verify signatures in 3D. A stereo camera setup gives the 3D trajectory of the pen tip position, i.e., $X, Y, +Z$ in each frame. The pressure digitizer gives pressure information when the pen is in contact with the surface during which time the pen tip has insignificant Z motion. We consider pressure information as the $-Z$ component of the 3D trajectory. By combining both, we get integrated representation for the entire signature. This is called a feature level fusion method. In this method we have also included the pen's inclination in terms of θ and ϕ to make our system more discriminative and robust.

The paper is organized as follows: Section 2 describes the system used by us for online signature verification. It includes the description of the hardware, i.e., a stereo camera setup and a pressure digitizing tablet as well as the technique of stereo calibration. Section 3 illustrates the algorithms for feature vector generation. Section 4 deals with the dynamic time warping algorithm and decision making. The experimental results are presented in Section 5. Section 6 concludes the paper with the discussion on future scope.

2 SYSTEM DESCRIPTION

The whole signature verification process is divided into two phases:

1. Database collection phase: In this phase, the database of all the users is created. We obtain multiple signatures' patterns of each user by taking their signatures at different times, since any user cannot exactly replicate his signature without variations.
2. Verification phase: In this phase, the user's signature is compared to the database. The distance is calculated between the input and the stored template signature vectors. A threshold is used to make a decision.

2.1 Stereo Camera & Calibration

A stereo camera system works on the concept of stereo vision to get the depth information of a scene. It has two or more lenses with separate image sensors for each lens. Stereo camera setup can be made by using two ordinary cameras. Both the cameras which are located at two different places, take individual images of the same scene. The 3D view of the scene is created by combining these two images. Our stereo camera setup, using two PS3 Eye cameras is shown in Fig. 1. These cameras are fixed on the aluminium sheet and the positions of their lenses are secured with an aluminium plate.

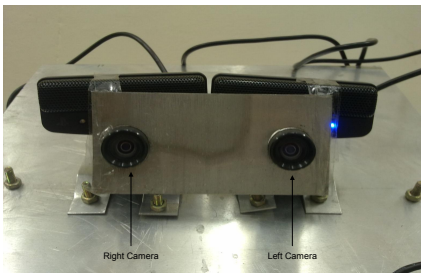


Figure 1: *Stereo Camera setup used in the Experiment*

The stereo camera calibration is the backbone of this project, as the calculation of the 3D world coordinates of the scene is on the basis of calibration. For the final product design, given sufficient standardization and precision, calibration can be restricted to the production stage. Calibration is used to obtain the "Projection matrix" and the "Distortion parameters" of the camera.

We have used the "Four-step Camera Calibration Procedure" as proposed by Heikkila & Silven [heikkila1997] for camera calibration. This method was implemented using the Camera Calibration Toolbox developed for MATLAB [matlabcameratoolbox] which is based on the OpenCV implementation.

2.2 Calibration Result

Intrinsic parameters of the left and right cameras are given in Table 1. Here focal lengths (f_x, f_y) and principle points (C_x, C_y) are given in terms of pixel units. (k_1, k_2) shows the Radial Distortions and (p_1, p_2) shows the Tangential Distortion. Extrinsic parameters, i.e., position of the right camera with respect to the left camera are: rotation vector $(\theta_x, \theta_y, \theta_z) = (-0.01405, 0.19919, -0.11265)$ and translation vector $T = (T_x, T_y, T_z) = (-77.55565, 7.00082, 16.28213))$ (in mm units). Rotation matrix R can be found by using rotation vector.

 ISBN 978-80-86943-65-7

Parameters	Left Camera	Right Camera
(f_x, f_y)	$(755.2, 754.33)$	$(765, 765)$
(C_x, C_y)	$(313, 267)$	$(320, 236)$
(k_1, k_2)	$(-0.081, 0.278)$	$(-0.079, 0.222)$
(p_1, p_2)	$(0.006, 0.001)$	$(-0.003, -0.001)$

Table 1: Intrinsic parameters of Left and Right Cameras

2.3 Pressure Digitizing Tablet

The tablet [geniustablet] used as a pressure digitizing device is shown in Fig. 2. The tablet measures the pressure of the pen tip on the scale of 10 bits, i.e., the maximum pressure level is 1023. The tablet also measures the trajectory of the pen tip while writing on it.

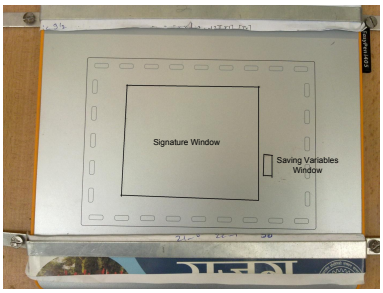


Figure 2: *Pressure Digitizing Tablet*

We have made two windows on the tablet surface for user's ease.

1. Signature window: Users have to sign in this window only. This window size is application dependent.

2. Saving Variables window: The user has to press this window once the signing is complete, to save all the features provided by the tablet.

3 FEATURE VECTOR GENERATION

3.1 Pen tip Detection and Tracking Algorithm

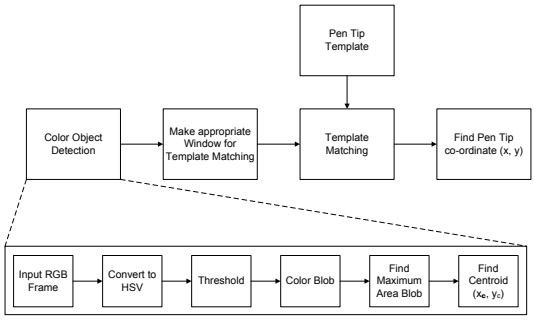


Figure 3: *Block Diagram of Pen tip Detection algorithm*

Fig. 3 shows the block diagram of the pen tip detection algorithm. We have used the template matching method for detecting exact pen tip location followed by color object detection as shown in Fig. 4. The centroid of the blob is found using [bradski1998].

$$x_c = \frac{M_{10}}{M_{00}}; \quad y_c = \frac{M_{01}}{M_{00}}$$

where $M_{00} = \sum_x \sum_y I(x,y)$ is the zeroth moment. $M_{10} = \sum_x \sum_y xI(x,y)$ and $M_{01} = \sum_x \sum_y yI(x,y)$ are the first moment.

3.2 Stereo Triangulation

The 3D coordinates of the object are calculated by using stereo triangulation. This is also known as 3D recovery [hillman2005]. In order to get the 3D coordinates of the object, we have to back project the line of the pixel in the left camera and the right camera as shown in Fig. 5. In this way we apply the inverse projection matrix to get from the 2D image point to the 3D line. These lines are the 3D back projection lines which usually meet at exactly one point.

For simplicity, we have taken left camera coordinate system as the world coordinate system as shown in Fig. 5. Hence, the left camera center becomes the world's origin $(0,0,0)$ and all the three axes of the left camera becomes the axes of world coordinate system respectively. Now the relation between the right camera coordinate system and the left camera coordinate system is:

$$C_r = RC_l + T$$

where $C_l = (X_l, Y_l, Z_l)$ and $C_r = (X_r, Y_r, Z_r)$ are object point location with respect to the left and the right camera respectively, R = Rotation Matrix which shows the rotation between the right camera coordinate system and the left camera coordinate system, and $T = \begin{bmatrix} T_x & T_y & T_z \end{bmatrix}'$ = Translation Matrix which shows the translation between the left camera coordinate system and the right camera coordinate system.

We can write the above equation as:

$$\begin{bmatrix} X_r \\ Y_r \\ Z_r \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_l \\ Y_l \\ Z_l \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (1)$$

Now from the Inverse Perspective Transformation:

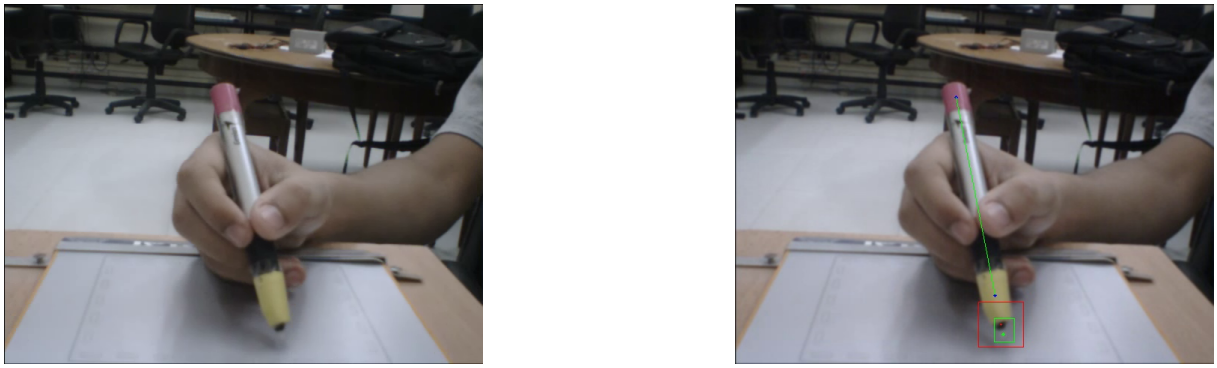
$$X_l = \frac{x_l' * Z_l}{f_{x_l}}, \quad Y_l = \frac{y_l' * Z_l}{f_{y_l}} \quad (2)$$

where $x_l' = x_l - C_{x_l}, \quad y_l' = y_l - C_{y_l}$

$$\text{and} \quad X_r = \frac{x_r' * Z_r}{f_{x_r}}, \quad Y_r = \frac{y_r' * Z_2}{f_{y_r}} \quad (3)$$

where $x_r' = x_r - C_{x_r}, \quad y_r' = y_r - C_{y_r}$, (f_{x_l}, f_{y_l}) and (f_{x_r}, f_{y_r}) are the focal lengths of the left and right camera in the x and y direction respectively.

 ISBN 978-80-86943-65-7



(i) *Input BGR Frame* **(ii)** *Detected Pen tip point as a green point*

Figure 4: *Pen tip detection*

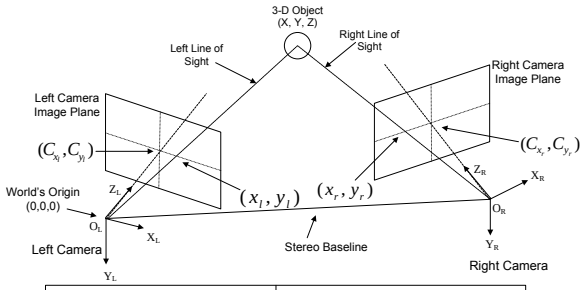


Figure 5: *Reconstruction of 3D Point in Space*

3.3 Tablet Feature Vector

The tablet gives the pressure information p along with the x and y position of pen tip at the rate of 100 Sa/s. We include time-stamp with these features. We use the system time for the time-stamp. Hence, the system records $\{x(i), y(i), p(i)\}$ feature values approximately at each time $t = 10ms$, and the tablet feature vector becomes $F_{tab} = \{t_{tab}(i), x(i), y(i), p(i)\}$.

The 2D reconstruction of the signature using tablet features is shown in Fig. 6.

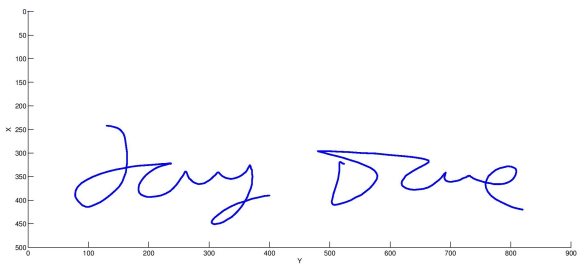


Figure 6: *2D reconstruction of signature using tablet features* $\{x(i), y(i)\}$

3.4 Camera Feature Vector

The stereo camera setup gives the continuous 3D trajectory of the pen tip location by applying stereo triangulation. We have added two trackers to get two 3D positions, where the first corresponds to the pen tip position $\{X_t, Y_t, Z_t\}$ and the second corresponds to the pink marker $\{X_h, Y_h, Z_h\}$ which sticks on the head of the pen. Hence we get the two end points of the pen. We can find the orientation of the pen by using these two 3D points with respect to the tablet surface. Pen orientation can be calculated as:

$$\theta = cos^{-1}\left(\frac{Z_h - Z_t}{r}\right); \qquad \phi = tan^{-1}\left(\frac{Y_h - Y_t}{X_h - X_t}\right)$$

where $r = \sqrt{(X_h - X_t)^2 + (Y_h - Y_t)^2 + (Z_h - Z_t)^2}$

We have also added the time stamp along with these features. The PS3 Eye camera works at 60 fps. Hence the system records camera feature values at approximately every $t = 17$ ms. The final camera feature vector becomes $F_{cam} = \{t_{cam}(i), X_t(i), Y_t(i), Z_t(i), \theta(i), \phi(i)\}$.

The 3D reconstruction of the signature using camera features is shown in Fig. 7.

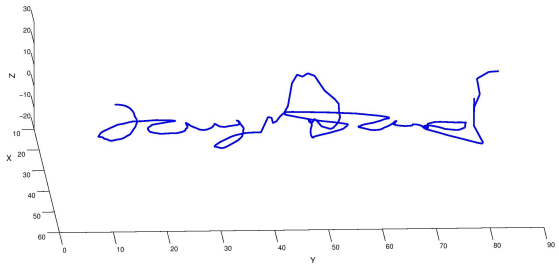


Figure 7: *3D reconstruction of signature using the camera features*

There are significant differences between feature values of the two feature vectors on account of the differences between the respective sensors' ranges and resolutions. This necessitates feature normalization. The main objective of a feature normalization process is to

 ISBN 978-80-86943-65-7

modify the mean and variance of the feature values by applying suitable transformation functions (max, min-max, median, z-Score, etc.). The normalization process maps the feature values of different feature vectors into a common domain. In our algorithm, we use Min-Max normalization for normalizing signature features from the two sensors. which is defined as:

$$a_i' = \frac{a_i - \text{Min}}{\text{Max} - \text{Min}}$$

where, a_i and a_i' denote the i^{th} feature value before and after normalization process respectively, $\text{Max} = \max_i\{a_i\}$, finds the maximum value, and $\text{Min} = \min_i\{a_i\}$, finds the minimum value.

3.5 Synchronization

In our experiment, we obtain the tablet feature vector at 100 Sa/s and stereo camera feature vector at 60 fps. To combine these feature vectors, we normalize each of them by using Min-Max normalization technique, and then use linear interpolation for the purpose of synchronization. After the synchronization, time stamps for both the feature vectors will become identical, i.e., $t(i) = t_{tab}(i) = t_{cam}(i)$.

After normalization and synchronization, we can combine the two different feature vectors. This combining process is known as feature level fusion. We have used feature level fusion for combining F_{cam} and F_{tab}. We have made one common feature vector which contains $\{t(i), x(i), y(i), p(i), \theta(i), \phi(i)\}$ when the pen tip touches the tablet surface and $\{t(i), X_l(i), Y_l(i), Z_l(i), \theta(i), \phi(i)\}$ when the pen tip is poised above the surface without contact. Hence, our final feature vector after feature level fusion becomes $F_{Final} = \{t(i), X(i), Y(i), Z(i), \theta(i), \phi(i)\}$.

The 3D reconstruction of the features generated from the feature fusion method is shown in Fig. 8, where blue is used under the pen down condition and red color under pen up. All the features are shown as a function of time in Fig. 9.

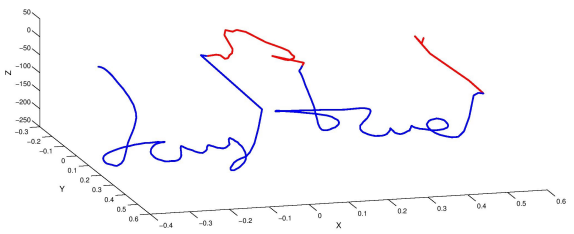


Figure 8: *Reconstruction of 3D signature made by Feature Fusion method*

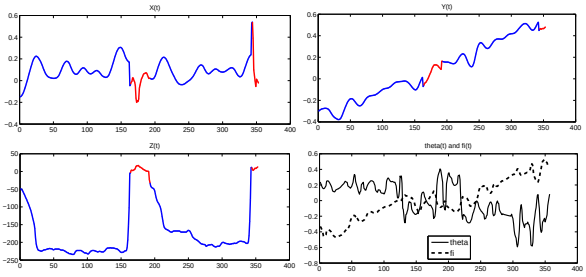


Figure 9: *Final features made by feature fusion as a function of time*

4 SIGNATURE VERIFICATION ALGORITHM

4.1 Dynamic Programming Matching (DPM)

We can summarize DPM by following way:

1. Initialization :

$$dist(1,1) = 0; \qquad \xi(1,1) = (1,1)$$

where, $dist(x,y)$ = total distance upto (x,y) point and $\xi((n_x,n_y),(n_{x'},n_{y'}))$ shows the warping path between nodes (n_x,n_y) and $(n_{x'},n_{y'})$.

2. Recursion : for $1 \le i \le N_a, 1 \le j \le N_b$, such that i and j must follow the monotonicity constraint,

$$dist(i,j) = min \begin{cases} dist(i-1,j) + d((i-1,j),(i,j)) \\ dist(i-1,j-1) + \\ \quad d((i-1,j-1),(i,j)) \\ dist(i,j-1) + d((i,j-1),(i,j)) \end{cases}$$

$$\xi(i,j) = argmin \begin{cases} dist(i-1,j) + d((i-1,j),(i,j)) \\ dist(i-1,j-1) + \\ \quad d((i-1,j-1),(i,j)) \\ dist(i,j-1) + d((i,j-1),(i,j)) \end{cases}$$

where, $d(,)$ is the Euclidean norm function.

3. Termination :

$$Dist(S_1,S_2) = dist(N_a,N_b); \qquad \theta_1 = (N_a,N_b)$$

Here $Dist(S_1,S_2)$ represents the final distance between two signals.

4.2 Signature Verification using DPM

It is practically impossible for a user to produce exactly the same signature every time. Hence we can not compare two signatures and find the distance between them by using a simple Euclidean distance formula. DPM (Dynamic Programming Matching) [DPBellman] is a method that finds correspondences between two signatures. It takes each sample in the 1^{st} signature and finds

the closest sample in the 2^{nd} signature using a predefined metric. Given this similarity, it is possible to estimate a distance between the two signatures.

The warping function for two signatures is not linear. Here the signature varies with time, hence DPM is called Dynamic Time Warping (DTW).

Fig. 10 shows one example of signature verification process. Two different signatures of the same user is shown in Fig. 10i. 1^{st} signature is shown by red line and 2^{nd} signature is shown by blue line. Only min-max normalized x position $x_i(t)$ and min-max normalized y position $y_i(t)$ of the signatures are considered here as a feature vector (see Fig. 10ii and Fig. 10iii). Then DTW algorithm is applied to calculate the distance between $x_1(t)$ & $x_2(t)$ and $y_1(t)$ & $y_2(t)$. After applying DTW, we get warped $x(t)$ and $y(t)$ sequences as shown in Fig. 10iv and Fig. 10v respectively. The optimal correspondence paths for both the sequences are also shown in Fig. 10vi and Fig. 10vii. The distance between two x sequences is $Dist_x = 0.0437$ and between two y sequences is $Dist_y = 0.1834$. And the overall distance is $Dist_{overall} = 0.1885$ which is calculated by using equation (4). In general if we have N different sequences, the overall distance is calculated by:

$$Dist_{overall} = \sqrt{\sum_{i=1}^{N} Dist_i^2} \tag{4}$$

4.3 Threshold Selection

Threshold value selection is a very critical task in the verification process as it affects the classification accuracy. A high threshold value increases the FAR (False Acceptance Rate) of the system and a low threshold value increases the FRR (False Rejection Rate). As explained in [jain2002], we can choose either a global threshold for all the users or a user dependent threshold.

We have used a global threshold based method to find the threshold value in our algorithm. The global threshold can be calculated as :

$$\text{Threshold} = \frac{\sum_{i \neq j, i<j} \text{dist}(S_r^i, S_r^j)}{\frac{N(N-1)}{2}} \times \gamma$$

where, γ is the adjustment factor, N is the total number of reference signatures, S_r^i is the i^{th} reference signature, $Dist(:,:)$ function finds the distance between two signatures.

In general, if we have N different sequences, the overall distance is calculated by:

$$dist_{overall} = \sqrt{\sum_{i=1}^{N} dist_i^2}$$

4.4 Decision Making

In this step we identify the user as fraud or genuine. The distance is calculated between user's unknown signature S_u and all the reference signatures S_r^i of that user. If the calculated distance is lower than the threshold value, then the signature is a genuine otherwise it is a forged one. Alternate formulations of the decision criterion are possible.

$$\begin{cases} S_u = \text{genuine if} & \frac{1}{N}\sum_{i=1}^{N} dist(S_r^i, S_u) < \text{Threshold} \\ S_u = \text{fraud if} & \frac{1}{N}\sum_{i=1}^{N} dist(S_r^i, S_u) \geq \text{Threshold} \end{cases}$$

5 EXPERIMENTAL RESULTS

We have used the setup shown in Fig. 11. The stereo camera setup is mounted on an aluminium sheet, which is fixed on the wooden plank at about 45° angle, to enable the cameras to see the tablet surface and the head of the pen. The stereo camera and the tablet are fixed rigidly to maintain calibration.

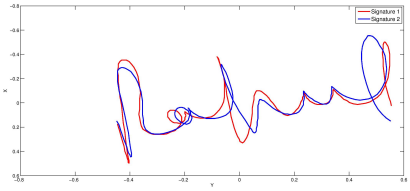
We have collected signatures from 8 users. Table 2 shows the number of signatures used for database collection, verification and forgery.

We compute Equal Error Rate (EER) of each feature individually, separately for the stereo camera setup as well as the pressure digitizing tablet, to evaluate the accuracy of each feature. We also compute EER of each feature individually of the feature level fusion method. The value of γ is set to be 1.5.

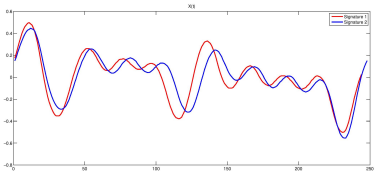
- Camera Features : Table 3 shows the FAR, FRR and EER values of the camera features. Fig. 12 shows the error trade off curve of camera features.
- Tablet Features : Table 5 shows the FAR, FRR and EER values of the tablet features. Fig. 13 shows the error trade off curve of tablet features.
- Feature level Fusion method : Table 4 shows the FAR, FRR and EER values of the all the features calculated using the feature level fusion method. Fig. 14 shows the FAR and the FRR curves and Fig. 15 shows the error trade off curve for feature level fusion method.

6 CONCLUSIONS AND FUTURE WORKS

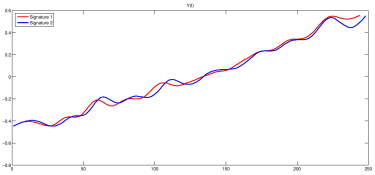
We have presented a novel low-cost approach to online signature verification method, built with off the shelf components such as a pressure digitizing tablet and a stereo camera pair. The tablet gives the pressure information as well as the pen tip position. The stereo camera setup gives the 3D trajectories of the pen tip.



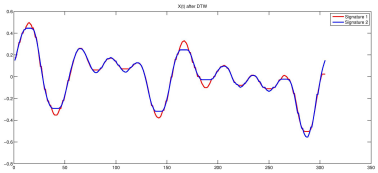
(i) *Two different signatures of the same user*



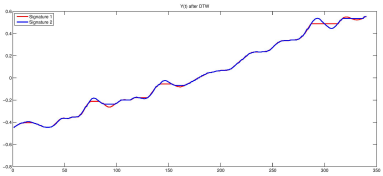
(ii) $x(t)$ *of two signatures*



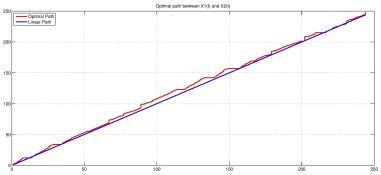
(iii) $y(t)$ *of two signatures*



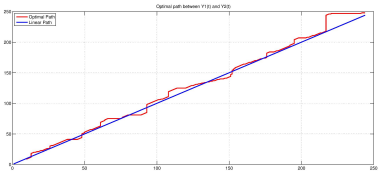
(iv) $x(t)$ *of two signatures after applying DTW*



(v) $y(t)$ *of two signatures after applying DTW*



(vi) *Optimal correspondence between* $x_1(t)$ *and* $x_2(t)$



(vii) *Optimal correspondence between* $y_1(t)$ *and* $y_2(t)$

Figure 10: *Matching of signatures parameters using Dynamic Time Warping (DTW)*

	Total Signatures per User	Total Signatures
For Database collection	10	80
For Verification purpose	15	120
Fraud Signature	6	48

Table 2: Total number of signatures

Features	X	Y	Z	θ	ϕ	*Overall*
FRR	25	20.83	20.83	12.5	20	14.17
FAR	50	41.67	52	43.75	31.25	31.25
EER	36.27	33.45	40.93	28.62	22.39	20.54

Table 3: Evaluation of Camera Features

Features	X	Y	Z	θ	ϕ	*Overall*
FRR	20	20	14.17	12.5	20	8.33
FAR	14.58	18.75	20.83	43.75	31.25	8.33
EER	14.58	19.58	16.55	28.62	22.39	8.33

Table 4: Evaluation of Features of feature level fusion method

Features	X	Y	P	*Overall*
FRR	21.67	30.83	30.83	21.67
FAR	6.25	6.25	16.67	4.17
EER	15.41	16.46	20.83	12.5

Table 5: Evaluation of Tablet Features

In our analysis, we found EER = 20.54% for the camera features and EER = 12.5% for the tablet features. For the proposed feature level fusion method, FAR = 8.33%, FRR = 8.33% and EER = 8.33%. These figures are likely to improve considerably with a better finished prototype.

A larger database can significantly decrease the FAR error rate. Pen tip tracking was done by color blob detection followed by template matching. Color blob detection is affected by the background color, and hence we should try and make the pen tip tracking independent of color blob detection. Accurate techniques like Kalman filter for pen tip detection and tracking can increase the precision and accuracy of the system. We considered only local features of the signatures for verification and

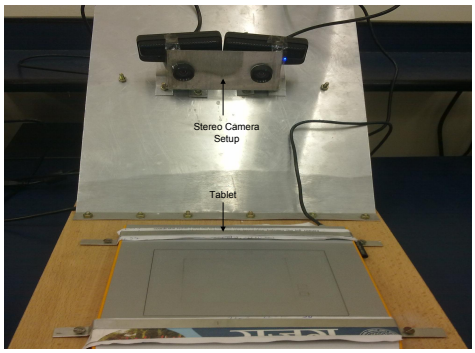


Figure 11: *Experimental setup*

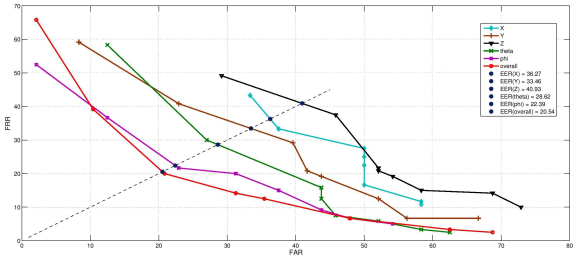


Figure 12: *Error Trade off Curve of Camera features*

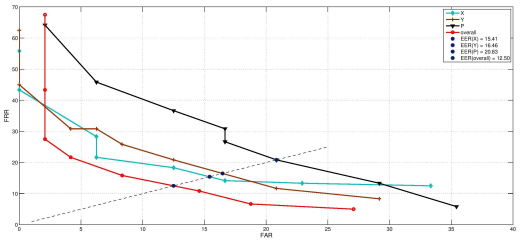


Figure 13: *Error Trade off Curve of Tablet features*

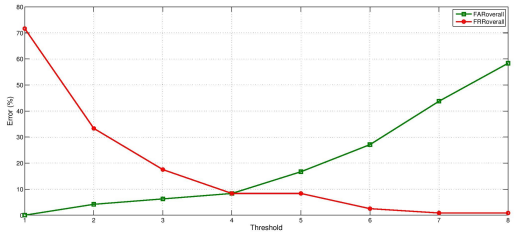


Figure 14: *FAR and FRR curves of the Feature Fusion method*

evaluation. We can additionally use global features like velocity, acceleration, etc., to make the system more reliable and robust. The use of a user dependent threshold can also result in lower error rates.

7 REFERENCES

[yasuda2010] Yasuda, K, Daigo M, Satoshi S, and Takashi M. Visual-based online signature verification using features extracted from video. Journal of Network and Computer Applications 33, no. 3 (2010): 333-341.

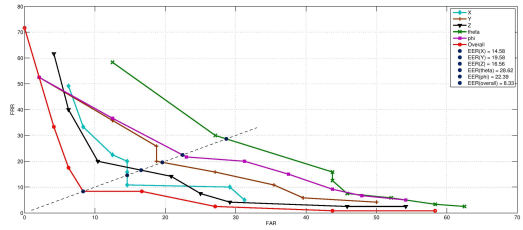


Figure 15: *Error Trade off Curve of the feature fusion method*

[nalwa1997] Nalwa, Vishvjit S. Automatic on-line signature verification. Proceedings of the IEEE 85, no. 2 (1997): 215-239.

[pippin2004] Pippin C.E., Dynamic signature verification using local and global features, *Georgia Inst. Inform. Technol, Atlanta, Technical report*, 2004.

[feng2003] Feng, Hao, and Chan Choong Wah. Online signature verification using a new extreme points warping technique. Pattern Recognition Letters 24, no. 16 (2003): 2943-2951.

[munich2003] Munich, Mario E., and Pietro Perona. Visual identification by signature tracking. Pattern Analysis and Machine Intelligence, IEEE Transactions on 25, no. 2 (2003): 200-217.

[kamel2008] Kamel, Nidal S., et al. Glove-based approach to online signature verification. Pattern Analysis and Machine Intelligence, IEEE Transactions on 30, no. 6 (2008): 1109-1113.

[heikkila1997] Heikkila, Janne, and Olli S. A four-step camera calibration procedure with implicit image correction. In Computer Vision and Pattern Recognition. Proceedings., IEEE Computer Society Conference on, pp. 1106-1112. IEEE, 1997.

[matlabcameratoolbox] J. Y. Bouguet, Camera calibration toolbox for Matlab, 2008.

[geniustablet] http://www.geniusnet.com/ Genius tablet i405

[DPBellman] R. Bellman, Dynamic Programming, Princeton University Press, 1957.

[jain2002] Jain, Anil K., Friederike D. Griess, and Scott D. Connell. On-line signature verification. Pattern recognition 35, no. 12 (2002): 2963-2972.

[hillman2005] Hillman, Peter. White paper: Camera calibration and stereo vision. Lochrin Terrace, Edinburgh EH3 9QL, Tech. Rep (2005).

[bradski1998] Bradski, Gary R. Computer vision face tracking for use in a perceptual user interface. (1998).

Vision and Virtual-based Human Computer Interaction Applications for a New Digital Media Visualization

Chutisant Kerdvibulvech

Graduate School of Communication Arts and Management Innovation
National Institute of Development Administration
118 SeriThai Rd., Klong-chan, Bangkapi, Bangkok 10240, Thailand

chutisant.ker@nida.ac.th

ABSTRACT

With the rise of smartphones and tablets interactively, human computer interaction is a very popular topic for engineers, artists, designers and computer scientists around the world in both industry and academia. This topic was studied and researched over many years ago. Nevertheless, most of previous works were studied separately between communication arts (e.g., advertising and marketing communication research) and computer science. Indeed, there has been little work giving an overview of recent integrated research of digital media and some new technologies, such as computer vision, virtual reality, and human computer interaction for visual communication. Therefore, our contribution of this paper is to discuss the recent state-of-the-art development of the digital media research work using and applying these aforementioned multimedia-based technologies. A literature review of the novel digital media and interactive augmented reality researches is also discussed. More importantly, this paper also provides a work-in-progress framework for future digital media research when applying graphical visualization, human computer interaction such as haptic, and sensor technologies into every traditional sense of human interactively, from vision to touch and from smell to taste. In general, this paper will be beneficial for any related field of interactive multimedia, communication arts and human computer interaction both industrial and educational aspects and also for any related researcher such as computer science art communicator.

Keywords

Human Computer Interaction, Robotic, Communication Arts, Taste Communication, Touch Communication, Smell Communication, Multimedia, Digital Media, Graphical Visualization, Advertising, Marketing, Visual Communication

1. INTRODUCTION

With the rise of advanced technologies in computer science such as human computer interaction and graphical visualization, the way people send information and communicate has gradually transformed in this 21st century. Since the emerging of the computer and social media revolution, the lure of the virtual communication has attracted engineers, artists, designers, communicators, computer scientists, and others with the great idea that we might someday accomplish with technologies both physically and virtually [Bai12a]. On one hand, one of the fundamental purposes of communication research is to allow two people to be in two different locations at the same time (or nearly the same time), but can communicate human-to-human smoothly and robustly in every sense of human as face-to-face interaction. For this reason, digital media expand increasingly and scientifically horizons into new sensory modalities every day. On the other hand, dealing with problems for communication perfectly for human-to-human remotely is not a trivial task. Communications in different senses of human pose different requirements of the communication challenging. The main challenge is how to reproduce the perception realistically and robustly for every five sense: sight, audition, touch, smell and taste. Even though some communication-based systems with different capabilities have been built, most of previous works were developed separately between communication arts and computer science. Over decades, studies of communication media by communication researchers were usually achieved quite separately. At the same time, studies of communication for digital media by computer scientists were studied in some limited dimensions. Rice from Department of Communication in University of California, Santa Barbara, and

Leonardi from Department of Communication Studies in Northwestern University, give a good overview [Ric12a] of research on the merging relationship between technology and communication. It now obviously makes sense to merge and integrate the two related-theoretical fields.

This paper discusses a preliminary framework for innovative technologies applying in the digital media for every sense of human. The first one is visual communication using augmented reality (AR) and computer vision (CV). Second, we discuss recent technologies used for touch communication. The third one is for smell communication technologically and chemically. The last communication we mention is taste.

For visual communication, one of the very effective tools used is augmented reality. In fact, augmented reality has, in recent decades, increasingly become an extremely popular field in computer science. It is recently utilized in wider fields, including digital media and other fields of communication arts such as marketing and advertising. As defined by many technologists, augmented reality is a technology combining the physical world and virtual media (graphical aid information) together.

Figure 1. A Mad Tea-Party augmented reality experience using media theory was designed by [Mac01a] from Georgia Institute of Technology.

One of very first works for applying augmented reality to conventional media theory in the 21st century is, in [Mac01a], implemented by a research group from Georgia Institute of Technology. They discussed the development of augmented reality as a new medium for digital media interactively. They showed two creative examples of augmented reality experiences (i.e., A Mad Tea Party and Ghosts of Sweet Auburn) using their conceptual frameworks from media studies. Figure 1 shows A Mad Tea Party augmented reality experience. They attempted to leverage the information exchange between the virtual world and the real world. It is just as the same as the concept of augmented reality and mixed reality, but for remediating traditional media, e.g. stage, film, and compact disc. After that, they built on interactively the cultural expectations of users. More recently, augmented reality is similarly used for a real-time mobile-based film scholarship as presented by Chao et al. [Cha14a]. The application gives a kinetic and aural experience for users. It lets them to conveniently access augmented multimedia content using this augmented reality technology. Also, augmented reality is recently used in the digital marketing and advertising arena. It can be seen in many recent augmented reality research works. To begin with, several concepts for augmented reality digital advertising for paper-based leaflets for attempting to bridge the digital divide were investigated in [Loc13a]. The first prototype explained by Lochtefeld et al. is the Guerrilla marketing approach (called GuerrillAR), while the second one (called PageAR) is the approach using visualizing cross-selling recommendations. Next, [Sin14a] studied reasonably why augmented reality can make an effective choice for the marketers utilized in the advertising campaigns. We, in this paper, review this augmented reality technology that is now more and more linked to digital media.

Nonetheless, in many augmented reality researches, it usually deals with only vision, specifically computer vision, often abbreviated briefly with the acronym CV. We believe that digital media are soon deemed to go beyond vision. Martin described in [Mar13a] about the future of augmented reality when extending it technologically to other senses (i.e., hearing, touch, smell, and taste), rather than just sight. However, due to many technical reasons, over past decades the sense of touch has uniquely been difficult for people's communication in term of digital media. Right now, it is true that it is more common and easier to use smartphone to connect people to people by talking, messaging and even FaceTime. But it is not obviously and widely yet for the sense of touch. This paper also studies about this possibility. Some case studies of huggable internet from Cheok's work [Che13a] and the therapeutic robot for affective touch from Massachusetts Institute of Technology (MIT) Media Lab's work [Sti06a] will be discussed and mentioned. In addition, the possibility of communication for digital media by the senses of smell and taste will be reviewed in this paper. It is true that dealing with the sense of smell and taste is obviously extremely difficult since they often require some dangerous chemical substances to interact. However, there are still some recent researches trying to achieve these goals. We review some new technologies used for the communications of smell and taste, such as [Kay08a] [Ran14a] and [Saa14a]. We define the work-in-progress framework for future digital media when applying computer vision, graphical visualization, human computer interaction such as haptic, and sensor technologies into every traditional sense of human interactively,

from vision to touch and from smell to taste. We coin this framework as 'new digital media' in this paper. In fact, this term was used once in [Jam09a] by James from Harvard University, but in different meaning. In addition, according to the very similar definition between new media and digital media, it is acceptable to call the term 'new new media' too, even though 'new digital media' is more preferable. This paper is organized as follows. We will talk about the new digital media in section 2. In section 3, we will focus on sense of sight. The next section will present technologies used in touch communication. Smell communication will be explained in section 5. Section 6 will propose technologies for taste communication. In section 7, we conclude the paper and give some possible future works.

2. NEW DIGITAL MEDIA

Digital media are defined quite widely, even though they should frequently be encoded in a computer-readable format. In today's world, the new media include increasingly countless examples, such as digital audio, e-books, databases (CD, DVD and hard drive), digital imagery and video, and web pages. Social media websites such as Facebook, YouTube and Twitter are also included in its definition. Since the smartphone revolution takes the next big leap in recent years, digital media trends continue obviously to drive innovation. Newman [New15a] from Reuters Institute predicted and explained interestingly that the new media have become one of the defining factors of society; from the political election and the country's economic power to the social ideology and the technology we use every day. In other words, people, especially young people, in 2015 have increasingly engaged more and more through digital media. For instance, social media websites via smartphones, wearable devices, or even advanced drones are included.

In a multidisciplinary context, digital media from the fields of traditional communication arts and humanities are merging with many new scientific technologies more and more every day. It is crystal clear that this integration is so beneficial for the world's innovation. In this paper, we present a work-in-progress framework for the new digital media research when applying augmented reality and other computer science technologies to communicate human-to-human using all five senses of human. We call this framework for communication of every human sense as 'new digital media'. Figure 2 represents this framework of new digital media we defined. The technologies used and applied for the new digital media can expand widely while implementing and integrating, from graphical visualization to human computer interaction, from augmented reality to computer vision, and from haptic to sensor. However, it is important to note that this paper will not focus on technologies used for hearing communication. This is because it has been widely used for long time ago. Thus, we start with visual communication in the following section.

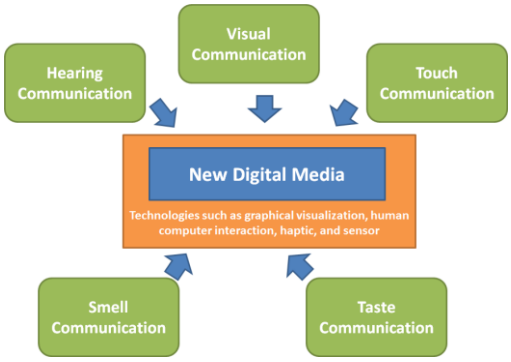


Figure 2. The framework of New Digital Media we defined in this paper

3. VISUAL COMMUNICATION

Augmented reality is a view of a natural environment (real scene) whose elements are merged by graphical digital information and media. [Wan13a] provides a good survey of the augmented reality studies in built environment in the last decade. Over time, augmented reality has been touted as an important technology of the future but has been utilized mainly in only high-end, advance and novelty settings. However, this technology [Kre10a] now intends to move from only inside research laboratories into other related-areas, such as consumer markets. According to the valuable surveys [Sin14a] and [Kim14a], it is observed obviously that augmented reality has become one of the emerging tools for another arena of digital media, games and communication arts (e.g., advertising and marketing communication research). In fact, there are many interesting research works towards augmented reality in advertising strategies [Vau09a]. For instance, Lochtefeld et al. [Loc13a] built two prototype systems of augmented reality-based advertising concepts for paper leaflets. The first system, called GuerrillAR, overlays a competing retailer's paper leaflet with one's own content in a real environment. By using this prototype, it is more convenient for a retailer to check and compare both the price and the quality of the products. At the same time, the smartphone can be used to point to leaflet of the competitor. Then, he/she can see a personal augmented overlay on the smartphone. The second system is called PageAR. It aims for cross-selling suggestions between different products inside the leaflet for advertising purpose.

In addition, augmented reality use is not just only found in advertising systems, but also in marketing

prototypes. This statement is believed to be true because we find technological marketers use this technology to promote their products online. It is deemed to change conservative marketing strategies enormously. A study of Singh and Pandey's work [Sin14a] has reaffirmed once again that augmented reality is a valuable choice in their marketing strategies for the marketers. It is noted in the study that consumers do not prefer passive monologues which are designed by the marketers anymore. They prefer to be interactively a part of the conversation with the brand to make a purchase final decision. To make the marketing strategies more interactively, augmented reality has the potential to deal with this issue, especially for new age consumer engagement.

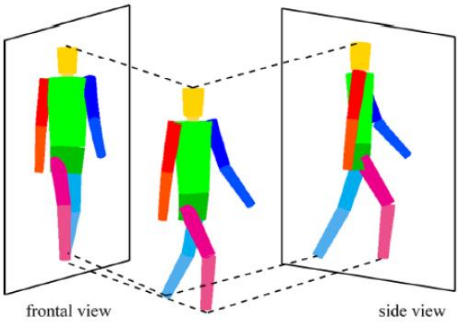


Figure 3. The virtual front and side views for 3D structural human shape analysis using computer vision was presented by [Ker14a].

It is not only augmented reality, but also computer vision that has played an important role for visual communication. Briefly speaking, computer vision is a branch of computer science for study of acquiring, analyzing, understanding and interpreting images from the physical world. For years, it has been touted as a great technology in related fields [Rus14a], such as robotics, neurobiology, solid-state physics, pattern recognition, and artificial intelligence. For example, it is used for a method for vision-based 3D structural human shape analysis by Kerdvibulvech and Yamauchi [Ker14a] to model and recognize from gait signatures, as represented in Figure 3 for the virtual front and side views for human shape analysis.

4. TOUCH COMMUNICATION

As stated earlier, communication is currently drifting away from each individual sense of human. Digital media are able to extend technologically to other particular senses. Those senses are touch, smell, and taste. They are also quite important means of communication. One of very essential senses is touch. Touch is, in fact, an obviously vital sense, as it can convey so many kinds of communicative intents. Touch can promote trust, a foundational element of interpersonal relationships, and cooperation of human, so that it is a great enabler of trust between group members, as explained in [Kra10a] by Kraus et al. from University of California, Berkeley. There is also a study of the communication of emotion via touch [Mat09a] reviewed that the sense of touch can affect emotion directly such as anger, disgust, love, fear, sympathy, and gratitude. Although it is recently common to connect people to people by talking and imaging via smartphone or any particular device, it is still not a trivial task for the sense of touch. To deal with this, Teh and Cheok [Teh08a] built a prototype system for pet, called Pet Internet, to allow human to touch his/her pet distantly by touching an interface. In fact, more recently, a similar research for pet is also conducted by Murata et al. [Mur14a] for remote haptic communication interactively. According to their pilot study, this interactive system makes human and his/her pet to communicate and feel closer to one another via haptic responses by exchange of haptic feedback.

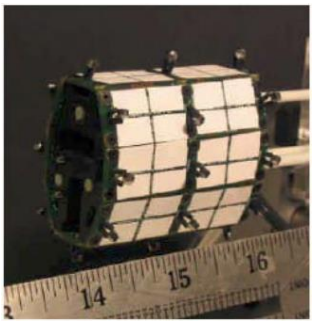


Figure 4. The arm section of Teddy Bear, used in Huggable, was built by Stiehl et al. [Sti06a] from MIT.

Rather than just the aforementioned systems of human-animal interaction, the topic of human-human interaction by touching is studied by other leading computer scientists. For example, [Che13a] presented a parent and child hugging communication wearable system, called Huggy Pajama (later called T Jacket commercially). It can allow children to hug their loved one from a remote distance by reproducing a hug between two people, focusing on a parent and child, but not specifically. This Huggy Pajama system was also featured in NHK's (Japan Broadcasting Corporation) TV program in Japanese language [Teh08a] to show its demonstration. In addition, an MIT research group, led by Stiehl [Sti06a], presented a therapeutic robotic companion (i.e., Teddy Bear) using touch interactions relationally, called similarly Huggable. It is a robotic companion for affective and relational touch-based interactions with a human. This system uses a large number of somatic sensors over the whole surface of the Huggable, as shown in Figure 4 for the arm

section of the robot. It uses a combination of electric field, temperature, and QTC (Quantum Tunneling Composite) force sensors to send affective and relational touch-based interactions.

Figure 5. A robotic platform for socially embodied telepresence was presented by [Ada10a].

A similar idea was implemented in [Ada10a] for a socially expressive telerobotic, called MeBot, as shown in Figure 5. They used the assumption that they want to build a system that is able to communicate more than just video or audio, but expressive gestures, body pose and proxemics also. They believe that their assumption can lead to be more enjoyable for human interaction. Their hypothesis is that a telerobot should embody the operator in a way that gives them with enough representation remotely. Thus, it is able to take a perfectly involved part in the interaction and is perceived by their friends as being present closely, but remotely. The MeBot can allow people to show and express several of the non-verbal behavior portably that people use in normal face-to-face interactions (e.g., gestures of each human organ, interpersonal distance, and eye contact). Although this system is not allowed people to hug a robot as Huggy Pajama system, this study has shown interestingly that a socially expressive robot is dynamically more likable and engaging than a very static telerobot. Also more recently, this idea was extended to help sick children feel less sad and blue in the ICU of Boston Children's Hospital by [Jeo15a] from MIT Media Lab. Figure 6 depicts this extended Huggable's work from the Wired Magazine. This robot is a blue fancy bear built to relieve pain for kids who are hospitalized in Boston for cancer treatment. Under their small scale preliminary study conducted in the real hospital, sick children played and interacted with this blue fancy robotic bear in their private rooms. They shook his foot and played with it positively. Rather than hugging the whole body, Choi et al. [Cho14a] designed an interactively tele-hug ring wearable system for sending various kinds of finger hug, such as mini, intense, and urgent, depending on the sender from a remote distance through subtle colored lighting and tactile expressions digitally. Also, Wang et al. reaffirmed in [Wan12a] that mediated social touch in digital manner can quite influence the sense of link between a speaker and the experience emotionally. They evaluate the assumption by using an augmented storytelling approach. Furthermore, a capacitive touch communication approach was developed interactively by Vu et al. in [Vu12a]. This approach takes advantage of the capacitive touchscreens, which are currently used in many laptops and smartphones as a signal receiver. After that, this signal is able to be produced by a transmitter embedded into their devices. The devices they used in the system are ring and watch. They designed two example prototype systems with a single transmitter. This transmitter is low-power continuous, and can communicate through the skin and a finger ring bearing a signet touched to the screen.

Figure 6. A blue fancy robotic bear was designed to relieve pain for kids in the Wired Magazine [Fal15a].

5. SMELL COMMUNICATION

The next vital sense is smell. Smell has numerous benefits, since it spontaneously and quickly informs us about the state of things in our immediate vicinity [Row09a]. However, can smell be sent by computer as safely as we send other kinds of media, such as sight and touch? The answer is simply 'No'. This is because it usually requires some chemical substances to interact, so that the safety is obviously an important issue. For smell communication, Kaye et al. [Kay08a] presented an olfaction system for computerizing smell output. More specifically, they created a smell-based system called Dollars & Scents inside the entrance to their laboratory at MIT. The system releases scents and perfumes into the air differently depending on stock market changes. For example, they emit a lemon-like fragrance if NASDAQ index is going down, while they emit a rose-like fragrance if the stock market is going up. Similarly, they also built an olfaction system, called inStink, for conveying ambient presence and activity awareness for food. Moreover, Braun and Cheok [Bra14a] designed and built a smell-based computer-dream interface for dreaming of human. They assumed that emotions, particularly during sleep, and smell are intensively interlinked. In this way, they

studied the perception of emotions, when dreaming unconsciously, and how they can be affected from outside environment via smell-based computer-dream interfaces. Moreover, Choi et al. [Cho12a] from Keio University designed and created interactively an olfactory wearable application, called Light Perfume. This application automatically tracks several factors of each user (e.g., speed of eye blinking), and then it emits a delicate fragrance from a wearable device interactively during a face-to-face talk to affect a person's impression emotionally and softly.

6. TASTE COMMUNICATION

The last one is taste, probably one of the most difficult senses for communication. Taste bud is an importantly peripheral sensory organ and has also many benefits. We say this because the sense of taste, such as salty, sour, sweet and bitter, helps people decide what to consume and influences how effectively we digest the meal. In this way, it can affect feeling, mood and emotion straightly and dramatically [Rop06a]. Nevertheless, for taste communication, it is not trivial to transmit this sense. This is because this sense will be aroused when some chemical substances activate specialized receptor cells within the mouth cavity (e.g., cheeks, lips, palate, and floor of the mouth), as explained in [Bre13a], so that it often requires some chemical compounds to interact, largely due to the physical contact unavoidably. Wei et al. [Wei13a] from National University of Singapore designed and built an interactive multimodal system, called Food Media, for the telepresent family dinner context from a remote distance. Even though it is not directly and only for taste communication, it is intuitively an interactive exploration of suitable food and food activities as a medium for communication within the family warmly. In fact, it uses multi-sensory interactions from three major senses (i.e., touch, smell and taste) to connect each family member during the dinner hour. Figure 7 depicts the settlement of integrated system in a room-like environment using two interaction screens remotely. Ranasinghe et al. [Ran14a] developed a taste-based interactive control system, called Digital Flavor Interface. It is done by using electrical and thermal stimulation methods to create different tastes virtually and digitally on the tongue. It simulates different tastes such as sourness, saltiness, and bitterness by using two utensils, a bottle and a spoon as a small box. These utensils can enhance taste sensations differently, so that taste communication can be achieved. Interestingly, rather than just transmitting a normal taste on our touch, Saadatian et al. [Saa14a] built a haptic system interactively that allows people to send and transfer kiss physically over distance. They called it Kiss-Messenger, abbreviated with the acronym Kissenger. This lovotics application consists of a pair of charming wearable robots for couples to transmit kiss from our mouth remotely. This device is paired with another similarly. The lips are controlled by motors inside the two robots which are equipped with soft silicon pads. The shape of the kiss by the first user and amount of force that appeared to be necessary can be transferred to another device for the second user that is simulated as close as possible to the real kiss using actuators.

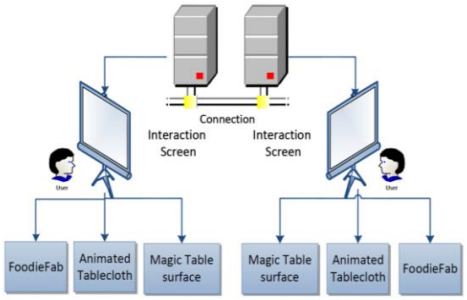


Figure 7. The Food Media's system configuration is set up by [Wei13a] from National University of Singapore.

7. CONCLUSIONS

Each of the five senses performs an important role in enhancing one's daily life experiences through moods, emotions and memory. Previously, communication was often only limited through human hearing system, such as mobile phone. However, in today's world, communication and digital media go far beyond than just audition. Augmented reality, computer vision, haptic, sensor and other new 21st century technologies expand communication into every sensory modality, from vision to touch and from smell to taste. In this paper, we propose a work-in-progress framework for new digital media research when applying modern technologies into every traditional sense of human interactively. It is done by dividing technological communications into five main categories as same as the senses of human: sight, touch, smell and taste systems. Each system has their unique difficulties. For digital sight communication, it has to deal with illumination robustness, computational speed and tracking accuracy. For digital touch communication, the main challenge is how to reproduce the perception realistically, naturalistically and correctly. For taste and smell systems, their main difficulty is usually about dealing with chemical elements. Hence, safety issue is still an essential part in these two senses. Although we believe that the communication for every sense of human has obviously so many challenges for human, it will lead to one big leap for the development of human being.

Hence, the research and development in this area is indispensable and evitable. We intend to build a robust system of the new digital media dealing with every sense to solve these aforementioned problems in the future.

8. ACKNOWLEDGMENTS

Some of this work presented herein was partially supported by a research grant from the Research Center, NIDA (National Institute of Development Administration).

9. REFERENCES

[Ada10a] S. Adalgeirsson and C. Breazeal, Mebot, a robotic platform for socially embodied telepresence, In Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction, pp. 15–22, Osaka, Japan, 2010.

[Bai12a] Diane E. Bailey, Paul M. Leonardi, and Stephen R. Barley, The lure of the virtual, Organization science: Journal of the Institute for Operations Research and the Management Sciences; bridging disciplines to advance knowledge of organizations (INFORMS), Vol. 23, No. 5, pp. 1485–1504, 2012.

[Bra14a] Marius H. Braun and Adrian D. Cheok, Towards an olfactory computer-dream interface, In Proceedings of the 11th Conference on Advances in Computer Entertainment Technology (ACE), Article No. 54, ACM New York, NY, USA, 2014.

[Bre13a] Breslin, P. A. S., An Evolutionary Perspective on Food Review and Human Taste. Current Biology : CB, 23(9), R409–R418, 2013.

[Cha14a] Joseph T. Chao, Tanxin Du, Christopher Wagenheim, Theodore Rippey, and Mise en Scène: A Film Scholarship Augmented Reality Mobile Application, Interdisciplinary Journal of Information, Knowledge, and Management, Volume 9 , 2014.

[Che13a] Adrian Cheok, "Making a Huggable Internet over" on IEEE Spectrum, January 2013.

[Cho12a] Yongsoon Choi, Rahul Parsani, Xavier Roman, Anshul Vikram Pandey, Adrian David Cheok, Light perfume: Designing a wearable lighting and olfactory accessory for empathic interactions, In Proceedings of the Advances in Computer Entertainment, Lecture Notes in Computer Science (LNCS), Volume 7624, pp 182-197, Publisher Springer Berlin Heidelberg, 2012.

[Cho14a] Yongsoon Choi, Jordan Tewell, Yukihiro Morisawa, Gilang A. Pradana, Adrian David Cheok, Ring*U: a wearable system for intimate communication using tactile lighting expressions, In the Proceedings of the 11th Conference on Advances in Computer Entertainment Technology (ACE), Article No. 63, ACM New York, NY, USA, 2014.

[Jam09a] Carrie James, Young People, Ethics, and the New Digital Media, A Synthesis from the GoodPlay Project, Project Zero, Harvard Graduate School of Education, MIT Press, Cambridge, Massachusetts, ISBN 978-0-262-51363-0, 2009.

[Fal15a] Sarah Fallon, A Blue Robotic Bear to Make Sick Kids Feel Less Blue, the Wired Magazine, March 2015.

[Jeo15a] Sooyeon Jeong, Deirdre E. Logan, Matthew S. Goodwin, Suzanne Graca, Brianna O'Connell, Honey Goodenough, Laurel Anderson, Nicole Stenquist, Katie Fitzpatrick, Miriam Zisook, Luke Plummer, Cynthia Breazeal, Peter Weinstock, A Social Robot to Mitigate Stress, Anxiety, and Pain in Hospital Pediatric Care. In Proceedings of the 10th Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts, New York, NY, USA, 2015.

[Kay08a] Kaye, J., Computer-controlled smell output, Perfumer & Flavorist. Chapter 36, Fragrance for Personal Care. Allured Publishing, 2008.

[Ker14a] C. Kerdvibulvech and K. Yamauchi, "Structural Human Shape Analysis for Modeling and Recognition," In Proceedings of the Joint IAPR International Workshops on Structural and Syntactic Pattern Recognition (S+SSPR), Lecture Notes in Computer Science (LNCS), Volume 8621, Joensuu, Finland, Springer-Verlag Berlin Heidelberg, pp.282–290, August 20-22, 2014.

[Kra10a] Michael W. Kraus, Cassy Huang, Dacher Keltner. Running Head: Touch, Cooperation and Performance. Tactile Communication, Cooperation, and Performance: An Ethological Study of the NBA, University of California, Berkeley, 2010.

[Kim14a] Eugene Kim & Jaewon Choi, Fine Dust in Augmented Reality: Creating Public Service Announcement, COMPUSOFT, An international journal of advanced computer technology, 3 (11), Volume-III, Issue-XI, November-2014.

[Kre10a] V. Krevelen, D. W. F. and Poelman, R., A Survey of Augmented Reality Technologies, Applications and Limitations, International Journal of Virtual Reality, vol. 9(2), pp.1-20, 2010.

[Loc13a] Markus Löchtefeld, Matthias Böhmer, Florian Daiber, Sven Gehring, Augmented Reality-Based Advertising Strategies for Paper Leaflets, In Proceedings of the ACM Conference on Pervasive and Ubiquitous Computing Adjunct

Publication, UbiComp Adjunct, pp.1015-1022, ACM, 2013.

[Mac01a] Blair MacIntyre, Jay David Bolter, Emmanuel Moreno, and Brendan Hannigan. Augmented Reality as a New Media Experience, In Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR), Page 197, IEEE Computer Society Washington, DC, USA, 2001.

[Mat09a] Hertenstein, Matthew J.; Holmes, Rachel; McCullough, Margaret; Keltner, Dacher, The communication of emotion via touch. Emotion, Vol 9(4), 566-573, Aug 2009.

[Mar13a] Rick Martin, "The next step in augmented reality: Electrify your taste buds", SD Japan, June 2013.

[Mur14a] Kazuyoshi Murata, Kensuke Usui, and Yu Shibuya, Effect of Haptic Perception on Remote Human-Pet Interaction, Human Interface and the Management of Information. Information and Knowledge Design and Evaluation, Lecture Notes in Computer Science (LNCS), Volume 8521, 2014, pp.226-232, In Proceedings of the 16th International Conference, HCI International 2014, Heraklion, Crete, Greece, June 22-27, 2014.

[New15a] Nic Newman, Media, Journalism and Technology Predictions 2015, Reuters Institute for the study of Journalism, January 2015.

[Ran14a] Nimesha Ranasinghe, Gajan Suthokumar, Kuan Yi Lee, and Ellen Yi-Luen Do. ``Digital flavor interface," In Proceedings of the adjunct publication of the 27th annual ACM symposium on User interface software and technology (UIST), pp.47-48. ACM, 2014.

[Ric12a] Rice, R.E. and Leonardi, P. M., Information and Communication Technology Use in Organizations. In L.L Putnam & D. K. Mumby (Eds.) The Sage Handbook of Organizational Communication, 2012.

[Rop06a] S. D. Roper, Signaling in the Chemosensory Systems, Cellular and Molecular Life Sciences CMLS, Volume 63, Issue 13, pp.1494-1500, July 2006.

[Row09a] David J Rowe. Chemistry and Technology of Flavours and Fragrances, Wiley-Blackwell, February 2009.

[Rus14a] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, Li Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge", In Proceedings of the Computer Vision and Pattern Recognition, Cornell University Library, 2014.

[Saa14a] Elham Saadatian, Hooman Samani, Rahul Parsani, Anshul Vikram Pandey, Jinhui Li, Lenis Tejada, Adrian David Cheok, Ryohei Nakatsu, Mediating intimacy in long-distance relationships using kiss messaging, International Journal of Human-Computer Studies, Volume 72, Issue 10, pp.736-746, 2014.

[Sin14a] Poonam Singh and Mrinalini Pandey, Augmented Reality Advertising: An Impactful Platform for New Age Consumer Engagement, IOSR Journal of Business and Management (IOSR-JBM), e-ISSN: 2278-487X, p-ISSN: 2319-7668. Volume 16, Issue 2. Ver. II, pp. 24-28, Feb. 2014.

[Sti06a] Walter Dan Stiehl, Kuk-Hyun Han, Jeff Lieberman, Levi Lalla, Allan Maymin, Jonathan Salinas, Daniel Fuentes, Robert Toscano, Cheng Hau Tong, Aseem Kishore, Matt Berlin, Jesse Gray, The huggable: a therapeutic robotic companion for relational, affective touch, In Proceedings of the ACM SIGGRAPH 2006 Emerging technologies, Article No. 15, ACM New York, NY, USA, 2006.

[Teh08a] James Keng Soon Teh and Adrian David Cheok, Pet internet and huggy pajama: A comparative analysis of design issues, The International Journal of Virtual Reality, Volume 7, Issue 4, Pp. 41-46, 2008.

[Vau09a] Vaughan-Nichols, S.J., Augmented Reality: No Longer a Novelty, IEEE Computer (Volume:42 , Issue: 12), pp.19-22, Dec. 2009.

[Vu12a] Tam Vu, Akash Baid, Simon Gao, Marco Gruteser, Richard Howard, Janne Lindqvist, Predrag Spasojevic, Jeffrey Walling. Distinguishing Users with Capacitive Touch Communication, In Proceedings of the 18th annual international conference on Mobile computing and networking (Mobicom), pp.197-208, ACM New York, NY, USA, 2012.

[Wan13a] Xiangyu Wang, Mi Jeong Kim, Peter E.D. Love, Shih-Chung Kang, Augmented Reality in built environment: Classification and implications for future research, Automation in Construction, Volume 32, pp.1-13, July 2013.

[Wan12a] Rongrong Wang, Francis Quek, Deborah Tatar, Keng Soon The, Adrian Cheok, Keep in touch: channel, expectation and experience. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp.139148, ACM New York, NY, USA, 2012.

[Wei13a] Jun Wei, Adrian David Cheok, Shengdong Zhao, Food Media: Interactive Entertainment Over Telepresent Dinner, International Journal of Advanced Computer Science, Volume 3, Issue 12, 2013.