ISSN 2464–4617 (print) ISSN 2464–4625 (CD-ROM)

CSRN 2702

(Eds.)

Vaclav Skala
University of West Bohemia, Czech Republic

Computer Science Research Notes

25. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision
WSCG 2017
Plzen, Czech Republic
May 29 – June 2, 2017

Proceedings

WSCG 2017

Short Papers Proceedings

ISSN 2464–4617 (print) ISSN 2464–4625 (CD-ROM)

ISSN 2464–4617 (print) ISSN 2464–4625 (CD-ROM)

CSRN 2702

(Eds.)

Vaclav Skala
University of West Bohemia, Czech Republic

Computer Science Research Notes

25. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision
WSCG 2017
Plzen, Czech Republic
May 29 – June 2, 2017

Proceedings

WSCG 2017

Short Papers Proceedings

Vaclav Skala – Union Agency

ISSN 2464–4617 (print) ISSN 2464–4625 (CD-ROM)

Computer Science Research Notes
CSRN 2702

Editor-in-Chief: 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

Publisher & Author Service Department & Distribution:
Vaclav Skala - UNION Agency
Na Mazinach 9
CZ 322 00 Plzen
Czech Republic
Reg.No. (ICO) 416 82 459

ISSN 2464-4617 (Print) ***ISSN 2464-4625*** (CD/DVD)
ISBN 978-80-86943-50-3 *(Print)* ***ISBN 978-80-86943-45-9*** *(CD/DVD)*

WSCG 2017

International Program Committee

Adzhiev, Valery (United Kingdom)
Anderson, Maciel (Brazil)
Benes, Bedrich (United States)
Bilbao, Javier,J. (Spain)
Bourke, Paul (Australia)
Daniel, Marc (France)
de Geus, Klaus (Brazil)
Drechsler, Klaus (Germany)
Feito, Francisco (Spain)
Ferguson, Stuart (United Kingdom)
Galo, Mauricio (Brazil)
Giannini, Franca (Italy)
Gobbetti, Enrico (Italy)
Gudukbay, Ugur (Turkey)
Juan, M.-Carmen (Spain)
Kenny, Erleben (Denmark)
Kim, Jinman (Australia)
Kim, HyungSeok (Korea)
Lobachev, Oleg (Germany)
Molla, Ramon (Spain)
Montrucchio, Bartolomeo (Italy)
Muller, Heinrich (Germany)
Murtagh, Fionn (United Kingdom)
Pan, Rongjiang (China)
Pedrini, Helio (Brazil)
Platis, Nikos (Greece)
Ramires Fernandes, Antonio (Portugal)
Richardson, John (United States)
Ritter, Marcel (Austria)
Rojas-Sola, Jose Ignacio (Spain)
Sanna, Andrea (Italy)
Segura, Rafael (Spain)
Skala, Vaclav (Czech Republic)
Sousa, A.Augusto (Portugal)
Szecsi, Laszlo (Hungary)
Teschner, Matthias (Germany)
Tokuta, Alade (United States)
Umetani, Nobuyuki (Japan)
Wu, Shin-Ting (Brazil)
Wuensche, Burkhard,C. (New Zealand)
Wuethrich, Charles (Germany)
Yao, Junfeng (China)

WSCG 2017

Board of Reviewers

Aburumman, Nadine (France)
Adzhiev, Valery (United Kingdom)
Anderson, Maciel (Brazil)
Assarsson, Ulf (Sweden)
Averkiou, Melinos (Cyprus)
Ayala, Dolors (Spain)
Benes, Bedrich (United States)
Bilbao, Javier,J. (Spain)
Capobianco, Antonio (France)
Carmo, Maria Beatriz (Portugal)
Charalambous, Panayiotis (Cyprus)
Cline, David (United States)
Daniel, Marc (France)
Daniels, Karen (United States)
de Geus, Klaus (Brazil)
De Martino, Jose Mario (Brazil)
de Souza Paiva, Jose Gustavo (Brazil)
Diehl, Alexandra (Germany)
Dokken, Tor (Norway)
Dong, Yue (China)
Drechsler, Klaus (Germany)
Eisemann, Martin (Germany)
Feito, Francisco (Spain)
Ferguson, Stuart (United Kingdom)
Galo, Mauricio (Brazil)
Garcia-Alonso, Alejandro (Spain)
Gdawiec, Krzysztof (Poland)
Giannini, Franca (Italy)
Gobbetti, Enrico (Italy)
Gobron, Stephane (Switzerland)
Goncalves, Alexandrino (Portugal)
Gonzalez, Pascual (Spain)
Gudukbay, Ugur (Turkey)
Hernandez, Benjamin (United States)
Jones, Mark (United Kingdom)
Juan, M.-Carmen (Spain)
Kenny, Erleben (Denmark)
Kim, HyungSeok (Korea)
Kim, Jinman (Australia)
Kurillo, Gregorij (United States)
Kurt, Murat (Turkey)
Lee, Jong Kwan Jake (United States)
Liu, Yang (China)
Liu, Beibei (United States)
Liu, SG (China)
Liu, Damon Shing-Min (Taiwan)
Lobachev, Oleg (Germany)
Luo, Shengzhou (Ireland)
Marques, Ricardo (Spain)
Mei, Gang (China)
Mellado, Nicolas (France)
Meng, Weiliang (China)
Mestre, Daniel,R. (France)
Meyer, Alexandre (France)
Molina Masso, Jose Pascual (Spain)
Molla, Ramon (Spain)
Montrucchio, Bartolomeo (Italy)
Muller, Heinrich (Germany)
Murtagh, Fionn (United Kingdom)
Nishio, Koji (Japan)
Oberweger, Markus (Austria)
Oyarzun Laura, Cristina (Germany)
Pan, Rongjiang (China)
Pedrini, Helio (Brazil)
Pereira, Joao Madeiras (Portugal)
Pina, Jose Luis (Spain)
Platis, Nikos (Greece)
Puig, Anna (Spain)
Raidou, Renata Georgia (Austria)
Ramires Fernandes, Antonio (Portugal)

Richardson, John (United States)
Ritter, Marcel (Austria)
Rodrigues, Joao (Portugal)
Rojas-Sola, Jose Ignacio (Spain)
Sanna, Andrea (Italy)
Schwaerzler, Michael (Austria)
Segura, Rafael (Spain)
Serano, Ana (Spain)
Sik-Lanyi, Cecilia (Hungary)
Sommer, Bjorn (Germany)
Sousa, A.Augusto (Portugal)
Szecsi, Laszlo (Hungary)
Teschner, Matthias (Germany)
Todt, Eduardo (Brazil)
Tokuta, Alade (United States)
Tytkowski, Krzysztof (Poland)
Umetani, Nobuyuki ()
Umlauf, Georg (Germany)
Vanderhaeghe, David (France)
Vidal, Vincent (France)
Vierjahn, Tom (Germany)
Wu, Shin-Ting (Brazil)
Wuensche, Burkhard,C. (New Zealand)
Wuethrich, Charles (Germany)
Yao, Junfeng (China)
Yoshizawa, Shin (Japan)
YU, Qizhi (United Kingdom)
Zhao, Qiang (China)

WSCG 2017

Short Papers Proceedings

Contents

A fast and robust level set motion-assisted deformable registration method for surgical platform: Comparison with conventional methods

Daegwan Kim
Daegu-Gyeongbuk Medical Innovation Foundation
Daegu, Republic of Korea
dgkim0306@dgmif.re.kr

Namkug Kim
Asan Medical Center, Seoul, Republic of Korea
namkugkim@gmail.com

Wookeun Kim
Daegu-Gyeongbuk Medical Innovation Foundation
Daegu, Republic of Korea
wekim@dgmif.re.kr

Jongkyun Choi
Daegu-Gyeongbuk Medical Innovation Foundation
Daegu, Republic of Korea
cjk@dgmif.re.kr

Teaha Song
Daegu-Gyeongbuk Medical Innovation Foundation
Daegu, Republic of Korea
taeson@dgmif.re.kr

Semi Jeong
Daegu-Gyeongbuk Medical Innovation Foundation
Daegu, Republic of Korea
semi@dgmif.re.kr

Joon Beom Seo
Asan Medical Center, Seoul, Republic of Korea
joonbeom.seo@gmail.com

Sangmin Lee
Asan Medical Center, Seoul, Republic of Korea
sangmin.lee.md@gmail.com

ABSTRACT

A level set motion assisted deformable registration method for diagnostic computed tomography (CT) and preoperative CT images were proposed and its accuracy and speed were compared with those of other conventional methods. Fifteen 3D CT images obtained from lung biopsy patients were scanned. Each scan consisted of diagnostic and preoperative CT images. Each deformable registration method was initially evaluated with a landmark-based affine registration algorithm. Various deformable registration methods such as level set motion, demons, diffeomorphic demons, and b-spline were compared. Visual assessment by two expert thoracic radiologists using five scales showed an average visual score of 3.2 for level set motion deformable registration, whereas scores were below 3 for other deformable registration methods. In the qualitative assessment, the level set motion algorithm showed better results than those obtained with other deformable registration methods. A level set motion based deformable registration algorithm was effective for registering diagnostic and preoperative volumetric CT images for image-guided lung intervention.

Keywords

Deformable registration, Surgical platform, 3D CT image, Level set motion, Lung intervention, visual scoring.

1. INTRODUCTION

Image-guided intervention is a medical procedure that was developed as an alternative to the traditional surgical process. Image-guided procedures have a lower risk of complications than traditional procedures because of the short operating times and small incisions. A fast patient recovery is another advantage of image-guided intervention.

Image registration is performed using either rigid registration or deformable registration algorithms [1]. Various rigid image registration algorithms have been described [2-5], and several deformable image registration techniques have been developed for use in medicine as rigid registration algorithms were implemented. Medical imaging applications based on these registration techniques are undergoing wide-scale development for specific purposes and in different medical environments. The use of the appropriate deformable registration technique in each case may improve the results of image-guided intervention.

The purpose of the present study was to suggest appropriate image registration procedures and suitable deformable registration methods for image-guided intervention in the lung. We used a level set motion method and compared it with other de-

formable image registration techniques such as the demons, diffeomorphic demons, and b-spline algorithms. The initial rigid registration was computed using an affine transformation approach, and level set motion registration was used for preoperative image registration.

2. METHOD

In general, the image-guided intervention in the lung requires the registration of images of different respiratory states of the same patient, thus establishing a correspondence between the same lung structures in both images. There are two main approaches towards image registration. The overall procedure for a proposed registration method for robotic intervention consists of the following two main steps as out-lined in Figure 1: 1) computing geometric transformation that aligns the planning CT image and preoperative CT images given a set of pair landmarks and 2) matching of the lung structure with the deformable registration method.

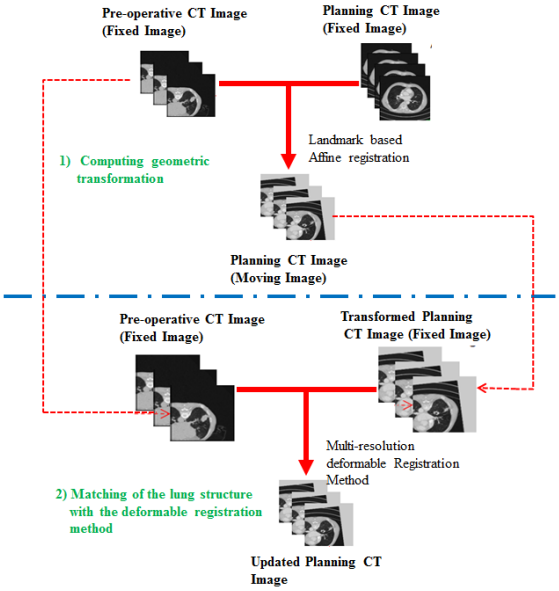


Figure 1. Workflow of the deformable image registration algorithm for robotic intervention.

Figure 2 shows preoperative (fixed) CT and planning CT (moving) images for the application of landmark-based transformation algorithms.

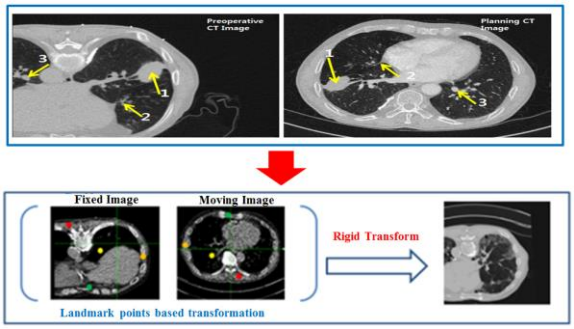


Figure 2. Initial landmarks on preoperative CT and planning CT images: three point landmarks are selected by clinical experts.

In this case, the planning CT scans were performed with the patient in the supine position, whereas the pre-operative CT images were acquired in the prone position because of the intervention procedures. To align the positions, the clinical experts extracted three landmark points from consolidations, lung anatomy or nodules on these images. The following sequence of steps is typical for the extraction of landmarks using preoperative and planning images: 1.Planning and preoperative CT images are acquired; 2.The clinical experts check the position of the patient (supine or prone); 3.The clinical experts search for anatomical landmarks such as lesions, consolidations, or vessels in the planning and preoperative CT images; 4.The clinical experts use the computer system to extract the center point of the anatomical landmarks on display; 5.CT images with a set of point landmarks are obtained upon procedure completion.

After extracting landmarks, the planning and preoperative CT image is aligned by the affine transformation algorithm [6,7]. The initial part of the algorithm looks at defining similar landmarks between two different images, and solving the equations based off of said landmarks to obtain an affine transformation between the source image and the target image. Due to the fact that an affine transformation is defined by six parameters, it is necessary to have at least three landmarks defined in each image. Second step is to compare four automatic methods to perform the deformable registration of image-guided lung intervention; 1) Demons and 2) diffeomorphic demons registration widely used deformable image registration algorithms. 3) Level set motion registration using level set theory. 4) The deformable registration model based on B-splines.

In this study, we focused on validation of level set deformable registration for the image-guided lung intervention. Level set methods measures intensity difference between the two image whereas demons registration method calculates gradients of the moving image on a smoothed image. As shown in Figure 2, the quality of planning and preoperative CT images differed in this study. CT image quality is related to radiation dose, with increased radiation dose resulting in better image quality [8]. As with other diagnostic-type CT images, the quality of a planning CT image is important for a confident diagnosis and to design intervention plans. However, in this case, the quality of preoperative CT images is low, and the reduction in the radiation dose to the patient results in noise.

Deformable registration based on Level set motion

The level set framework for image registration was proposed by Vemuri [9]. The equation was derived from the optical flow equation. The main idea of the level set motion framework is to register iteratively image I1(x) along its gradient direction until it deforms to the target image I2(x).

The equation for motion is similar to those of the original demons registration.

To transform the level-sets of the intensity function of the source image into the level-sets of the intensity function of the target image, the formulation of the registration is as follows:

$$I_t(X,t) = I_2(X) - I(X,t)\left\|\nabla I(X,t)\right\|, \quad \text{with } I(X,0) = I_1(X) \qquad (1)$$

where $I_2(X)$-$I(X,t)$ is the speed of evolution.

Using the above equation, we can derive geometric transformation explicitly be-tween the source and the target images for image morphing. The equation is ex-pressed as follows:

$$\vec{V}_t = [I_2(X) - I_1(\vec{V}(X))]\frac{\nabla I_1(\vec{V}(X))}{\left\|\nabla I_1(\vec{V}(X))\right\|}, \quad \text{with } \vec{V}(X,0) = \vec{0} \qquad (2)$$

where $\vec{V} = (u,v,w)^T$ is the displacement vector at in the 3D case and the operator is $V(X) = (x-u, y-v, z-w)^T$.

For additional mathematical details, we refer the reader to Vemuri et al. 2000 and 2003 [8,9]. As in most image-based registration methods, a multi-resolution approach is applied to registration algorithms to improve the robustness and convergence speed of the algorithm.

3. MATERIALS

The images were selected from clinical data sets obtained from 15 patients. All data sets were resampled to a resolution of 512×512×30 voxels to compare the registration performance and computation time. All experiments were performed using a 64 bit Intel Processor (2.4 GHz and 16 GB of RAM) running Windows because algorithms tend to use a large amount of system memory.

4. RESULTS

The registration techniques can be evaluated using various methods. In the present study, several quantitative assessment and visual scoring systems were used by clinical experts to evaluate the deformable registration methods.

Visual scoring by clinical experts

Two clinical experts independently scored the visual alignment of the lung for each registered CT image obtained using four techniques (level set motion, demon, diffeomorphic demons, and b-spline) and compared it with the corresponding fixed image. All deformable registration algorithms for the 15 patient data sets were assessed according to three criteria: lung anatomy, soft tissues, and lesions. The alignment of lung anatomy between the registered and reference images was considered only within the boundaries of the lung structure.

The visual scores ranged from 1 to 5, with 1 being the worst and 5 the best. Table 1 shows the assessment criteria for the scoring system, which were provided to each expert.

Rating score	Indicated quality	Definition
5	Excellent	Perfect registration (lung structure, soft tissue, and lesion)
4	Good	Almost completely registered with very minor mis registration
3	Adequate	Well registered with some mis-registration
2	Poor	Not well registered
1	Very poor	Not registered in entire images

Table 1. Definition of the 5-point rating score.

The mean (± standard deviation (SD)) of scores for the three assessment criteria were 3.12 (0.49), 2.18 (0.59), 2.74 (0.67) and 2.26 (0.69) for level set motion, demons, diffeomorphic demons, and b-spline, respectively. These were the average scores derived from 60 registration results (15 CT patient data sets, each with four registered images), obtained by two clinical experts using three assessment criteria, totaling 360 visual scores. The summary results of the visual scores for the four registration algorithms are presented graphically in Figure 3.

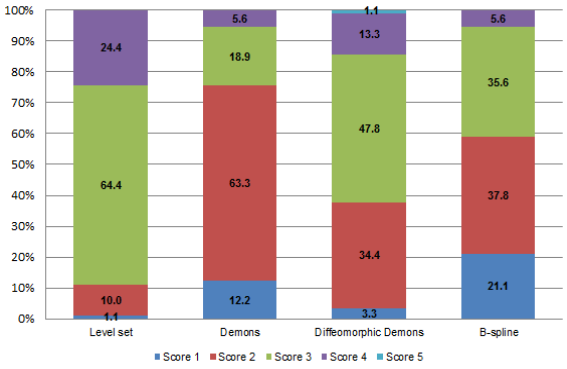


Figure 3. Summary of the visual scoring for each of the four registration techniques.

The results of comparisons performed using the paired t-test showed significant differences in the visual scores (Table 2).

Algorithms	T test	P value
Level set vs. Demons	9.23	0.000
Level set vs. Diffeomorphic demons	3.79	0.001
Level set vs. B-spline	6.95	0.000

Table 2. Paired t-test results for comparison of visual scores between level set motion and other image registration techniques.

Comparisons among all assessment criteria indicated that the level set motion deformable registration algorithm was significantly superior to the demons deformable registration, diffeomorphic demons registration, and b-spline deformable registration ($p<0.05$) with respect to all assessment criteria.

Registration assessment

To evaluate the accuracy of each registration method, we calculated the root mean square error (RMSE), median absolute deviation (MAD), and entropy. These measures are estimates of the quality and accuracy of registered images [10]. The mean and SD of the RMSE, MAD, and entropy are shown in Table 3.

	Mean (Standard Deviation)		
	RMSE	MAD	Entropy
Level set	18.67 (7.84)	5.27 (3.88)	8.98 (0.76)
Demons	24.21 (7.39)	6.73 (5.08)	9.20 (0.85)
Diff. Demons	21.31(6.89)	5.93 (4.23)	9.38 (1.01)
B-Spline	26.79 (6.70)	6.73 (4.85)	9.18 (0.84)

Table 3. Quantitative assessments of root mean squared error of intensity difference and median absolute deviation of intensity difference.

The RMSE, MAD, and entropy of the level set motion registration were lower than those of the other registration methods.

4. DISCUSSION

In the present study, the proposed level set motion registration technique achieved significantly better lung alignment than other deformable registration techniques for robotic intervention. Although a comparative analysis of level set motion registration and other deformable registration approaches has been made [10,11], previous studies provided a limited validation of the alignments achieved and the procedures were tested using images of similar quality that were acquired from patients.

Unlike previous studies, the present study assessed a level set motion deformable registration algorithm for robotic intervention. Such comparisons are important to define a simple and fast intervention procedure that might be required by a deformable registration algorithm. Although the overall best results were obtained using the level set registration, the diffeomorphic demon algorithm also achieved good results. No significant differences in visual scoring and qualitative assessment were observed between the level set motion and diffeomorphic demons algorithms. However, we showed that the level set motion registration technique yielded improved alignment results compared with other deformable registration techniques with regard to the robotic intervention. Moreover, the level set motion technique provides additional opportunities to recover misalignments beyond those of other deformable registration techniques during intervention procedures.

5. CONCLUSION

Overall, the level set motion technique showed good performance for the registration of diagnostic CT images versus preoperative CT images. Based on the visual and quantitative assessments, the level set motion technique showed good results regarding image registration for image-guided intervention in the lung. The evaluation revealed differences between the four registration techniques (Table 3 and Figure 3).

The results of the present study suggest that the proposed deformable registration technique can achieve a significantly superior alignment to that obtained with the currently available deformable registration techniques.

Future work should include upgrading of the framework of image guided intervention. First, a set of point landmarks should be selected automatically during the initial registration procedure. Second, the level set motion registration should be improved to reduce registration error and achieve faster computation times. Improved level set motion registration algorithms could be combined with other deformable registration algorithms

6. ACKNOWLEDGMENTS

This work is supported by “The Support for Joint Research & Development utilizing Daegu-Gyeongbuk Medical Device Development Center’s infrastructure” through the Ministry of Trade, Industry and Energy (MOTIE)
(Project Number 10049767, 2014)

7. REFERENCES

[1] Fitzpatrick, J.M., Sonka, M.: Handbook of Medical Imaging - Volume 2 Medical Image Processing and Analysis. In: Fitzpatrick, J.M., Hill, D.L.G., and Maurer, C.R. Jr. (ed), Image Registration, SPIE PRESS, Washington. pp 488-496, 2000.

[2] Ashburner, J.T., Friston, K.J.: Rigid body registration. In: Friston, K.J., Ashburner, J.T., Kiebel, S.J., Nichols, T.E., Statistical Parametric Mapping: The Analysis of Functional Brain Images. Academic Press, pp. 49–62, 2007.

[3] Fitzpatrick, J.M., West, J.B., Maurer, C.R. Jr.: Predicting error in rigid-body point based registration. IEEE Trans Med Img 17(5):694–702, 1998.

[4] Fitzpatrick, J.M., West, J.B.: The distribution of target registration error in rigid-body point-based registration. IEEE Trans Med Img 20(9):917–27, 2001

[5] Banerjee, S., Mukherjee, D., Majumdar, D.: Point landmarks for registration of CT and MR images. Pattern Recognition Lett 16 1033–1042, 1995

[6] Kim, E.Y., Johnson, H., Williams, N.: Affine transformation for landmark based registration initializer in ITK. The MIDAS Journal med Img and comp. http://hdl.handle.net/10380/3299, 2011

[7] Späth, H.: Fitting affine and orthogonal transfor-

mations between two sets of points. Math Comm 9:27–34, 2004

[8] Vemuri, B.C., Ye, J., Chen, Y., Leonard, C.M.: A level-set based approach to image registration. IEEE Workshop on Mathematical Methods in Biomedical Image Anlaysis:86-93, 2000

[9] Vemuri, B.C., Ye, J., Chen, Y., Leonard, C.M.: Image registration via level-set motion application to atlas-based segmentation. Med Image Anal 7:1–20, 2003.

[10] Urschler, M., Kluckner, S., Bischof, H.: A framework for comparison and evaluation of nonlinear intra subject image registration algorithms. The Insight Journal.

http://hdl.handle.net/1926/561, 2007

[11] Yang, D., Deasy, J., Low, D., Naqa, M.: Level set motion assisted non-rigid 3D image registration. Medical Imaging, Proceedings of SPIE, doi: 10.1117/12.710024, 2007

QuadSIFT: Unwrapping Planar Quadrilaterals to Enhance Feature Matching

Marco Filax
Chair of Software Engineering
Otto-von-Guericke University of Magdeburg
marco.filax@ovgu.de

Tim Gonschorek
Chair of Software Engineering
Otto-von-Guericke University of Magdeburg
tim.gonschorek@ovgu.de

Frank Ortmeier
Chair of Software Engineering
Otto-von-Guericke University of Magdeburg
frank.ortmeier@ovgu.de

ABSTRACT

Feature matching is one of the fundamental issues in computer vision. The established methods, however, do not provide reliable results, especially for extreme viewpoint changes. Different approaches have been proposed to lower this hurdle, e. g., by randomly sampling different viewpoints to obtain better results. However, these methods are computationally intensive.

In this paper, we propose an algorithm to enhance image matching under the assumption that an image, taken in man-made environments, typically contains planar, rectangular objects. We use line segments to identify image patches and compute a homography which unwraps the perspective distortion for each patch. The unwrapped image patches are used to detect, describe and match SIFT features.

We evaluate our results on a series of slanted views of a magazine and augmented reality markers. Our results demonstrate, that the proposed algorithm performs well for strong perspective distortions.

Keywords

Projective Transformation, Perspective Distortion, Feature Detection, SIFT

1 INTRODUCTION

Feature matching is one of the fundamental issues in computer vision. It was successfully applied in a variety of applications, e. g., Recognition [18], Tracking [6], Stitching [2] or Visual Odometry [23]. These feature matching algorithms are typically divided into three steps: detection, description, and matching.

There exists a variety of methods to detect features, e. g., Maximally Stable Extremal Regions [8], Scale Invariant Feature Transform (SIFT) [19], or Speeded Up Robust Features [1]. After that, during the description phase, spacial information of features is extracted - SIFT is one of the most popular methods. In the matching step, features descriptors of two images are compared by calculating the distance, e. g., the Euclidean distance for SIFT features. The goal is to find corresponding features

If the viewpoint change is reasonable small, all of the above-mentioned methods typically produce good results. However, if the viewpoint change is large enough, the problem of detecting, describing, and matching features becomes challenging. An example is illustrated in Fig. 1. There, a square S is viewed from two different viewpoints ψ_1 and ψ_2. The resulting images depict two different quadrilaterals: one for every viewpoint. State-of-the-art descriptors do not compensate perspective distortion as the projective transformation, wrapping S_{orig} to S_1 or S_2, is unknown in advance. Morel and Yu [22, 26] demonstrated that the established meth-

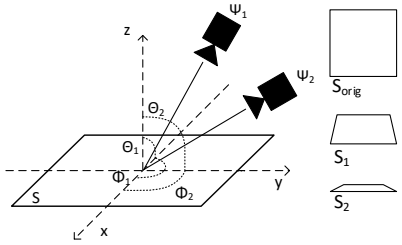


Figure 1: A planar square is viewed from two different viewpoints ψ_1 and ψ_2. S_{orig} represents a frontal view of a square. S_1 and S_2 demonstrate the projection of the square to different viewpoints. The square has been projected to two different quadrilaterals depending on the longitude ϕ and latitude θ values of the view-sphere [26].

ods, e. g., SIFT, do not provide suitable results, if the viewpoint change $|\theta_1 - \theta_2|$ is larger than 60°.

In this paper, we extend the well-known SIFT [19] framework. We increase the capabilities of SIFT by inverting the projective transformation introduced through a slanted view. Therefore, we focus on planar objects in man-made environments. Typically, objects in man-made environments represent some sort of structure, e. g., walls, windows or pictures - these structures form rectangular objects. This is especially the case for indoor environments. We exploit that observation to increase the capabilities of SIFT. We search for quadrilaterals, determined by a set of four non-collinear line segments. Given a valid quadrilateral, we determine the homography, that projects the quadrilateral into a square to unwrap the projective transformation. Finally, we utilize the SIFT framework to find correspondences. We evaluate our algorithm by matching different slanted views of a planar magazine and compare our approach with a well-known affine invariant extension of SIFT [22, 26].

The paper is structured as follows: In Sec. 2 related work of different authors is presented. We describe our approach in Sec. 3 and evaluate its performance in Sec. 4.

2 RELATED WORK

In this section, we summarize existing approaches aiming at detecting features in perspective distorted scenes.

Morel and Yu proposed an affine invariant feature matching approach - an extension of the well-known SIFT approach - called Affine SIFT (ASIFT) [22, 26]. ASIFT is designed to cope with viewpoint changes larger that 60°. To achieve this, different viewpoints of the original images are simulated by sampling the longitude and latitude of the view-hemisphere. SIFT features of two simulated images are matched and the highest amount of matches represents the result.

However, ASIFT is computationally intensive. For every parameter simulation features have to be detected, described and matched. To speed up the simulation, the authors proposed a two-resolution procedure. The parameter simulation is done with two low-resolution versions of the input images. In the case of success, the best set of parameters is used to compute SIFT Features on the original resolution images.

Cai et al. proposed an approach quite similar to ASIFT, called Perspective SIFT (PSIFT) [3]. The authors used a similar two-resolution procedure to increase the efficiency. In contrast to ASIFT, PSIFT unwraps perspective distortion by calculating a homography according to the longitude and latitude parameters of the view-hemisphere. The authors double the amount of parameters that have to be sampled, which additionally increases the computational effort.

To relax with the computational intensity, an iterative approach, iterative SIFT (ISIFT), has been proposed [28]. ISIFT is quite similar to ASIFT: the approach transforms one image to another view, more similar to the other image, by simulating viewpoint changes. First, features in both images are detected, described and matched. A homography is estimated which maps points in one image to points in the other image. Then, a view is simulated, by inverting the homography matrix, such that it is more similar to the other image. Additionally, illumination differences are detected and eliminated. This process is repeated until it converges to a stable amount of matches.

This approach has a significant drawback: its success is based on the initial matching of the two images. If, e. g., because of a strong viewpoint change, the feature matching stage fails, the algorithm is not able to produce reliable results. Thus, other authors used regions to increase the performance of SIFT.

Chen et. al. proposed to extract regions with the popular maximally stable extremal region (MSER) [20] detector and fitted them into ellipses [4]. These ellipses are assumed to be circular in a frontal view. Thus, they are transformed into circular areas according to the ellipses parameters and their second-order moment. Finally, the authors used SIFT to detect, describe and match features in the circular area. However, we believe circular areas are not common in man-made environments. In our opinion, man-made environments are more likely to contain line segments. Many lines in the scene are typically parallel and others are orthogonal [24]. This is satisfied, e. g., for doors, windows, shelves, buildings, and signs. This assumption led us to the approach, proposed in this paper. A key concept is the detection of quadrilaterals.

Different authors tackled the problem of quadrilateral detection. Typically, a set of constraints is used to detect if line segments form a quadrilateral [30, 17, 9]. Leung et al. proposed an algorithm to detect a planar quadrilateral in order to project an image on a squared surface. The authors measured the overlap of a detected line segment and segments of the quadrilateral hypothesis. Further, they imposed different constraints: e. g., opposite sides are of similar lengths or the angles of adjacent line segments are within a specific range. In [30, 9] the authors required opposite sides to be almost parallel. However, these approaches are too restrictive to serve as quadrilaterals detectors for strong viewpoint changes. If the camera moves to a position nearly parallel to the planar rectangle, most of the constraints do not hold, as illustrated in Fig. 4.

3 QUADSIFT: QUADRILATERAL SIFT

Feature detection algorithms have been a subject of scientific research for decades. Although there has been a

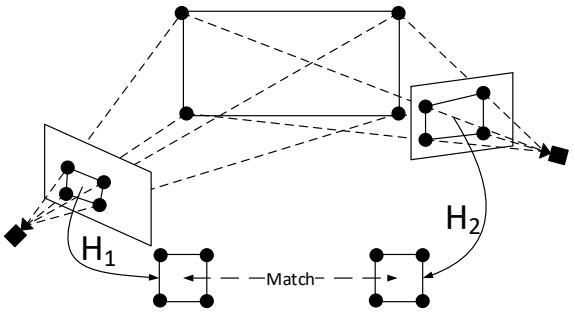


Figure 2: QuadSIFT: we detect quadrilaterals in two slanted views of a planar rectangle to compute two homography matrices that project the quadrilaterals to squared patches and finally detect, describe and match SIFT features.

lot of progress in that area, most feature detection algorithms suffer from the same problem: A strong viewpoint change results in a dramatic decrease of matching performance [22, 26]. This is because existing methods do not adhere to the perspective distortion introduced by a viewpoint change.

In this paper, we propose an extension of the well-known SIFT framework to revert the perspective distortion. To achieve this goal, we rely on additional information: rectangular structures. We make the following assumption to make that possible: The images used to detect, describe, and match features contain man-made objects. Based on the observation that objects in man-made environments typically contain line segments, we can assume that such an object most likely represent some rectangular structure, e. g., windows, walls or pictures. Given a perspective distorted rectangular object, we can revert the perspective distortion by wrapping the quadrilateral to a rectangle. Therefore, we propose an algorithm as illustrated in Fig. 2. We detect perspective distorted rectangles - quadrilaterals - in an image taken by a camera. We estimate the perspective distortion by calculating the homography that wraps a quadrilateral into a rectangular patch. In specific, we wrap the quadrilateral into a square as the aspect ratio of the physical rectangular structure is not known in advance. We detect, describe and match SIFT features in the squared patches.

Different problems have to be solved to reach this goal. First, we need to detect straight, linear contours: line segments. Second, scattered straight lines have to be grouped. Third, a quadrilateral has to be selected and a homography, unwrapping it into a squared patch, has to be computed. Finally, the unwrapped image area can be used to detect, describe and match SIFT features. We tackle every problem individually in the following sections.

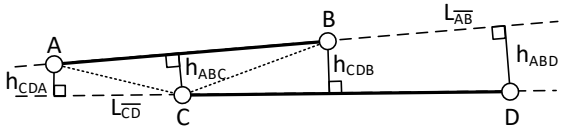


Figure 3: The line segment collinearity distance metric is defined as the absolute sum of the line segment endpoints heights to another line. The sum of all Euclidean distances (heights) $h_{ABC} + h_{ABD} + h_{CDA} + h_{CDB}$ defines the total absolute error.

3.1 Line Segment Detection

In the recent years, the scientific community made a huge progress in line detection algorithms - they became fast and reliable. There exists a variety of algorithms to detect lines and line segments, e. g., Hough Lines [7], probabilistic approaches [16], the Binary Descriptor [29], or Line Segment Detector [10, 11]. Choosing an appropriate detection algorithm is typically up to the specific use-case. Through a preliminary empirical evaluation, we found that the LSD detector yields better results than others with respect to the computation time for the given purpose. Thus, we used its OpenCV implementation to detect line segments longer than 75 pixels.

3.2 Detecting Scattered Line Segments

The detected line segments shall be used to rectify a quadrilateral. Therefore, we need to detect valid quadrilaterals from the set of line segments by selecting four line segments and evaluating their geometrical feasibility. However, if we view a perfectly rectangular object, e. g., a sheet of paper, from a viewpoint nearly collinear to the surface, its surface might be bent and thus, the line segments might be scattered into smaller segments or hidden by other objects.

To gain robustness for a quasi-planar object, we group nearly collinear line segments. We use an agglomerative clustering algorithm [5] to group the line segments. For this bottom-up approach, we need to define a distance metric specifying that two clusters have to be merged together. We use the relaxed collinearity property of two line segments as the distance metric.

We derive the distance metric as follows: Determining the collinearity of two line segments is a boolean operation. Either the line segments are collinear or not. We have to relax the collinearity property to use it as a distance metric, such that it is unaffected by the length, absolute positions and collinear distances of the line segments. Relying on the slope of a line segment does not account to parallel segments. Further, we cannot rely on the convex area defined by four endpoints of two line segments as it is affected by the length of the segments. Instead, we consider every combination of three endpoints as a triangle: If two line segments are

nearly collinear, the height of the four triangles, defined by line segments as its base and another line segments endpoint, becomes close to zero. Thus, we sum up the absolute heights of every triangle defined a line segment endpoints and another line segment endpoint.

An example for the proposed distance metric is depicted in Fig. 3. $\overline{AB}$ and $\overline{CD}$ represent two line segments with the endpoints A,B,C,D. $L_{\overline{AB}}$ and $L_{\overline{CD}}$ are the two lines, defined by the finite line segment. The endpoints define four triangles ABC, ABD, CDA and CDB. Note, that every triangle has to have one of the lines $L_{\overline{AB}}$ or $L_{\overline{CD}}$ as its base. The total distance is $dist_{\overline{AB}\,\overline{CD}} = h_{ABC} + h_{ABD} + h_{CDA} + h_{CDB}$ If $dist_{\overline{AB}\,\overline{CD}}$ is suitable small, we consider these line segments as collinear and group them to a single segment.

3.3 Quadrilateral Detection

The next step is to select a valid perspective distorted rectangle - a not self-intersecting convex quadrilateral. A convex quadrilateral is defined by four line segments. However, if we view a perfectly rectangular object under an extreme viewpoint, especially if we position the camera nearly collinear to the surface as shown in Fig. 1, we experience a series of problems.

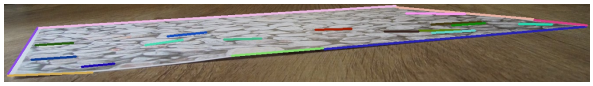


Figure 4: Slanted view of a sheet of paper. Note that, although the sheet is perfectly rectangular, opposite edges are not similar in length. The angles of two adjacent line segments vary strongly. Further, the edges of the paper are scattered into smaller line segments.

First, we detect quadrilaterals in man-made environments, based on line segments. Typically, quadrilaterals are detected with line segments, grouped by their vanishing points [25, 21]. However, if the set of line segments is sparse or includes a large set of outliers, these algorithms yield insufficient results. Given a minimal example, as shown in Fig. 4, vanishing point based grouping algorithms would fail, because there are only four line segments present. Reliably calculating two vanishing points is not possible given four line segments. Thus, we can not rely on vanishing points.

Second, we cannot impose constraints, that are typically used to detect quadrilaterals [15, 17, 12], e. g., opposite segments are of similar length, the angle of two intersecting line segment is within a specific range or two line segments have a similar slope. These conditions do not hold if the camera moves to a position nearly collinear to the planar surface. An example is depicted in Fig. 4.

To account for these difficulties, we evaluate the intersections of the lines, defined by the detected line segments. Other approaches simply evaluate intersection

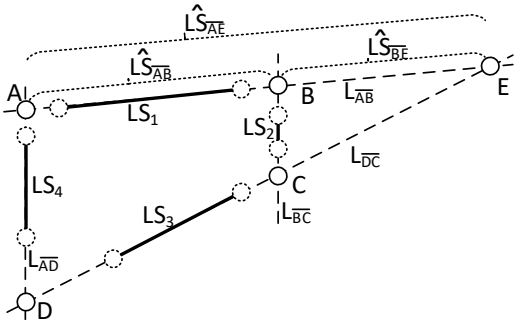


Figure 5: Given four line segments $LS_1,..,LS_4$ in general order: we compute their intersections A,B,C,D,E and eliminate outliers by evaluating the overlap ratio of detected line segments $LS_1,..,LS_4$ and virtual line segments, e. g., $\hat{LS}_{\overline{AB}}$, $\hat{LS}_{\overline{AE}}$, $\hat{LS}_{\overline{BE}}$.

points *within* a line segment or rely on vanishing points. While using intersections within a line segment restricts to rely on perfectly planar objects, detecting two or more vanishing points requires a sufficient amount of line segments. Our algorithm succeeds even for the minimum set of line segments and is robust against bent surfaces. The algorithm to detect valid self-intersecting convex quadrilaterals is summarized as follows: First, select four lines from the set of detected line segments. Second, compute the intersections of the lines to cope with scattered lines segments. Finally, remove outliers if there are more than four intersection points, by evaluating the overlap of line and line segments to eliminate invalid intersection points.

To demonstrate our approach we depict an example in Fig. 5. We selected four line segment $LS_1,..,LS_4$ from the set of all detected line segments. Then, based on their collinear lines $(L_{AB}, L_{BC}, L_{DC}, L_{AD})$, we can calculate the five intersections A, B, C, D and E. It is clearly visible that $ABCD$ represents the valid quadrilateral defined by $LS_1,..,LS_4$. This, however arises a problem: there exist multiple subsets of four intersection points describing a quadrilateral, e. g., $ABEC$. Thus, we have to detect the invalid intersection point E.

Other algorithms try to calculate vanishing points to rectify a given image [25, 21]. Given vanishing points it is possible, to cluster the line segment according to their geometrical feasibility. Then, one can detect valid quadrilaterals. However, as we have a set of exactly four line segments, where every intersection is defined by two lines, e. g., $L_{AB} \times L_{BC} = B$ and $L_{AB} \times L_{DC} = E$, it is not possible to reliably calculate vanishing points. Thus, we evaluate the feasibility of intersection points by maximizing the overlap ratios of detected line segments and their virtual line segments. We will use the example configuration depicted in Fig. 5 to describe the approach.

We identify valid quadrilateral points with the overlapping ratio of a detected line segment and a virtual line

segment. Virtual line segments are defined by their intersection points on a line collinear to a detected line segment and other lines. We exemplary depict the three virtual line segments for LS_1 in Fig. 5 $\hat{LS}_{\overline{AB}}$, $\hat{LS}_{\overline{AE}}$ and $\hat{LS}_{\overline{BE}}$. It is visible that $\hat{LS}_{\overline{AB}}$ and $\hat{LS}_{\overline{AE}}$ overlap LS_1 whereas $\hat{LS}_{\overline{BE}}$ does share an overlapping region with LS_1. Thus, $\hat{LS}_{\overline{BE}}$ does not represent a candidate. Further, we observe the order $O_{LS_1,\hat{LS}_{\overline{AB}}} > O_{LS_1,\hat{LS}_{\overline{AE}}}$ whereas $O_{LS,\hat{LS}}$ is the overlap ratio. This indicates that $\hat{LS}_{\overline{AB}}$ with its intersection points A and B represent a possible quadrilateral candidate. Four candidates are required to detect a quadrilateral. For the given example we found that $\hat{LS}_{\overline{AB}}$, $\hat{LS}_{\overline{BC}}$, $\hat{LS}_{\overline{CD}}$ and $\hat{LS}_{\overline{AD}}$ with the points A, B, C and D share the largest overlap with $LS_1, ..., LS_4$ and thus, $ABCD$ describes the valid quadrilateral.

3.4 Homography Estimation

In order to unwrap an image, we need to estimate a homography that transforms a given set of at least four points to into another set of points [13]. Given a valid quadrilateral in an image, defined by four points, we need to find four rectangular points.

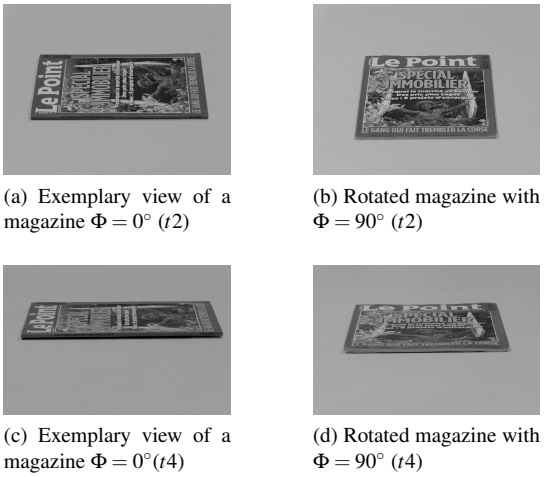
To determine four rectangular points with the help of a quadrilateral, Hua et al. [14] estimated a rectangle, whereas the aspect ratio of the physical rectangle is known. However, it is impossible to determine the aspect ratio for a quadrangle if the physical rectangle is unknown in advance. It is not possible to estimate the physical aspect ratio directly from a single image. The measured aspect ratio in an image can differ from the physical aspect ratio, especially in slanted views.

To overcome this issue, we chose to unwrap a detected quadrangle to a square. This eliminates the need of detecting the aspect ratio of an arbitrary quadrangle. We empirically chose to unwrap quadrangles into squares with 500-pixel length on each side.

Given four corner points of a detected quadrilateral and four corner points of a square, we calculate the homography mapping the quadrilateral to a square [13]. We use SIFT [19] to detect and describe features in two squared patches. We employ a brute force matching strategy [19]. To detect weather a descriptor in one squared patch matches some descriptor in the other squared patch we apply Lowe's ratio test: If the nearest distance of the best match for a descriptor is smaller than k times the second best match for that descriptor, the best match is considered to be valid, with k=0.8. Finally, we remove duplicated matches.

4 EVALUATION

We evaluate our method with the test set used by Morel and Yu [22]. We chose this test set because it is commonly used by other works in the domain. The test



(a) Exemplary view of a magazine $\Phi = 0°$ ($t2$)

(b) Rotated magazine with $\Phi = 90°$ ($t2$)

(c) Exemplary view of a magazine $\Phi = 0°$($t4$)

(d) Rotated magazine with $\Phi = 90°$ ($t4$)

Figure 6: Examples from the database: (a) depicts a slanted view of a magazine from $t2$. In (b) the magazine was rotated. (a) and (b) shall be matched later - SIFT is unable to detect matching features. (c) depicts a view of the magazine from $t4$. In contrast to (a), the camera was moved nearly collinear to the planar object. (d) depicts a rotated version of (c). (c) and (d) shall be matched later. Again, SIFT is unable to detect matching features.

set consists of different slanted and zoomed views of a magazine and other objects. As an evaluation on scale invariance of SIFT is not in the scope of this paper, we omit the zoomed views. We focus on the different slanted views of a magazine - resulting in two different test sets called $t2$ and $t4$. Fig. 6 depicts examples from the database. We determine the quality of our approach by determining the total number of feature correspondences found with the SIFT framework. The better our framework corrected the perspective distortion introduced through a viewpoint change the higher is the total number of correspondences. We compare the proposed approach with a well-known affine invariant extension of SIFT: ASIFT [22][1].

We implemented our method, described in Sec. 3, with OpenCV in C++. We used the OpenCV implementation of the Line Segment Detector [11] to find line segments in the images. With these segments, we detect quadrilaterals as described in Sec. 3.3. We used a greedy strategy to select a single quadrilateral by maximizing the quadrilateral area in an image. Then, we unwrapped the perspective distortion, introduced through a slanted view, by mapping the detected quadrilateral to a squared patch. We detect, describe and match SIFT features in the squares. We used a brute force matching algorithm, an outlier suppression proposed by Lowe [19] with k=0.8 and eliminated duplicated

[1] Implementation available online: `http://demo.ipol.im/demo/my_affine_sift/`

Φ	SIFT	ASIFT	QuadSIFT
10°	284	3496	1457
20°	22	2185	1399
30°	0	1729	1339
40°	0	1264	1215
50°	0	964	1142
60°	0	961	1054
70°	0	921	984
80°	0	879	918
90°	0	789	933

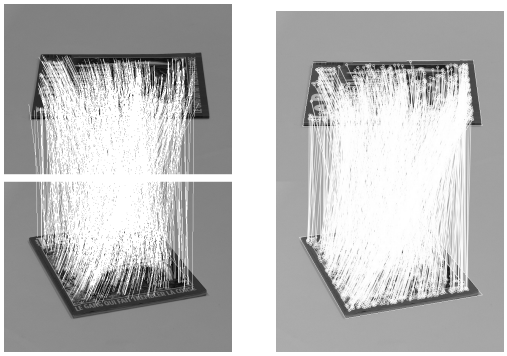
Table 1: Evaluation based on the test set $t2$ taken from [22]. Every row represents another rotation of the magazine denoted by Φ. We depict the number of correspondences when matched with Fig. 6(a) for every approach.

matches. Finally, we wrapped the matches to the original images and displayed the results.

We evaluate our algorithm by matching slanted views of a magazine. Examples of the test set $t2$ are depicted in Fig. 7. Table 1 depicts the result for test set $t2$. The first column depicts the rotation angle Φ of the magazine - every rotated magazine is matched against Fig. 6(a) with $\Phi = 0°$. The second column presents the total matches with the SIFT algorithm. The third column depicts ASIFT results. The last column depicts the results of QuadSIFT.

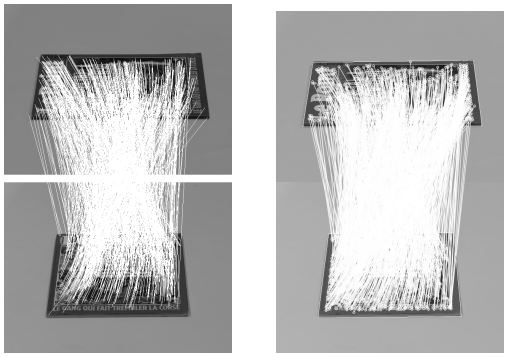
QuadSIFT is more efficient: QuadSIFT requires six seconds per image for the given test set on average. ASIFT required nearly ten seconds on average. For a rotation of the magazine from $\Phi = 10°$ to $\Phi = 30°$ ASIFT provides better results than QuadSIFT. This observation is present in every test set evaluated in this paper. This is because the implementations of ASIFT and QuadSIFT differ. While Morel and Yu used a pure C++ implementation of SIFT, we used the implementation available in OpenCV. If the rotation of the magazine is large enough QuadSIFT produces slightly more correspondences than ASIFT. In combination with the first observation, this means that QuadSIFT produces better results than ASIFT, if the perspective distortion is large enough.

The test set $t4$ is quite similar to $t2$. The major difference is that the camera moved to a viewpoint nearly collinear to the magazine plane - the perspective distortion is higher than the distortion in the test set $t2$. Examples from the test set are depicted in Fig. 6 (c) and (d). The results of the tests are shown in Tab. 2 and examples in are shown Fig. 8. Starting from $\Phi = 40°$ the number of correspondences calculated with QuadSIFT is higher than those calculated with ASIFT. Taking the differing implementations into account, this means, that QuadSIFT outperformed ASIFT. Again, ASIFT produces better results than QuadSIFT if the rotation angle is small.



(a) $\Phi = 70°$, ASIFT, 921 correspondences

(b) $\Phi = 70°$, QuadSIFT, 984 correspondences



(c) $\Phi = 90°$, ASIFT, 789 correspondences

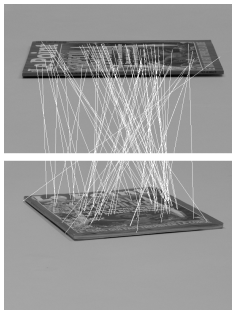
(d) $\Phi = 90°$, QuadSIFT, 933 correspondences

Figure 7: Evaluation based on the dataset $t2$ from [22]. Every row depicts the matching results of Fig. 6(a) and a rotated magazine. SIFT (not shown) found 0 correspondences.

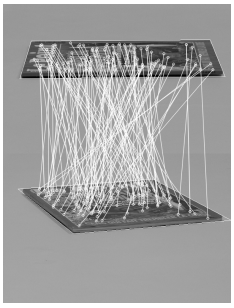
Finding nearly twice as many correspondences, for the two images of the magazine with a rotation of $\Phi = 90°$, we gain the following observation: If the distorted object is of a nearly quadratic shape, we achieve better results. We believe, that this observation is clarified if we consider the aspect ratio of the planar object. While ASIFT does not directly affect the aspect ratio of the

Φ	SIFT	ASIFT	QuadSIFT
10°	15	1079	380
20°	0	589	341
30°	0	310	293
40°	0	232	262
50°	0	130	198
60°	0	98	150
70°	0	74	120
80°	0	70	106
90°	0	43	104

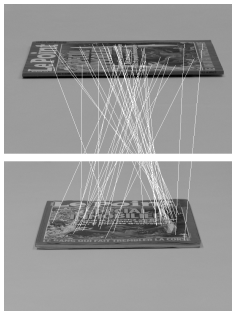
Table 2: Evaluation based on the test set $t4$ taken from [22]. Every row represents another rotation of the magazine denoted by Φ. We depict the number of correspondences when matched with Fig. 6(c) for every approach.



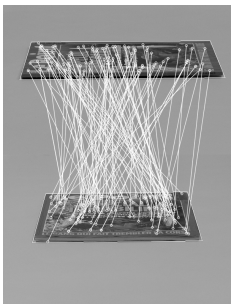
(a) $\Phi = 70°$, ASIFT, 74 correspondences



(b) $\Phi = 70°$, QuadSIFT, 120 correspondences



(c) $\Phi = 90°$, ASIFT, 43 correspondences



(d) $\Phi = 90°$, QuadSIFT, 104 correspondencess

Figure 8: The matching results of Fig. 6(c) and a rotated magazine from testset $t4$. SIFT (not shown) did not produce correspondences.

planar object, our approach is designed to map an object into a square. If we now consider quadratic objects captured by the camera, e. g., augmented reality markers, this drawback is negligible. If we consider non-quadratic objects, we can overcome this drawback if the aspect ratio is known in advance.

To clarify these claims, we evaluate our approach on an additional test set. We took ten images of a squared marker in a similar manner as Morel and Yu. We rotated the marker in between every image by 10°. In a similar manner, we took ten images of a rectangular marker with an aspect ratio of 3:1. We executed the proposed approach on both markers whereas the aspect ratio of the objects is unknown. The results are depicted in Tab. 3 and Tab. 4. Additionally, we used our approach with a known aspect ratio on the rectangular marker. We compared the results with ASIFT.

Tab. 3 depicts the number of correspondences calculated with SIFT, ASIFT, and QuadSIFT for the squared marker. We depicted an example in Fig. 9. It is clearly visible, that QuadSIFT produces better results than ASIFT. Comparing these results to the results of the test set $t4$, we come to the following observation: Increasing the rotation of the non-squared magazine from $\Phi = 0°$ to $\Phi = 90°$ decreases the number of found correspondences by 73%. Increasing the rotation of the squared marker from $\Phi = 0°$ to $\Phi = 90°$ decreases the

ϕ	SIFT	ASIFT	QuadSIFT
10°	498	4712	2681
20°	201	3778	2563
30°	23	2710	2418
40°	0	2424	2307
50°	0	2202	2189
60°	0	2080	2121
70°	0	1832	2073
80°	0	1701	2076
90°	0	1711	2021

Table 3: Evaluation based on a squared marker: Φ denotes the rotation of the marker. The markers are matched to $\Phi = 0$ (Shown in Fig. 9(a)). We depict the number of correspondences calculated with SIFT, ASIFT, and our approach.

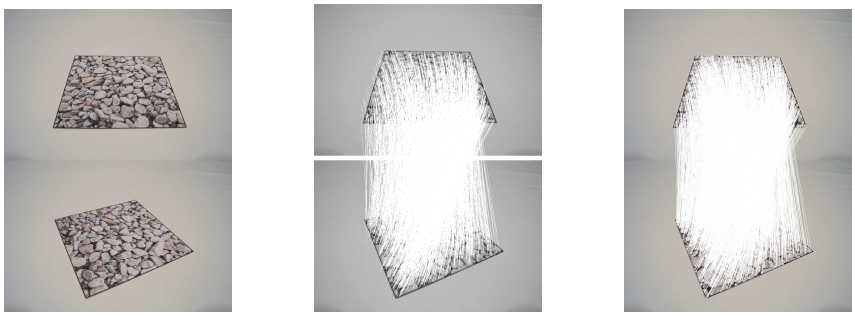
ϕ	ASIFT	QuadSIFT (1:1)	QuadSIFT (3:1)
10°	5450	1616	2469
20°	3702	1487	2410
30°	2382	1270	2145
40°	2292	1239	2065
50°	1885	1266	2006
60°	1530	1230	1827
70°	1379	1260	1815
80°	1257	1197	1641
90°	1348	1267	1727

Table 4: Evaluation based on a rectangular marker: Φ denotes the rotation of the marker. Further, we depict the number of correspondences calculated with SIFT, ASIFT, and the proposed approach with unknown (we assume an aspect ratio of 1:1) and a known aspect ratio (here 3:1).

number of found correspondences by only 25%. This is because of the squared shape of the marker. If the planar objected is close to a physical square our algorithm is superior.

To determine the influence of the aspect ratio, we calculated the absolute number of correspondences of a non-squared marker with known and unknown aspect ratio. The results are depicted in Tab. 4. Φ denotes the amount of rotation applied to the marker in between two images. The second column depicts the absolute number of correspondences calculated with ASIFT. The third and the fourth column depict the results of the proposed approach, once with an unknown aspect ratio - mapping the quadrilateral to square - (cf. third column) and one with known aspect ratio - mapping the quadrilateral into a 500x1500px rectangle (cf. fourth column). It is visible, that if the aspect ratio of the object is known in advance, the results of QuadSIFT are superior if the viewpoint change is large. However, if the aspect ratio is unknown in advance and the object is not of a squared shape, ASIFT produces slightly better results.

Based on the different tests evaluated in this paper we can formulate the following observations. Detecting

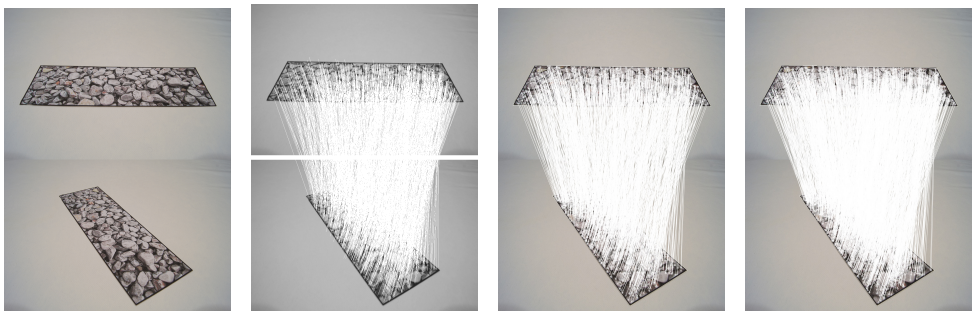


(a) A rotated, squared marker with $\Phi = 60°$

(b) ASIFT, 2080 Correspondences found

(c) QuadSIFT, 2121 Correspondences found

Figure 9: A comparison of ASIFT and QuadSIFT using a squared object. QuadSIFT produces more correspondences as ASIFT.



(a) A rotated, rectangular marker with $\Phi = 60°$

(b) 1530 ASIFT Correspondences

(c) QuadSIFT (1:1), 1230 Correspondences

(d) QuadSIFT (3:1), 1827 Correspondences

Figure 10: A comparison of ASIFT and QuadSIFT using a rectangular object. QuadSIFT produces more correspondences as ASIFT if the aspect ratio is known in advance.

quadrilateral objects to unwrap the perspective distortion enhanced feature matching results for extreme viewpoints. QuadSIFT performs similarly to ASIFT if the viewpoint change is reasonably large. If the viewpoint change is small, ASIFT outperforms QuadSIFT in terms of found correspondences. If perspective distortion is large enough QuadSIFT outperforms ASIFT. QuadSIFT seems to be more efficient than ASIFT. Knowing the aspect ratio in advance is beneficial. If the aspect ratio is unknown QuadSIFT performs similarly to ASIFT.

5 CONCLUSION

In this paper, we proposed a new method to enhance feature matching in slanted views of planar rectangular objects. Planar rectangular objects are transformed into quadrilaterals due to the perspective distortion introduced by a slanted view. We designed a method to revert the perspective distortion: Based on line segments, we detected valid quadrilaterals, select a single quadrilateral, and unwrapped the perspective distortion by computing a homography mapping a quadrilateral to a square. Finally, we detect, describe and match features using SIFT. We want to point out, that the feature extraction method is not restricted to SIFT. QuadSIFT can be realized with other feature matching frameworks.

As shown in the evaluation, our algorithm is capable of detecting, describing and matching features, especially in slanted views of planar rectangular objects. We found that QuadSIFT produces results comparable to the state-of-the-art algorithm ASIFT. If the physical, planar object is of a squared form, our approach is superior. Further, our results indicate that QuadSIFT is more efficient than ASIFT.

The proposed approach is designed to detect quadrilaterals. If an image does not contain line segments, e. g., in nonman-made environments, it will fail. Further, we restricted the quadrilateral selection to a greedy approach. We always select the largest quadrilateral. However, we plan to extend the proposed approach: we want to detect multiple quadrilaterals. This is especially interesting if there are multiple planar objects on different planes, e. g., in a corridor with planar objects on the walls or floor. Thus, we plan to incorporate a better quadrilateral detection, e. g., using an approach as described in [27].

6 REFERENCES

[1] Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-Up Robust Features (SURF). Comput.

Vis. Image Underst. 110(3), 346–359 (2008)

[2] Brown, M., Lowe, D.G.: Automatic panoramic image stitching using invariant features. In: Int. J. Comput. Vis. vol. 74, pp. 59–73. Kluwer Academic Publishers-Plenum Publishers (2007)

[3] Cai, G.R., Jodoin, P.M., Li, S.Z., Wu, Y.D., Su, S.Z., Huang, Z.K.: Perspective-SIFT: An efficient tool for low-altitude remote sensing image registration. Signal Processing 93(11), 3088–3110 (2013)

[4] Chen, M., Shao, Z., Li, D., Liu, J.: Invariant matching method for different viewpoint angle images. Appl. Opt. 52(1), 96–104 (2013)

[5] Chidananda Gowda, K., Krishna, G.: Agglomerative clustering using the concept of mutual nearest neighbourhood. Pattern Recognit. 10(2), 105–112 (1978)

[6] Donoser, M., Riemenschneider, H., Bischof, H.: Shape guided Maximally Stable Extremal Region (MSER) tracking. In: Proc. - Int. Conf. Pattern Recognit. pp. 1800–1803 (2010)

[7] Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. Commun. ACM 15(1), 11–15 (1972)

[8] Extremal, M.S., Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust Wide Baseline Stereo from. Br. Mach. Vis. Conf. pp. 384–393 (2002)

[9] Fung, H.K., Wong, K.H.: A Robust Line Tracking Method based on a Multiple Model Kalman Filter Model for Mobile Projector Systems. Procedia Technol. 11(2), 996–1002 (2013)

[10] Grompone Von Gioi, R., Jakubowicz, J., Morel, J.M., Randall, G.: LSD: A fast line segment detector with a false detection control. IEEE Trans. Pattern Anal. Mach. Intell. 32(4), 722–732 (2010)

[11] Grompone von Gioi, R., Jakubowicz, J., Morel, J.M., Randall, G.: LSD: a Line Segment Detector. Image Process. Line 2, 35–55 (2012)

[12] Hartl, A., Reitmayr, G.: Rectangular Target Extraction for Mobile Augmented Reality Applications. IEEE Int. Conf. Pattern Recognit. pp. 81–84 (2012)

[13] Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press (2004)

[14] Hua, G., Liu, Z., Zhang, Z., Wu, Y.: Automatic Business Card Scanning with a Camera. 2006 Int. Conf. Image Process. pp. 373–376 (2006)

[15] Jagannathan, L., Jawahar, C.: Perspective Correction Methods for Camera Based Document Analysis. Proc. First Int. Work. Camera-based Doc. Anal. Recognit. pp. 148–154 (2005)

[16] Kiryati, N., Eldar, Y., Bruckstein, A.M.: A probabilistic Hough transform. Pattern Recognit. 24(4), 303–316 (1991)

[17] Leung, M.C., Lee, K.K., Wong, K.H., Chang, M.M.Y.: A Projector-based Movable Hand-held Display System. CVPR (2009)

[18] Lowe, D.G.: Object recognition from local scale-invariant features. Proc. Seventh IEEE Int. Conf. Comput. Vis. 2([8), 1150–1157 (1999)

[19] Lowe, D.G.: Distinctive image features from scale invariant keypoints. Int'l J. Comput. Vis. 60, 91–11020042 (2004)

[20] Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. Image and vision computing 22(10), 761–767 (2004)

[21] Mičušík, B., Wildenauer, H., Vincze, M.: Towards detection of orthogonal planes in monocular images of indoor environments (2008)

[22] Morel, J.M., Yu, G.: ASIFT: A New Framework for Fully Affine Invariant Image Comparison. SIAM J. Imaging Sci. 2(2), 438–469 (2009)

[23] Nister, D., Naroditsky, O., Bergen, J.: Visual odometry for ground vehicle applications. J. F. Robot. 23(1), 3–20 (2006)

[24] Rother, C.: A new approach for vanishing point detection in architectual environments. Proc. BMVC. 20(9), 382–391 (2000)

[25] Shaw, D., Barnes, N.: Perspective Rectangle Detection. Perspect. Rectangle Detect. pp. 1–152 (2006)

[26] Yu, G., Morel, J.: A fully affine invariant image comparison method. Int. Conf. Acoust. Speech Signal Process. 26(1), 1597–1600 (2009)

[27] Yu, S.X., Zhang, H., Malik, J.: Inferring spatial layout from a single image via depth-ordered grouping. In: 2008 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work. CVPR Work. (2008)

[28] Yu, Y., Member, S.S., Huang, K., Member, S.S., Chen, W., Member, S.S., Tan, T.: A novel algorithm for view and illumination invariant image matching. IEEE Trans. Image Process. 21(1), 229–40 (2012)

[29] Zhang, L., Koch, R.: An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency. J. Vis. Commun. Image Represent. 24(7), 794–805 (2013)

[30] Zhang, Z., Wu, Y., Shan, Y., Shafer, S.: Visual panel: Virtual Mouse, Keyboard and 3D Controller with an Ordinary Piece of Paper. Proc. 2001 Work. Percetive user interfaces (2001)

"HapticVive" - A Point Contact Encounter Haptic Solution with the HTC VIVE and Baxter Robot.

Scott Devine
Queen's University
Belfast
Northern Ireland
sdevine08@qub.ac.uk

Karen Rafferty
Queen's University
Belfast
Northern Ireland
K.Rafferty@ee.qub.ac.uk

Stuart Ferguson
Queen's University
Belfast
Northern Ireland
R.Ferguson@ee.qub.ac.uk

ABSTRACT

With the release of various low cost consumer head mounted displays, such as the HTC Vive, virtual reality (VR) visualisation technology is becoming common place. However, haptic interactions continue to lag behind the visual developments. Touch feedback from the HTC Vive system is only provided by way of vibrations in the physical controllers. There are currently no large scale haptic devices that allow a user to experience force feedback in a room scale VR environment. The research presented in this paper demonstrates this problem can be addressed through the use of a large robotic arm to create an encounter haptic solution. Our haptic VR system uses the HTC Vive and the Baxter robot. Positional data is taken from the Vive controllers and sent to one of the Baxter's 7 degrees of freedom arms, which is used to provide force feedback to the user. An experiment was created where a user pushes wooden boxes off a wall in a VR environment. Several tests were performed. Different virtual boxes with a different simulated weight were simulated by varying the speed at which the Baxter moves away from the user. Results from a thirty participant user study indicate that desirable haptic effects can be achieved in a large room scale environment.

Keywords

Encounter Haptics; HTC VIVE; Baxter Robot; Haptic Feedback; Force Feedback;

1 INTRODUCTION

Within the last five years there has been extremely rapid development and production of new Virtual Reality (VR) technology, specifically with regards to Head Mounted Displays (HMDs). Devices such as the HTC Vive [1] and Oculus Rift [2], currently on the market, offer high resolutions with low latencies. These HMDs allow users to experience immersive VR environments. They are almost identical in terms of resolution and Field of View (FOV). The Vive is designed to be used in a room scale setting, where the user can walk around a VR environment within a set of pre defined bounds. The Oculus can also be used in a room scale setting, though in a smaller working area. Although both the Oculus and Vive use physical hand held controllers for interaction, a major component lacking in both systems is feedback to the sense of touch. The controllers utilise vibrations to stimulate the sense of touch, however these are limited interactions. For example, vibrations cannot render a wall or solid surface. In all simulations the controllers rendered can pass through any surface. Vibrations can only give an indication that the controller has passed through a surface.

The research in this paper proposes to solve the aforementioned limitations of the vibrational feedback by using the Vive controllers and a robotic arm. This robotic arm will provide encounter haptic force feedback to render solid surfaces to a user in VR. We use the Baxter research robot [3] to provide this feedback. Its large 7 degrees of freedom (DOF) arms afford a large working volume and has numerous safety features as standard. These include arm collision detection. The safety features allow for operation in close proximity to a user without a cage, a necessity for haptic feedback.

Providing large, room scale force feedback has yet to be achieved with recent HMDs such as the Vive. Specifically, a solution that can render solid surfaces. Improving haptic interactions helps to increase realism and immersion within VR. The HapticVive system demonstrates that this is possible, and sets the stage for future room scale force feedback solutions.

2 RELATED WORK & BACKGROUND

Haptic devices have been created to provide natural interactions between a user and a virtual environment by rendering forces to the users hands. Haptic devices help

improve the operability of tele-operation [4], allow for surgeons to be trained with realistic haptic feedback [5] and also include multi finger approaches providing feedback to more than just the whole hand [6]. Haptics devices should allow for high impedances to be rendered to the user when contact occurs with virtual objects. Also, rendering zero impedance when there is no contact with the environment, thus, rendering the device as transparent as possible. Unfortunately, zero impedance is nearly impossible because the user is always tethered to the haptic device. Achieving a negligible impedance requires that the haptic device has low inertia motors with low gear ratios. However, these may impact force feedback levels. Force control theories, such as impedance or admittance concepts, can also be used to help reduce the impedance in free space [7, 8]. Their reductions are however limited.

Encounter-type haptic devices have been created as a solution to the inability of common haptic devices to render zero impedance in free space, and their lack of realistic transition between free space and contact. Encounter-type devices only collide with the users hand and provide forces if a collision occurs in the virtual environment [9, 10, 11]. When in free space, the encounter-type end-effector is mechanically detached from the user and precisely tracks the position of their hand, granting zero impedance in free space. Encounter-type devices have a range of sizes, or a range of working volumes. The smallest of these employed a 2D circular finger tip encounter haptic interface [12], allowing for feedback around the end of the finger tip. There are a number of large and room scale encounter haptic interfaces for rendering a control panel, juggling simulation and large surface exploration for the palm of the hand [13, 14, 15]. However, none of these incorporate a room scale HMD.

In essence, encounter-type haptic devices are robotic arms. The Baxter robot is a robot made by Rethink Robotics and is comprised of a torso, two 7-DOF arms and a head. The torso is mounted on a wheeled trolley with each arm connected to the torso via an arm mount. Each arm consists of seven rotational joints with the end effector containing a 640 x 400 camera and infrared range detection. The head of the Baxter is represented by a screen attached to the top of the torso also including ultrasonic range detection. Positional accuracy is quoted as ± 5 mm [3]. The Baxter Robot is designed with safety in mind. It has a number of features that allow it to operate in close proximity to humans, without a cage. These features include non-locking joints and sensors in each of the joints to make the robot aware of collisions. The size of the robotic arm, and therefore the working volume along with the safety features make it suitable to act as the force component in an encounter haptics system.

Figure 1: Example of user interacting with the Baxter.

As encounter haptic devices cannot rely on the haptic interface for tracking, they rely on external forms of tracking. The 2D circular finger tip device mentioned previously uses infrared proximity sensors [12]. Filippeschi [16] used the Leap Motion optical tracking device for the tracking element in a seated, non-HMD system for use in medical remote palpitation. The Leap also had the added benefit of increasing tracking and working volume. Infrared camera solutions are also employed in non-HMD roomscale solutions [14, 15], using seven and six VICON MX Motion Capture cameras.

The Vive features five hardware components, a HMD, two base stations and two controllers. The headset contains two screens and has a resolution of 2160x1200 (1080x1200 per eye) and has a refresh rate of 90hz [1]. The base stations are mounted at opposite corners of a workspace up to 4.6m^2, they sweep structured light lasers across the workspace, which are picked up by sensors in the headset and controllers. The vive control system is extremely fast with worst-case latency for head tracking data around 4.444ms (250hz), and worst-case latency for controller tracking data at 4.0ms (250hz). The lighthouse system of tracking is also very accurate, with an RMS accuracy of 1.9mm [17]. This makes the system suitable for encounter haptics, it has both low latency and high accuracy.

There is a real lack of force feedback devices for current generation VR devices. Since many additions to both the Oculus and Vive rely on vibrations. To the best of the authors' knowledge, there are no large scale encounter haptic interfaces for use with HMDs. With the success of the HTC Vive and that 82% of owners use room scale [18], there is motivation to develop room scale haptic solutions. Current haptic devices, such as the well known Phantom device [19], are not suitable for this task due to the small working volume. An encounter type interface comes into its own in a room scale environment and would help to enhance user experience.

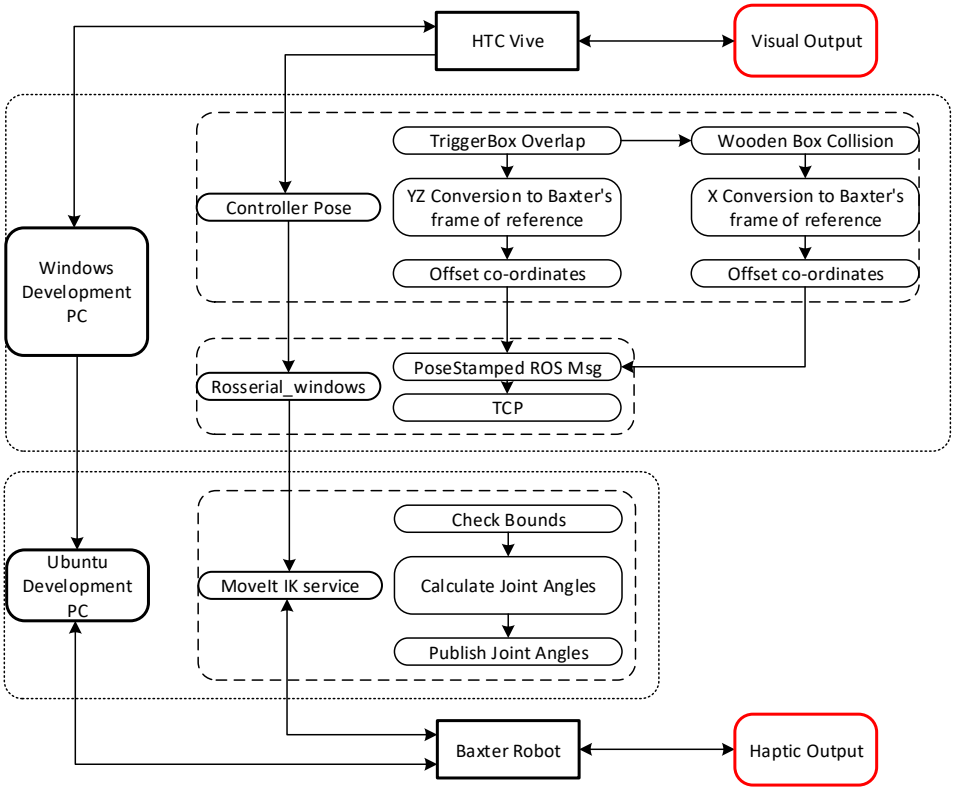


Figure 2: Flow diagram for the HapticVive System. Coordinates pass from the HTC Vive into the Windows PC. Then onto the Ubuntu PC through a TCP Connection. Finally they are sent to the Baxter through ROS.

3 ENCOUNTER HAPTICS SOLUTION

The system is made up from two development PCs, an HTC Vive HMD and a Baxter Robot. The Vive HMD provides the visual component. The Vive's physical controllers provide positional data of a user's hand and the Baxter's robotic arm provides force feedback when required. The computation is split into two halves: one Windows development machine (Windows 10, i7 4790k, GTX 970) is used to run the virtual environment and Vive HMD. The other Ubuntu development machine is used to run the ROS (Robot Operating System) half of the system for the Baxter robot, which can be seen in Figure 2. The Ubuntu PC is running Ubuntu 14.04 and ROS Indigo. ROS Windows header files are used to allow for the use of ROS message types on the Windows development machine. Coordinates are loaded into a PoseStamped ROS message, so the coordinates can be sent between development machines using a TCP connection. A socket node along with *rosserial server* is used on the Ubuntu side to allow for communication. This is currently a one way communication. The virtual environment was created in Unreal Engine, primarily used for games design, it allows for quick drag and drop creation with native VR HMD plugins. The VR environment is shown in Figure 3.

The Vive controllers are tracked in 6 DOF within the same work space as the HMD. On the Windows development machine, the Unreal Engine SteamVR plugin can query for the position and orientation of the controller. This data is used to render a model of the controller in VR.

When the controller enters a trigger box, used within Unreal Engine to trigger events, coordinates are sent to the robot. As the controller is therefore in close proximity to a point of interaction. Sending the coordinate data of the controllers when they enter a trigger box is a preemptive measure, as the system is not solely relying on collision data. Were it to rely on only collision data and the robot was a considerable distance from the interaction site, it would not reach the user in time to provide feedback. The robot is always aligned with the controller preempting a collision, provided it is inside a trigger box.

As shown in Figure 2, the pose of the controller is converted onto Baxter's frame of reference, in two parts. Firstly, when the controller is in the trigger box the system converts and sends the YZ portion of the pose, left, right, up, down when facing the Baxter. The Baxter's arm can therefore move along a plane in front of the controller. When a collision occurs in VR between the rendered controller mesh and piece of geometry, the X

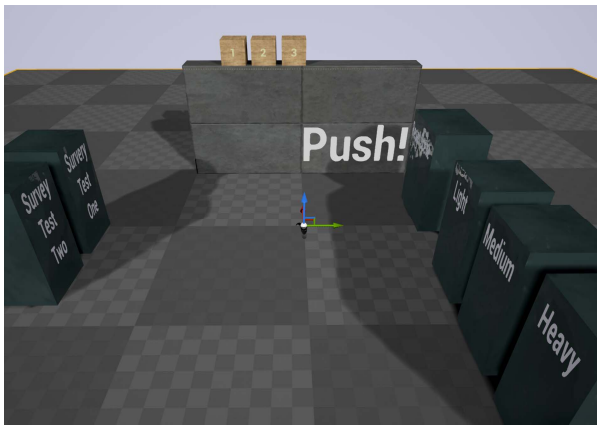


Figure 3: View of the VR environment inside the HMD.

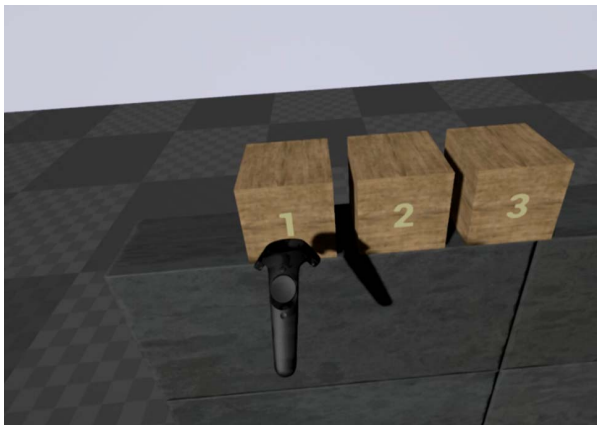


Figure 4: Users perspective inside the HMD.

coordinate of the pose is converted and sent. This is the axis moving in and out toward the user. This allows the Baxter to move in three DOF, and provide force feedback. In order for the Baxter to reach a desired position when the controller is inside a trigger box the Kinematics and Dynamics Library (KDL) inverse kinematics solver is used to calculate joint angles. This is accessed through the Moveit! [20] motion planning framework.

When the Baxter's arm has reached its desired position and an interaction is required, the user needs a physical surface to interact with. A custom end effector is screwed into Baxter's standard electric gripper, a picture of end effector can be seen in Figure 1. The end effector is a sheet of 150*mm* by 100*mm* plywood.

In positional mode the Baxter's speed can be adjusted in a ratio form from 0.1 to 1.0, the larger the number the greater the speed. In order to simulate different weights to the user the speed was altered dependant on the interaction geometry. Within the environment there are various pillars (Figure 3), which are used for functionality within the system, such as resetting the scene, changing condition and the speed of the Baxters arm. The pillars allow for the participant to stay in VR without having to take the HMD off. Allowing for multiple aspects of the system to be tested. A participant places the controller inside the pillar to trigger its functionality.

During initial testing it was observed that based on the inverse kinematics (IK) solution, the Baxters arm speed would slow down when in certain positions. For example, the Baxter has collision detection enabled on its arms and torso, so it cannot hit itself. When confronted with tight, curled in towards itself arm positions, the arms speed slows down, as the arm nears a collision with either itself or the torso of the robot. This has the effect of making items within the VR environment feel heavier, as they take longer to push away, or stop you from pushing them away altogether.

3.1 User Study

30 participants took part in the study. 70% were male with the remaining 30% female. The average age was 35.4 years. 40% of participants taking part in the study answered that they had used a HMD before, and 33% said that they understood the term haptics. Participants of the study were required to be fully mobile, as the study required them to be standing in a room scale environment, and move around in the workspace. The study required the use of their left hand and arm, participants were excluded from the study if they had any minor motor problems. If participants suffered from any illnesses such as epilepsy they were not allowed to take part in the study. During the study no participants were excluded based on the criteria.

The user study was undertaken in order to evaluate the system, in order to measure how well it provided operator feedback. We also wanted to analysis how the Baxter performed in this unorthodox role. The study contained two conditions: For each condition participants were asked to push sets of three boxes off a wall of the virtual environment. As the environment is on a one to one scale with the real world, the wall is one metre tall and the boxes were $0.2m^3$. Using the adjustments in speed of the Baxters arm, we simulated different weights of boxes. For the initial control condition, all the boxes were simulated to weigh the same. In the second condition the boxes got progressively heavier. In the first control condition the arm is set to move at its maximum speed of 1.0, for all three boxes. In the second condition the first box is set at 1.0, the second 0.5, and the third 0.1.

It was explained to the participants that the controller would look the same in VR as it does in the physical world, as the mesh being used to render the controller in VR was an official Vive asset. Time was given for the participant to get used to the HMD and being in the VR environment. Once they were comfortable with the HMD and the environment they were asked to walk forward towards the wall in the environment (Figure 3)

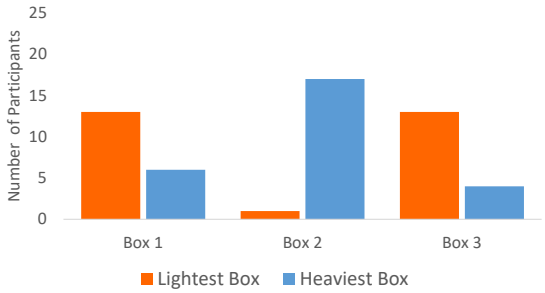


Figure 5: Condition One Results.

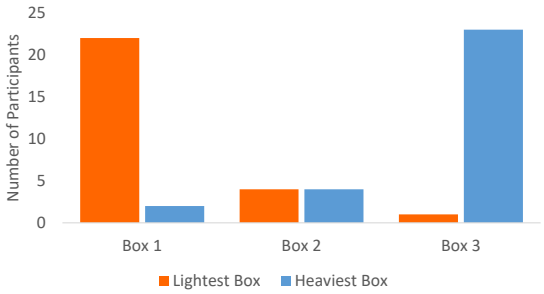


Figure 6: Condition Two Results.

and hold the controller a few centimetres in front of box number one (Figure 4). Participants were asked to slowly push each of the boxes off the wall in order, from one to three. With emphasis made to not attempt to ram the controller through the box, like a punching motion. Upon completion of the first control condition the participant was asked which box felt the lightest and which felt the heaviest. The participant then reset the scene themselves using the pillars placed in the scene (Figure 3). They then completed condition two by pushing the three boxes off the wall in order, and were again asked to note which felt lightest and which the heaviest.

4 RESULTS AND DISCUSSION

The study was developed to gauge the effect the Baxter had on the feedback, in different positions and whether these effects were noticeable to participants.

Dependant measures were analysed in separate one-factor ANOVAs with two levels (Condition: One (No speed variance), Two (Speed Variance)). Comparisons across the levels were made using 95% Confidence Intervals (CI's) generated from the ANOVAs [21]. Both dependant variables displayed a significant effect of condition. Heaviest Box: $F(1,58) = 19.13, p < .001$. Lightest Box: $F(1,58) = 8.97, p = .004$.

The expected outcome was that the participant was not sure which box was heaviest within the first condition. This is because the robot moved away at the same speed for all three boxes and therefore should of felt the same weight to participant. This was in fact not the case for condition one (Figure 5). 56.7% of participants indicated box two felt the heaviest. For the lightest box in condition one 43.3% of participants indicated box one and 43.3% also indicated box three felt the lightest. This is almost a perfect spilt, only one (3.3%) participant felt box two was the lightest, with the remaining 10.0% unsure.

This result is due to the Baxter: The boxes were set up in a manner that had a distinct space between them, they were also quite large, $0.2m^3$. The total distance for the length of the working volume across the wall was close to the overall working width of one of the Baxter's arm's. However, when the robot was presented with moving backwards towards itself, in this case for for the second box, the IK solution curled the arm toward the torso of the robot. When presented with these tight joint angles, the Baxter moves more slowly. When the Baxter moves slowly whilst a participant is pushing a box off the wall, the box feels heavier. The use of IK constraints could limit these effects. Anecdotal observations suggest that over elbow positioning would be preferred to under elbow positioning. Where an over elbow position curls the arm from a high elbow point, down and out away from the robot.

When the speeds of the arm are varied in condition two, across the boxes, the results indicate that the participants can pick out the lightest vs the heaviest. With 73.3% and 76.7% picking correctly respectfully. This is shown in Figure 6.

The arm is moving at 10% of its original speed on the third box, this change is dramatic enough for the user to pick it as the heaviest box. This speed is also lower than the reduction in speed that the Baxter creates when its arm in curling in toward its own torso. The arm position therefore has no effect on heavy objects.

As the Vive controllers are not flat front ended, they angle down from the top and angle up from the bottom to meet in the middle at a point. This means there are different ways in which a participant can interact with the wood on the end of the Baxter. If they angle the controller down, they will meet the wood with a flat part of the controller, allowing them to more easily push against the robot. This became apparent with some participants as the controller slid down the wood when pushing the end effector. Some participants tend to push down and forward. A few went as far to completely push down off the wooden plate. This was also an issue when a number of participants attempted to ram through the boxes almost like a punching simulator although they were instructed to slowly push through the boxes.

Although using trigger boxes as preemptive measure to make sure the robot was always aligned with the controller. When a participant moved the controller between trigger boxes, no coordinate data was sent. This left the robot at the previous interaction site. Which oc-

casionally meant the participant had to briefly wait for the robot. Using ray traces from the controller to geometry in the scene is a possible solution to this problem.

Smaller variations in the speed of Baxter's arm may have yielded more beneficial results, along with a greater number of variations. Some participants struggled to push the heaviest box off the wall, with some females opting to use two hands on the one controller when the box stopped moving. The motors in the arm of the Baxter can create a large amount of noise when it moves. Participants were therefore asked about the noise of the Baxter robot moving, whether this was off putting or gave anything away. Most participants were focused on what was presented to them visually and were not influenced or distracted by the noise.

The Vive controllers already include vibration for small interactions, but this breaks down for solid surfaces. The point contact system presented in this work is simple yet effective at bridging this large gap. It also caters to an extremely large working volume, making it ideal for the direction VR applications are going. The Baxter brings some down sides to the effectiveness, in relation to the varying forces that are dependant on arm configuration and orientation, this can be attributed to both its safety features and the simple IK solution employed.

5 CONCLUSIONS

The work reported in this paper has demonstrated that a large robotic arm can be used together with a HMD to augment the virtual reality experience by providing haptic force feedback, and do so in a safe manner. The Baxter is inherently a safer robot, not requiring a cage. However, these safety features come at the price of increased movement latency and wider positioning tolerance. Results from the user study indicate participants can identify changes in weight of virtual objects by changes in speed of the Baxter's arm. When presented with tight joint angles the Baxter's collision detection slows arm speed. Giving the impression that virtual objects are heavier than intended. Future refinement would hope to mitigate the effect on the user of some of these issues, including the correlation of Baxter speed ratios with real world weight and the use of the second arm to create a larger working volume. A lip around the edge of the interaction plate could also be included so the controller cannot slide off the interaction site. A more complex solution with active feedback where the robot pushes toward the participant would help the system be more effective, and provide more uses.

6 REFERENCES

[1] HTC Corporation, 2016. [Online]. Available: https://www.htcvive.com/uk/

[2] OCULUS Inc, 2015. [Online]. Available: https://www.oculus.com/en-us/

[3] RethinkRobotics, "Baxter specifications," 2015. [Online]. Available: http://sdk.rethinkrobotics.com/wiki/Hardware_Specifications

[4] D. Kim, K. W. Oh, D. Hong, J.-H. Park, and S.-H. Hong, "Remote control of excavator with designed haptic device," in *Control, Automation and Systems, 2008. ICCAS 2008. International Conference on*, Oct 2008, pp. 1830–1834.

[5] M. Mortimer, B. Horan, and A. Stojcevski, "4 degree-of-freedom haptic device for surgical simulation," in *2014 World Automation Congress (WAC)*, Aug 2014, pp. 735–740.

[6] H. Kawasaki, J. Takai, Y. Tanaka, C. Mrad, and T. Mouri, "Control of multi-fingered haptic interface opposite to human hand," in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 3, Oct 2003, pp. 2707–2712 vol.3.

[7] K. Vlachos and E. Papadopoulos, "Using force control for fidelity in low-force medical haptic simulators," in *IEEE Conference on Computer Aided Control System Design*, 2006, pp. 181–186.

[8] N. Bernstein, D. Lawrence, and L. Pao, "Dynamics modeling for parallel haptic interfaces with force sensing and control," *IEEE Transactions on Haptics*, vol. 6, no. 4, pp. 429–439, Oct 2013.

[9] W. A. McNeely, "Robotic graphics: a new approach to force feedback for virtual reality," in *Virtual Reality Annual International Symposium, 1993., 1993 IEEE*, Sep 1993, pp. 336–341.

[10] Y. Yokokohji, R. L. Hollis, and T. Kanade, "What you can see is what you can feel-development of a visual/haptic interface to virtual environment," in *Virtual Reality Annual International Symposium, 1996., Proceedings of the IEEE 1996*, Mar 1996, pp. 46–53, 265.

[11] J. K. Salisbury and M. A. Srinivasan, "Phantom-based haptic interaction with virtual objects," *IEEE Computer Graphics and Applications*, vol. 17, no. 5, pp. 6–10, Sept 1997.

[12] F. Gonzalez, W. Bachta, and F. Gosselin, "Smooth transition-based control of encounter-type haptic devices," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 291–297.

[13] M. Yafune and Y. Yokokohji, "Haptically rendering different switches arranged on a virtual control panel by using an encountered-type haptic device," in *World Haptics Conference (WHC), 2011 IEEE*, June 2011, pp. 551–556.

[14] P. Tripicchio, E. Ruffaldi, C. A. Avizzano, and M. Bergamasco, "Control strategies and perception effects in co-located and large workspace dynamical encountered haptics," in *EuroHaptics*

conference, 2009, March 2009, pp. 63–68.

[15] E. Ruffaldi, C. A. Avizzano, P. Tripicchio, A. Frisoli, and M. Bergamasco, "Surface perception in a large workspace encounter interface," in *RO-MAN 2008 - The 17th IEEE International Symposium on Robot and Human Interactive Communication*, Aug 2008, pp. 21–26.

[16] A. Filippeschi, F. Brizzi, E. Ruffaldi, J. M. Jacinto, and C. A. Avizzano, "Encountered-type haptic interface for virtual interaction with real objects based on implicit surface haptic rendering for remote palpation," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, Sept 2015, pp. 5904–5909.

[17] O. Kreylos, "Lighthouse tracking," 2016. [Online]. Available: http://doc-ok.org/?p=1478

[18] Steam, "Vive usage stats," 2016. [Online]. Available: http://steamcommunity.com/app/358720/discussions/0/350532536103514259/?ctp=2

[19] Geomagic, "Geomagic touch," 2016. [Online]. Available: http://www.geomagic.com/en/products/phantom-omni/overview

[20] I. A. Sucan and S. Chitta, "Moveit!" 2017. [Online]. Available: http://moveit.ros.org/

[21] J. Jarmasz and J. G. Hollands, "Confidence intervals in repeated-measures designs: The number of observations principle." *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, vol. 63, no. 2, p. 124, 2009.

Intelligent Clothing Size and Fit Recommendations based on Human Model Customisation Technology

Li, Runze, Zhou, Yangping, Zhu, Shuaiyin, Mok, P.Y*.
Institute of Textiles and Clothing, The Hong Kong Polytechnic University
Hunghom, Hong Kong.
*tracy.mok@polyu.edu.hk

ABSTRACT

E-commerce and mobile commerce have become part of our lives; the growth of online sales has continuously outpaced that of through traditional retail channels. However, even though nearly everyone now shops online, it is still challenging to sell clothes online. One of the major reasons is that size selection and clothing fit are still difficult to address within the current online platforms; this is reflected by the alarming return rate, between 20% and 40%, of online purchased clothes. New technology recently becomes available to create accurate 3D human models for individual customers by taking two photos using smart phone. The resulting models have accurate size and detailed local shape characteristics. The accuracy of the resulting models is comparable to scan, meeting the size tolerance of the clothing industry. Extending from this photo-based precise human modelling technology, we propose new solutions in this paper to advise suitable clothing sizes for individual customers, and to visualise virtual try-on effects of clothing on customised human models for online fit and size evaluation. We verify the size and fit recommendation solutions by experiment on clothing items of dress and shirt in this paper. With these novel sizing and fitting solutions, fashion brands can better serve their customers by attracting, engaging and delighting them with a brand new shopping experience.

Keywords

Human modelling, virtual try-on, clothing simulation, size and fit evaluation

1. INTRODUCTION

During the past five to ten years, e-commerce and mobile commerce have become part of our lives. According to China Internet Network Information Centre (CNNICC) [1], clothing, shoes and hats are among the top five product categories that people shop online. Despite the large online sale figures, selling clothes online is still challenging because clothing fit is difficult to address. It is not uncommon that people find the clothing products ordered online do not fit them at all when orders are received, and they have to arrange for exchange or return. Research studies [2,3] reported that most complaints about online clothing purchases are related to size and fit, and 'fit' is one of the major reasons for purchase return. In addition, uncertainty of clothing fit not only causes high return rate, but also affects customers' overall satisfactions towards the e-stores/brands, leading to loss of sales. It is thus important to research on new solutions for online fitting suggestions and size recommendations.

Recent developments in computer graphics and virtual reality have brought about new possible solutions to facilitate fit analysis in online shopping, that is, virtual garment fitting on human models. Existing one fitting processes usually start from creating avatars using subject's body measurements, followed by a number of interactive 3D operations for selecting and positioning clothing pattern pieces around avatars and virtual sewing in order to generate the final virtual try-on results. Virtual try-on applications enable users to visualise the wearing effects without physically trying on garments, improving shopping efficiency [4] and reducing product return rates [5].

The research work of virtual try-on are mainly focused on better modelling and rendering the time-varying geometrical and mechanical behaviours of clothing and its interaction with human models. Geometry-based simulation and physics-based simulation are the two categories of methods in virtual try-on or clothing simulation. Although previous research efforts have successfully modelled the clothing models in accordance with the physical laws and ensured attractive rendering results, virtual fit evaluation is still difficult to address. One of the

main reasons is that human shapes vary substantially, while the existing try-on simulation are not on individual shapes, but avatars with average figure shape. We propose in paper a complete pipeline that integrates virtual fitting and human model customisation technologies to realize fit and size recommendation, facilitating online shopping of clothes. The rest of the paper is organised as follows. We first review the related work in virtual simulation and human modelling technologies in Section 2. We next explain our proposed pipeline in Section 3. We conduct experiment in Section 4 to demonstrate the proposed pipeline by trying on a dress model onto three female subjects, and trying on a shirt model onto three male subjects. After physical draping, collision detection, response and rendering, the result provides visual references for fitting analysis on different figures or body shapes.

2. RELATED WORK

Virtual fitting solution has aroused much research attentions in recent years. In general, a typical virtual fitting system consists of three subsystems, namely, body-shape modelling system, clothing simulation system, and interactive 3D exhibition room [6]. Fitting a 3D garment model onto an individual human model requires a perfect match between the two models and a minimal distortion of the garment model. Therefore, typical virtual fitting solutions must address to two problems [7]: garment simulation and human modelling technologies.

2.1 Clothing Simulation Technology

Garment simulation or draping means computing the shape of a piece of cloth under the control of gravity [8]. Traditional virtual try-on simulations have two approaches: either sewing 2D patterns around a 3D character [20,21] or fitting 3D garment models directly onto a 3D human model [7,22]. Comparing with other 3D simulation, garment simulation is unique and challenging for two reasons: garment fabric is soft and human body is usually in movements.

Normally, a garment on human body can be classified into fit part and fashion part [9,10]: the former is in direct contact with the human model and the latter is draped freely to create aesthetic appearance. The fit part can be directly determined by semantic features on a body model [11], while the fashion part is simulated according to the physical properties of garment materials.

Before the pioneer work of Prof. Nadia Magnenat-Thalmann, little has been done on realistic garment simulation [12,13]. Only simple scenes such as blowing flag could be simulated. In the last two decades, many efficient simulation models were proposed, with which customers can visualise the realism of virtual try-on [8].

Garment simulation methods can be divided into two classes − geometric based and physical based methods. Geometric methods describe complex geometric details by mathematical models, such as wrinkles and folds [14]. It treats garment as developable surfaces. Developable surface means that the materials cannot be stretched or torn during the process of virtual sewing and simulation. Considering that most textiles have little stretch in physics, the geometric model could indeed achieve acceptable level of accuracy. On the other hand, in physical based simulation, the mechanical properties of fabrics are calculated, leading to higher level of simulation accuracy. However, its computing speed is much slower than the geometric methods. Many physical models were proposed, such as the mass-spring models [15,16], the particle models [17] and the elasticity-based models. To take advantages of the merits of both geometric methods and physical based method, to be specific, the speed of the geometrical methods and the accuracy of physical methods, some researchers proposed hybrid models [18,19] for clothing simulation.

2.2 Human Modelling Technology

3D human body modelling technology has been widely studied by engineers and researchers in computer graphics area. Body-scanning technology enables each individual to be the centre of garment development and enhances experience in size and fit [23]. In most cases, 3D body scanners capture subject's body surfaces using optical techniques [24]. Body scanning has been applied in sizing and mass customisation. Scan-based methods have known drawbacks such as high cost and privacy concerns [25,26]. Other fitting tools usually build virtual avatars based on users' body measurement input [27]. However, inaccurate body measurements are often input by general consumers [28]. Moreover, measurement-driven deformable models are often established by example-based methods, which create realistic average figures, but cannot capture detailed local shape characteristics of individuals.

In the age of smartphones, It is appealing if accurate 3D models of individuals can be created by taking photos. To this end, various image-based reconstruction methods were proposed. Traditional image-based reconstructions require a set of images to be simultaneously taken for body dimensions analysis. Moreover, subjects are asked to take photos against a mono-color background. For example in [29], two photographs were taken and verified, after which 36 landmarks were identified manually to define for collecting body dimensions. However,

model generation process is time consuming in the method of [29]. [25] used a segmentation method to accelerate extraction of 2D body contours from front-view and right-view photographs. However, there are known shape approximation errors in their method of constructing 3D body shape.

Recently, a new model customisation method was recently developed by segmenting raw contours of the human subject from images, from which to construct 3D shape representation for the subject using trained relationships between 2D profiles and 3D shape representation, and the shape representation guides the detailed shape deformation for subjects dressed in tight-fit clothing [26], in arbitrary clothing, or even loose-fit clothing [30]. A subject's 3D body model can be customised in a few seconds, and the output models have accurate sizes and realistic shape details. However, their method does require some manual work in image feature definitions.

3. SOLUTION TO THE PROBLEM

This paper proposes a virtual clothing simulation system that integrates human model customisation technology with virtual try-on simulation, as shown in Figure 1.

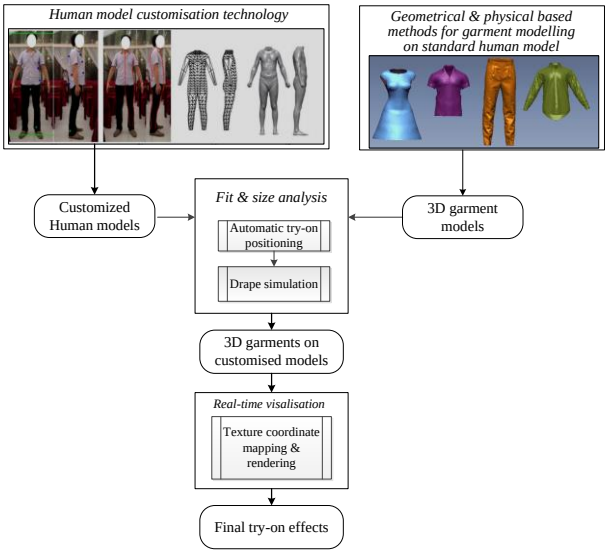


Figure 1. Virtual fitting flow chart

We will first describe briefly the human model customisation technology in section 3.1, which is based on our previous work [26,30]. Next, we briefly discuss two approaches to obtain 3D garment models in section 3.2. The key contribution of this paper lies in the part fit and size analysis by automatically trying on 3D garment models onto different customised human models and carrying out drape simulations. For simplicity, virtual garments are simulated using particle system model. Although particle model-based simulation may have deviation on the final try-on effects because of the simplification in the simulation computation, our method does visualise the garment and body relationship before the simulation. The geometrical relationship better illustrates the clothing 'fit'. We demonstrate that this approach can be used for efficient near real-time size recommendations and fit evaluation. After the drape simulation, the final garment shapes on customised models are shown to customers. By pre-computing texture coordinate mapping, users can be shown with the rendering results of different colours and texture choices on the final garments in real time.

3.1 Human Model Customisation

Although previous studies have attempted to use virtual try-on to analyse fit, most previous studies of clothing simulation use standard virtual dummy instead of customised human models for the simulation. One of the reasons is that clothing simulation is computational expensive, it is difficult, if not impossible, to simulate the try-on effects in real time when a clothing is selected. All simulations must be done beforehand and the final simulation results are recorded at the database for visualisation. Another reason, more critical one, is that most existing human modelling methods cannot create a very accurate body shape model for individual customers.

We proposed to apply the human model customisation method [26,30] for online virtual try-on and fit evaluation. We propose complete automatic pipeline to obtain accurate measurements and have 3D body models customised using two photographs. The human modelling method has the following advantages: (1) customised models have high resolution details and realistic appearance; (2) customised models are very accurate (in terms of sizes), comparable to scan; (3) the modelling process is efficient, meeting the requirement on real-time application; (4) the user involvement is simple and easy to handle, just by taking two photographs without restrictive clothing conditions.

Our method create accurate body models under clothing based on customer's photos. Photos only contain 2D information, we define customer's 2D feature as body profiles – front-view and side-view profiles. Since most parts of body profile are covered by clothing in the photos, the most challenging task is to predict a complete under-the-clothes profile based on some cues not being covered by clothing.

To do so, we first establish a database of normalized body profiles. The profiles are extracted from scans of human subjects with different body shapes. The database covers a wide range of body shapes; some example profiles are shown in Figure 2.

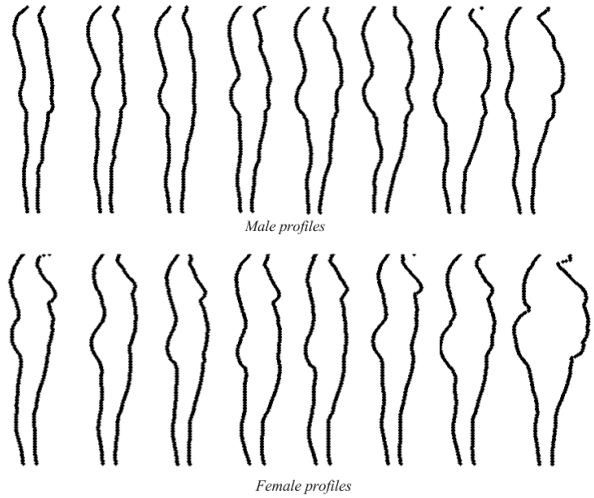


Figure 2. Example profiles in the profile database

We used the state-of-the-art semantic segmentation technology [31] to identify the human from the image background by training a fully convolution network (FCN), as shown in Figure 3. We then synthesise the under-the-clothes body profiles (Figure 4) of the subject with patented method [32] by taking advantage of the large profile database.

Figure 3. Human parsing

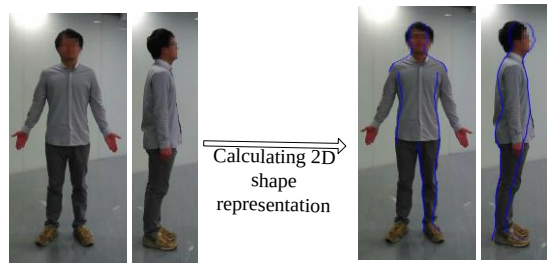


Figure 4. Body profile calculation

With the 2D profiles, we can use them to predict and construct 3D shape representation for the subject in the photos. To do so, we identify a number of body features (such as chest, waist, hips, crotch, shoulder, etc.) from the synthesise profiles using anthropometric knowledge. The corresponding width and depth of each body features are obtained and input to relationship models trained between 2D features and 3D shape, so as to generate the cross-sectional 3D shape at these feature levels. The predicted cross-sectional 3D shapes are used to reconstruct a large mesh model as a 3D shape representation of the subject (Figure 5). It is important to note that our 3D shape representation is feature aligned mesh, consist of parallel cross-sectional features. Each layer represents a cross-sectional shape of the subject's body corresponding the girth of to a feature. By defining the body shape in parallel cross-sections, we characterize different body figure types. The relative positions of these cross-sectional body features in fact describe the overall shape of the body, so they define the body model's global features. A layered structure with parallel cross-sections is an effective shape representation, because such a structure aligns with the clothing size definition. The parallel cross-sections characterize local features of the body in terms of shape and size, as some cross-sections indeed correspond to important body girth measurements, such as the bust, waist, and hip. The shape of these cross-sections gives detailed information on where the body has developed fat, for instance the shape of the waist girth. With these detailed shape representation, we can deform detailed 3D models by deforming a template using deformation algorithm [26,30].

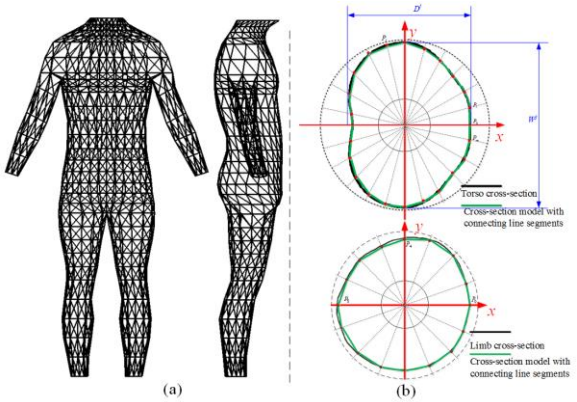


Figure 5. 3D shape representation

3.2 Garment Modelling

We used both two methods [33,34] to generate 3D garment model from 2D pattern pieces. Our system first imports pattern pieces in dxf format, and triangulates the 2D pattern pieces as mesh models by the constrained delaunay method (see Figure 6(a)).

3.2.1 Physical-based simulation model

Traditionally, 2D pattern pieces are place around human model in 3D space, and simulate garment try-on effect by physical-based method [33], as shown in the example of Figure 6(b). We used particle system to simulate clothing's physical properties, in which each vertex of the cloth mesh is treated as a particle, and its positions and velocities are updated under

different forces acting on the particle. The particle system can simulate highly deformable objects like clothing. A particle system mainly employs Newton's second law,

$$F(t) = M\frac{d^2P}{dt^2} \qquad (1)$$

where the vector $\boldsymbol{P}$ represents the particle position, $\boldsymbol{M}$ denotes the geometric state and mass distribution of the cloth, $\boldsymbol{F}$ is the sum of the forces (air-drag, contact and constraint forces, internal damping, and so forth) exerted to the particle.

Combining all forces into a force vector f_i, the acceleration a_i of the ith particle is simply defined by $a_i = \frac{f_i}{m_i}$, where m_i is the ith particle's mass. Given the known position $\boldsymbol{P}(t_0)$ and velocity $\boldsymbol{v}(t_0)$ of the system at time t_0, our goal is to determine a new position at each time step.

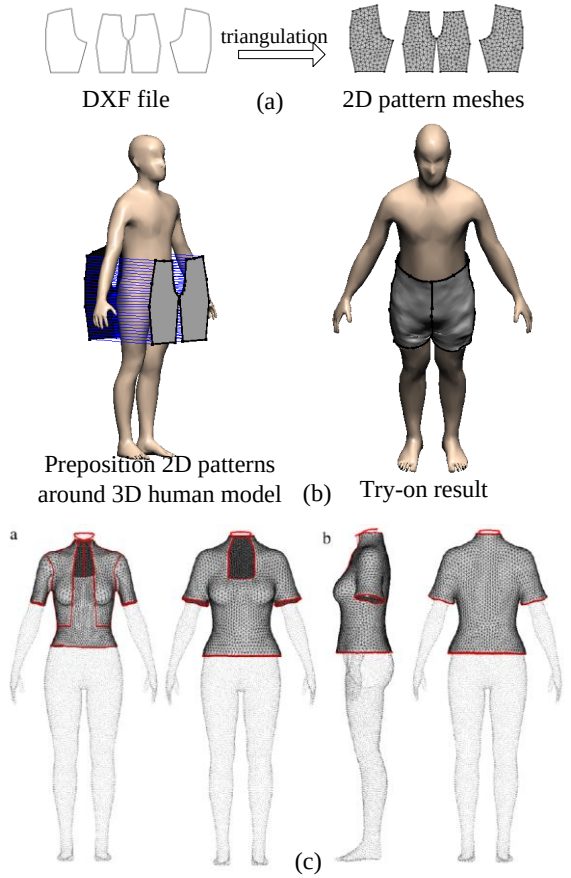


Figure 6 (a). Triangulation of pattern pieces; (b) modelling 3D garment by physical-based simulation [33]; (c) geometrical garment modelling method [34]

Physical-based simulation may generate slightly different garment models each time. One of the reasons is that pattern pieces are manually arranged around the human model before the simulation in most physical-based simulation. The preposition of 2D pattern pieces in relation to the 3D human model determines the final appearance of garment. In general, pattern pieces should be arranged as close as possible for better results. The output 3D garments will have folds and gathers formed in the physical drape simulation.

Another approach to generate 3D garments is by geometrical-based surface parameterisation [34], involving two steps (see Figure 6(c)).

Step one is to obtain an initial form of 3D garment and correspondence mapping relationship between human model of clothing. This is done by first defining corresponding feature points on both the human mesh model and the clothing pattern pieces. Next, the human mesh model is segmented to several patches to match with the patterns. Finally, a cross parameterization procedure is carried out by duplex mapping scheme.

Step two is to restore the desired shape and size of the 3D garment from its initial form. Iterative hybrid pop-up method is employed to deform the initial garment. First, the silhouette of initial form of garment (i.e. girth measurement at the hem level) is adjusted to the size of corresponding pattern pieces. A physical simulation is performed along the normal direction to update the garment length. As a result, the position of the boundary vertices is changed. Finally, the rest vertices are updated by the geometrical reconstruction with the use of pyramid coordinates. The pyramid coordinate of vertex v^{3d} is calculated as weight parameter ω_i^{3d} by:

$$\omega_i^{3d} = \omega_i^{2d}(1-\cos\theta_i) \qquad (2)$$

where ω_i^{2d} is the mean value coordinates weight, it is defined by its neighbour angle α_i and length l_i as follows:

$$\omega_i^{2d} = \frac{\tan(\alpha_i/2)+\tan(\alpha_{i+1}/2)}{l_i} \qquad (3)$$

where $l_i = |v^{2d} - v_i^{2d}|$.

$$\theta_i = (\arcsin\frac{l}{|v^{3d}-v_i^{3d}|} + \arcsin\frac{l}{|v^{2d}-v_i^{2d}|})/2 \qquad (4)$$

where $|v^{3d}-v_i^{3d}|$ is the length between v^{3d} and its neighbour v_i^{3d} on the 3D shape of the current reconstruction iteration, $|v^{2d}-v_i^{2d}|$ is the corresponding length on the 2D pattern, l is the length parameter value for the local coordinate of the current reconstruction iteration.

Geometrical models obtained by this approach are with smooth garment surfaces. We later obtain drape simulation by a numerical method, which involves Non-linear CG integrator and can solve particles' position and velocity at each time step due to gravity, stretch force, bend forces, and vertex/face and edge/edge collisions.

In this paper, we use the 3D shape representation constructed in the stage of human model customisation for automatic preposition of garment models. Each 3D shape representation has all feature positions known, which can be used to define the relative position of garment models.

4. EXPERIMENT

A garment try-on experiment was conducted involving three male subjects and three female subjects in The Hong Kong Polytechnic University. Firstly, we prepared 3D garment models of dress and polo shirt, which were not draped on human models, but could represent basic geometrical 3D clothing shape. We used the described human modelling method to obtain 3D customised human models for the three males and three females. All subjects were taken front-view and side-view photographs for model customisations.

4.1 Result

To illustrate experimental results, the following part will show the key steps in Figures 7 to 10. In these figures, garment model and customised human models are inputs to the system, the preposition and drape results are shown in blue, and the rendered results are presented in red.

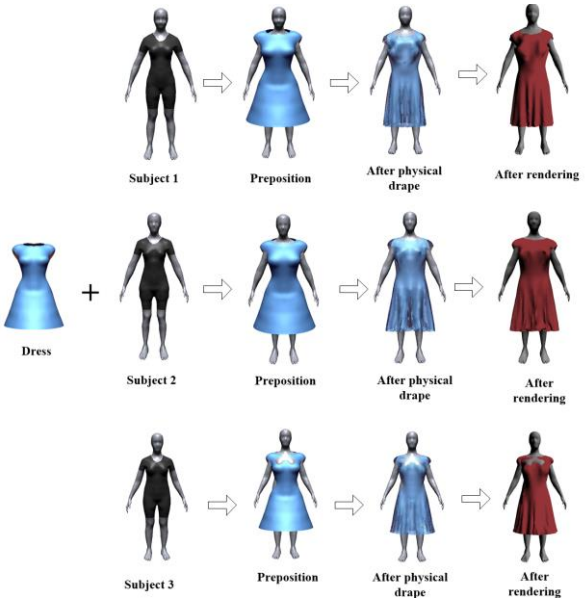


Figure 7. Dress try-on customised female models

Figure 7 demonstrates the results of virtual try-on dress experiment. After obtaining pictures and height information from three female subjects, their customised human models were created. From the prepositioned results where garment model are mapped to different human models automatically, we can found that the dress is fit for subject 2, but too large for subject 1, and relatively too small at bust for subject 3.

We compared the (a) before drape and (b) after drape results of dress on subject 2 in Figure 8. It is shown that the dress is rigid and relatively stiff before drape, and the dress has folds and wrinkles after drape.

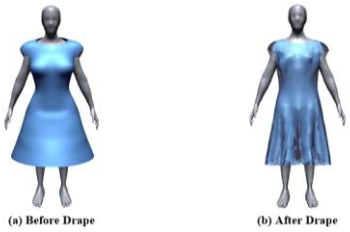


Figure 8. Subject 2's dress wearing effects (a) before drape and (b) after drape simulations.

The final rendered try-on simulation effects for subjects 1, 2 and 3 are shown in Figure 9.

Figure 9. Rendered result for dress on female subjects

We simulate a polo shirt on three male subjects with different body shapes, and the results are shown in Figures 10 to 12.

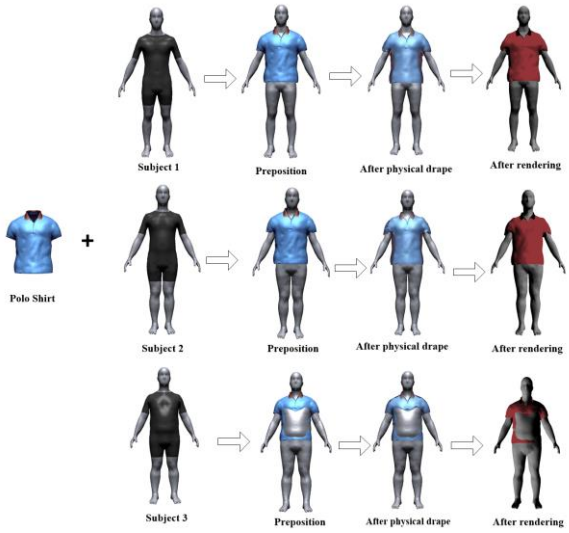


Figure 10. Polo-shirt try-on customised male models

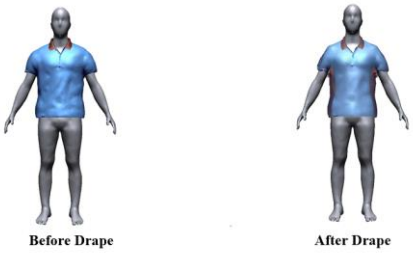


Figure 11. Subject 1's shirt wearing effects of (a) before drape and (b) after drape simulations

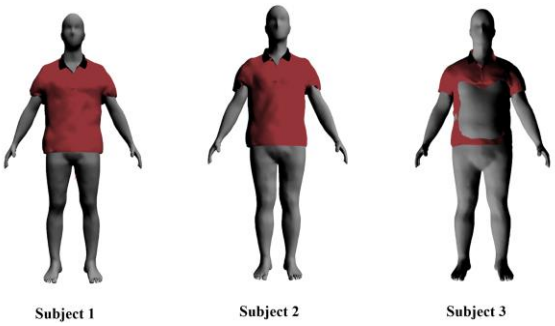


Figure 12. Rendered results for male subjects

4.2 Size and Fit Recommendations

By using virtual garment try-on technology and human model customisation, the experimental results show that subjects with different body sizes and shapes obtain different wearing effects. In addition, the system provides subjects with visual cues for size and fit evaluations.

4.2.1 Dress size recommendation:

As shown in Figure 7, the dress is loose at the waist and the bust for female subject 1. She might need to consider a smaller size. For the dress is fit for female subject 2, and the dress is obviously too small for subject 3.

4.2.2 Polo size recommendation:

As shown in both preposition and drape simulation results in Figure 10, the polo shirt fit for both male subjects 1 and 2, because the polo keeps nice appearance on these human models. The polo shirt is too tight for subject 3, he might is recommended to change to a bigger size.

We can conclude that by visualised simulation results, indirect information on fit and size are given to users.

5. CONCLUSION

In this paper, we explain the clothing industrial need on virtual fitting solutions. We propose complete pipeline obtain fit clothes using virtual fitting solution on the basis of human model customisation technology. We can conclude that customised human models are important for obtaining useful and reliable results in online fit evaluations.

In the future, clothing mechanical properties will be considered for size and fit recommendations. In practice, designers can distinguish fit from the folds and wrinkles of clothing. In the virtual environment, if clothing mechanical properties can be properly modelled the nonlinearities and deformations appearing on the virtual clothing in forms of 'winkles and folds' can be used for more detailed fit evaluations.

6. ACKNOWLEDGMENTS

The work described in this paper was partially supported by the Innovation and Technology Commission of Hong Kong under grant ITS/253/15, and a grant of Guangdong Provincial Department of Science and Technology under the grant R2015A015.

7. REFERENCES

[1] China Internet Network Information Center, http://www.cnnic.net.cn

[2] Kim, S.H. and Choi, H.S. A Study of the usage and sizing selecting of the apparels listed in on-line and catalog shopping. J Korean Soc Cloth Text. 26(7), pp.1015-25, 2002

[3] Park, J., Nam, Y., Choi, K.M., Lee, Y. and Lee, K.H. Apparel consumers' body type and their shopping characteristics. J Fash Mark Manag. 13(3), pp.372-393, 2009.

[4] Hauswiesner, S., Straka, M. and Reitmayr, G. Virtual try-on through image-based rendering. IEEE Trans. Vis. Comput. Graphics. 19(9), pp.1552-1565, 2013

[5] Divivier, A., Trieb, R., Ebert, A., Hagen, H., Gross, C., Fuhrmann, A. and Luckas, V. Virtual try-on topics in realistic, individualized dressing in virtual reality, 2004

[6] Li R, Zou K, Xu X, et al. Research of interactive 3D virtual fitting room on web envi-ronment. Computational Intelligence and Design (ISCID). 2011 Fourth International Symposium on IEEE, 1: 32-35, 2011

[7] Li, J., Ye, J., Wang, Y., Bai, L., & Lu, G. Fitting 3D garment models onto individual human models. Comput Graph. 34(6), 742-755, 2010

[8]Volino, P., Cordier, F., & Magnenat-Thalmann, N. From early virtual garment simulation to interactive fashion design. Comput Aided Des. 37(6), 593-608, 2005

[9] Cordier, F., & Magnenat-Thalmann, N. Real-time animation of dressed virtual humans. Comput Graph Forum, 21(3), 327-335, 2002

[10] Kim, S., & Park, C. K. Basic garment pattern generation using geometric modelling method. Int J Cloth Sci Tech. 19(1), 7-17, 2007

[11] Wang, C. C. L., Wang, Y., & Yuen, M. M. F. Design automation for customised apparel products. Comput Aided Des. 37(7), 675-691, 2005

[12] Carignan, M., Yang, Y., Thalmann, N. M., & Thalmann, D. Dressing animated synthetic actors with complex deformable clothes. Comput Graph. 26(2), 99-104, 1992

[13] Volino, P., Courchesne, M., & Thalmann, N. M. Versatile and efficient techniques for simulating cloth and other deformable objects. Proceedings of the 22nd Annual ACM Conference on Computer Graphics and Interactive Techniques. 137-144 , 1995

[14] Hinds, B. K., & McCartney, J. Interactive garment design. Vis Comput. 6(2), 53-61, 1990

[15] Provot, X. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In: Proceedings of the 1995 Graphics Interface Conference, Quebec, Canada. 147-154, 1995

[16] Fan, J., Yu, W., & Hunter, L. Clothing appearance and fit: Science and technology. Woodhead Publishing Ltd., Cambridge, 2004

[17] Fontana, M., Rizzi, C., & Cugini, U. 3D virtual apparel design for industrial applications. Comput Aided Des, 37(6), 609-622, 2005

[18] Kunii, T. L., & Gotoda, H. Singularity theoretical modeling and animation of garment wrinkle formation processes. Vis Comput. 6(6), 326-336 ,1990

[19] Volino, P., & Magnenat Thalmann, N. Implementing fast cloth simulation with collision response. Proceedings of The 18th Computer Graphics International 'Humans and Nature', CGI 2000, Geneva, Switzerland. 257-266, 2000

[20] Umetani, N., Kaufman, D. M., Igarashi, T., & Grinspun, E. (2011). Sensitive couture for interactive garment modeling and editing. ACM Trans. Graph., 30(4), 90.

[21] Meng, Y., Mok, P. Y., & Jin, X. (2010). Interactive virtual try-on clothing design systems. Computer-Aided Design, 42(4), 310-321.

[22] Li, Z., Jin, X., Barsky, B., & Liu, J. (2009, August). 3d clothing fitting based on the geometric feature matching. In Computer-Aided Design and Computer Graphics, 2009. CAD/Graphics' 09. 11th IEEE International Conference on (pp. 74-80). IEEE.

[23] Gill, S. A review of research and innovation in garment sizing, prototyping and fitting. Text. Prog. 47(1), pp.1-85, 2015

[24] Istook, C. L., & Hwang, S. J. (2001). 3D body scanning systems with application to the apparel industry. Journal of Fashion Marketing and Management: An International Journal, 5(2), 120-132.

[25] Wang, C.C., Wang, Y., Chang, T.K. and Yuen, M.M.: Virtual human modeling from photographs for garment industry. Comput Aided Des. 35(6), pp.577-589, 2003

[26] Zhu, S., Mok, P. Y., & Kwok, Y. L. An efficient human model customisation method based on orthogonal-view monocular photos. Comput Aided Des.45(11), 1314-1332, 2013

[27] Cordier,F., Seo,H., & Magnenat Thalmann, N. Made-to-measure technologies for an online clothing store. IEEE Comput Graph Appl. 23(1), pp. 38-48, 2003

[28] Park, J., Nam, Y., Choi, K.M., Lee, Y. and Lee, K.H. Apparel consumers' body type and their shopping characteristics. J Fash Mark Manag. 13(3), pp.372-393, 2009

[29] Lu, J.M., Wang, M.J.J., Chen, C.W. and Wu, J.H. The development of an intelligent system for customised clothing making. Expert Systems with Applications, 37(1), pp.799-803, 2010

[30] Zhu, S., Mok, P. Y. Predicting realistic and precise human body models under clothing based on orthogonal-view photos. Procedia Manufacturing, 3, 3812-3819, 2015

[31] Long J, Shelhamer E, Darrell T. (2015). Fully convolutional networks for semantic segmentation[C], Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 3431-3440.

[32] Mok, P.Y. and Zhu, SY. (2016). A method and/or system for reconstructing from images a personalized 3D human body model, and the relevant electronic devices., Patent filing number: 201610223606.0

[33] Delingette H (2008). Triangular springs for modelling non-linear membranes, IEEE Transaction on Visualization and Computer Graphics, 14(2): 329-341.

[34] Meng Yuwei, Mok P.Y. and X. Jin (2012). Computer aided clothing pattern design with 3D editing and pattern alteration, computer aided Design, 44, 721-734.

Efficient Pose Deformations for Human Models in Customized Sizes and Shapes

Shuaiyin Zhu and P.Y. Mok*
Institute of Textiles and Clothing, The Hong Kong Polytechnic University
Hunghom, Hong Kong.
sy.zhu@connet.polyu.hk; *tracy.mok@polyu.edu.hk

ABSTRACT

Modelling dynamic pose deformations of human subjects is an important topic in many research applications. Existing approaches of human pose deformations can be classified as volume-based, skeletal animation and example-based methods. These approaches have both strengths and limitations. However, for models in customized shapes, it is very challenging to deform these models into different poses rapidly and realistically. We propose a conceptual model to realize rapid and realistic pose deformation to customized human models by the integration of skeletal-driven rigid deformation and example-learnt non-rigid surface deformation. Based on this framework, a method for rapid automatic pose deformation is developed to deform human models of various body shapes into a series of dynamic poses. A series of algorithms are proposed to complete the pose deformation automatically and efficiently, including automatic segmentation of body parts and skeleton embedding, skeletal-driven rigid deformation, training of non-rigid deformation from pose dataset; shape mapping of non-rigid deformation, and integration of rigid and non-rigid deformations. Experiment has shown that the proposed method can customize accurate human models based on two orthogonal-view photos and also efficiently generate realistic pose deformations for the customized models.

Keywords

Human modelling, pose deformation, example-based approach, model customization, deformation transfer

1. INTRODUCTION

3D human body modelling is needed in many research applications, such as gaming, filming, medical, ergonomics and fashion. Modelling human body has two major aims: shape and pose modelling. A number of methods have been developed to address these two aims. For example, methods or applications [2-5,19] were reported to estimate global shape and pose model of human subjects using image or depth information as input deform a parametric model. Example-learnt parametric model can capture global shape and pose deformation realistically, and thus have used in many computer graphic applications. However, for fashion or ergonomic related research applications, the key focus of human modelling is accurate body shape of the resulting models. Parametric deformable models may not be able to accurately capture the shape details of individuals. This is because example-based methods often learn parametric deformable models form examples using Principal Component Analysis (PCA), resulting in models of 'average' looks, missing local shape characteristics of individuals. The existing human modelling methods could not address to the specific needs of fashion industry, where not only global features but also local details of the individual's body figures are precisely modelled.

We present an automatic method of customizing human body shape model from images, and deforming the customized model in different poses. We mainly follow the method of [13] for customizing accurate body shape of individuals from images, which is a complete automatic method from image processing to model deformation. However, [13] addressed to shape modelling of individuals from images only, however the customised model holds a standard standing post. In addition to shape customization, we propose a new method to deform the customized shape model into different dynamic poses. The rest of the paper is organized as follows. First, the related work of human shape and pose modelling will be reviewed in Section 2. The detailed of the proposed method will be described in Section 3, including shape customization and pose deformation. Section 4 shows some model customization results together with thorough discussion on the pros and cons of the current method. Section 5 concludes this paper and also suggestion the future work.

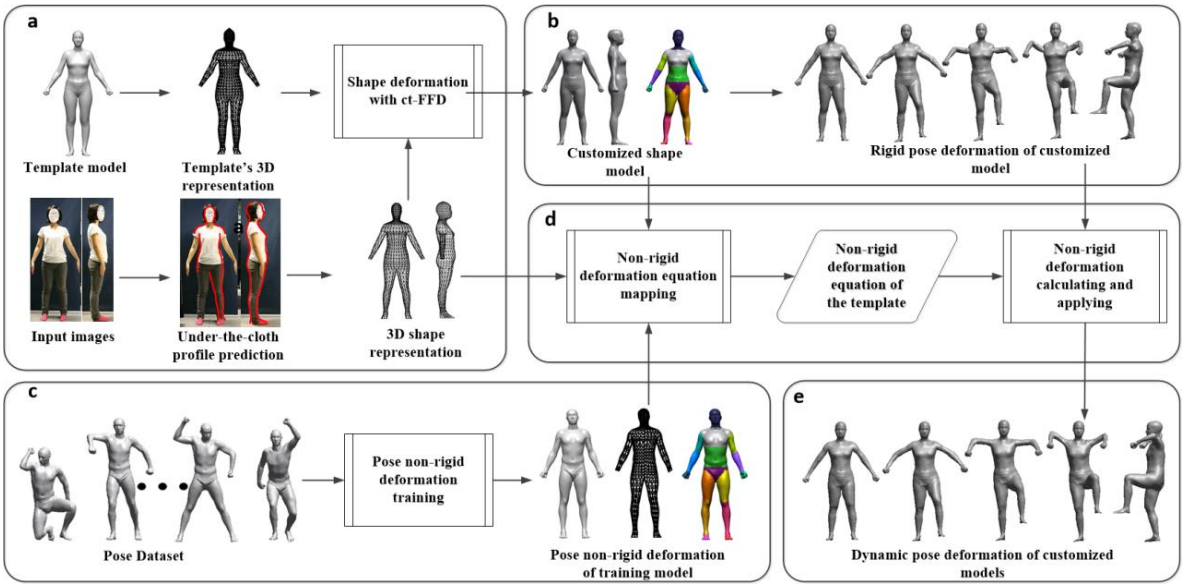


Figure 1. Dynamic pose modelling of customized human models – Method Overview

2. RELATED WORK

Existing approaches of modelling realistic pose deformations of human subjects can be classified as volume-based, skeletal animation and example-based methods. Volume-based methods [14,17] were proposed in the early stage of pose modeling research. Volume-based methods were based on physical analysis of muscle or tissue, anatomy and biomechanics theory. In volume-based pose deformation, the skeleton of a virtual 3D human body model, which includes bones, muscle, tissue and skin, was first deformed using motion data. The skeleton was used to compute the forces on the correlated muscle or tissue. The muscle and tissue deform according to computed forces, and so does the skin. Volume-based methods not only achieve realistic skin deformation but also compute the forces and kinematics of the correlated muscle or tissue. Volume-based methods are widely used in biomechanics research. However, these methods are all computationally expensive, and the pose deformation is not real-time. Furthermore, a detailed model of a human subject with skeleton, muscle and tissues is complex to create. The calculation of pose deformation for a standard model (in standard size and shape) is already tedious and time-consuming, and it is very difficult, if not impossible, to calculate the pose deformation of human models in customized shapes.

Direct-deformation methods were later proposed to model pose deformation as skin deformation, without modeling the bones, muscles and tissues under the skin. Direct-deformation methods can be classified into two categories: skeletal animation and example-driven methods. Magnenat-Thalmann [11] introduced the first skeletal animation algorithm that deforms a human hand with the assistance of virtual finger bones. The Linear Blending Skinning (LBS) algorithm [9], one of the most well-known skeletal animation algorithms, was introduced around the same time. Compared to volume-based methods, LBS provides a rapid solution for pose deformation. Thus, LBS is widely applied in game development and computer animation. However, the resulting skins of LBS algorithms have an unnatural appearance because of missing muscle deformation. Some skeletal animation algorithms were proposed to improve LBS [6-7,12,18]. For example, [18] described an enhanced skeletal animation algorithm called multi-weight enveloping (MWE), which replaces the linear model with a statistical model for the skin surface and the corresponding joints' deformation. MWE can generate realistic skin deformation if the deformed pose is within the training dataset. Other well-known extensions of LBS includes Spherical Blend Skinning [7] and Dual Quaternion blending Skinning [8]. In sum, skeletal animation algorithms provide solutions for rapid pose deformation, but the shape accuracy of the deformed model is questionable, especially in the case of large joint angle changes.

Example-driven pose deformations [2,10,12,15,18] were introduced in the last decade or so, by taking advantage of scanning technology development and the availability of scanning data. For example, [1] described the first example-driven method that trains scanned human models in different poses. More recently, SCAPE [2] was introduced to deform the shape and pose of a template model. However, example-based methods that model shape deformation in a global space are not good at

customizing local shapes. Besides, example-based methods generate erroneous deformation in the case of a subject with a shape out of the scope of the training dataset. It is thus not ideal to deform customized models into different poses.

Recently, some researchers proposed to use motion data for shape modelling or mapping [22,23], while we propose to use images in this paper. Pons-Moll [25] developed a novel approach to model the soft-tissue deformations of full-body human models using 4D capture system. It used example-based approach for volume-based deformation, adding more detailed precise pose modelling for different subjects.

3. METHOD

3.1 Method Overview

Figure 1 shows an overview of human modelling method, it integrate shape modelling and pose modelling of human subjects. It involves a shape modelling method of individuals based on two orthogonal-view photos (as shown in (a) of the figure); and a pose modelling method, which integrates rigid deformation and non-rigid deformation (shown as (b-e) of the figure).

3.2 Shape Customization

We describe here an automatic method that constructs a 3D shape model of an individual based on his/her front-view and side view photos. The detailed method has been reported in [13]. Our method of shape customization defined a feature-aligned 3D shape representation to characterize human body physique in detail (refer to (a) of Figure 1). The 3D shape representation is a layered structure, each layer represents a cross-sectional shape of the subject's body at a specific level.

A layered structure with parallel cross-sections is an effective shape representation for clothing applications, because such a structure aligns with the clothing size definition. The parallel cross-sections characterize *local features* of the body in terms of *shape and size*, as some cross-sections indeed correspond to important body girth measurements, such as the bust, waist, and hip. The shape of these cross-sections gives detailed information on where the body has developed fat, for instance the shape of the waist girth.

Different from the example-based methods, which acquired global shape variations from full body scans, our shape customization method is capable of capturing and modelling the local body shape characteristics of individuals, by learning the cross-sectional size and shape relationships from over 10000+ scanned models using neural networks. Our method extract 2D body features from subjects' orthogonal-view photograph, from which to predict the cross-sectional shape in 3D, and then to reconstruct the overall 3D shape representation. Without assuming linear shape deformation, the method is able to capture local shape characteristics and has been applied to customize 3D body models for subjects dressed in tight-fit clothing [20] or dressed in arbitrary clothing, even loose-fit clothing [21]. The recent advancement of the shape modelling method is that new algorithms were developed to extract all features needed for the shape customization process from input images. These 2D features can identify and align with the 3D layered structure of shape representation accurately using anthropometric knowledge [13]. It implies that the accurate 3D models can be customized from input images in a complete automatic manner. The experimental results showed that output human models (both males and females) have accurate sizes (anthropometric measurements) and realistic shape details. The size measurement discrepancies between the resulting models and the scanned models are less than 2cm in all key girth measurements like bust, waist and hip.

3.3 Pose Deformation

As shown in Figure 1, we propose to model dynamic pose deformation to any customized model as an integration of pose-induced rigid deformation and non-rigid deformation. Rigid deformation here represents the deformation induced by different body parts' rotation and translation, such as forearms or lower legs. The deformation of these body parts explains most of the global change in body shapes in different dynamic pose deformations. In rigid deformation, a human model is deformed like a puppet, and such deformations are modeled as the rotation of different body parts along an articulated skeleton. Realistic skin surface deformation due to different pose change is obvious very different from rotation of an articulated model, and we model such skin surface differences as non-rigid deformations. We use the pose dataset of [2], which has a wide range of scan of a single subject in different poses. All scan models are properly registered, meaning that they have the same topology structure. The pose dataset in fact provides examples of realistic skin surface deformations, we can obtain non-rigid deformation to each triangle by computing the differences between rigidly deformed mesh and the corresponding pose model in the pose dataset. We then learn a regression model to predict non-rigid deformations using motion data. Our idea of pose modeling is similar to that of SCAPE [2], but we improve the detailed implementation in different areas, which will be explained in details below.

Pose deformation is modelled with reference to each triangle of a mesh model (template). Such formulation enables easy representation of deformation by matrix operations:

$$\tilde{\mathbf{V}}_k = \mathbf{T}\mathbf{V}_k^0 = \mathbf{T}_{L(k)}^R \mathbf{T}_k^Q \mathbf{V}_k^0 . \quad (1)$$

As shown in Equation (1), each triangle $\mathbf{V}_k^0$ of the template is multiplied by transformation matrix $\mathbf{T}$ to obtain its pose deformation, where $\mathbf{T}$ is a matrix product of two components $\mathbf{T}_{L(k)}^R$ and $\mathbf{T}_k^Q$. $\mathbf{T}_k^Q$ represents non-rigid pose-induced deformations specific to triangle $\mathbf{V}_k^0$ of the template; $\mathbf{T}_{L(k)}^R$ denoted the rotation of the rigid body part in the articulated skeleton, where $L(k)$ denotes the rigid part that triangle $\mathbf{V}_k^0$ is belonged to. Both $\mathbf{T}_k^Q$ and $\mathbf{T}_{L(k)}^R$ are functions of motion data, represented as joint rotation angles of a defined skeleton structure.

3.3.1 Rigid pose deformation

To obtain rigid deformation of a human model, the first step is to group triangle faces of the human mesh model into different rigid parts. Automatically grouping triangle surfaces is a typical mesh surface clustering problem. There are many different algorithms reported in the literature to recover articulated object models from meshes. In the vast majority of the applications, the articulated skeleton structure and its parameters have to be manually specified. In the SCAPE method, object surface segmentation is done by iteratively finding decomposition of the object surface into rigid parts and finding the location of the parts based on a set of registered 3D scans in different configurations [2]. They estimated the locations of the joints from the resulting segmentation. Their algorithm not only recovered the parts and the joints, but also figured out the optimal number of parts numerically. However, the availability of a sample of object instances is a necessary input in the SCAPE method for mesh segmentation. Moreover, recovering articulated skeleton as a numerical optimization problem from the sample scans may not result in parts that aligned well with human body features, and its segmentation results may vary when different dataset is used. Numerically optimized part number may result in skeleton structure different from the typical skeleton definition for animation applications.

We develop an efficient and automatic method that segments triangle faces of the mesh model into meaningful rigid parts, using the 3D shape representation constructed in the phase of shape modeling. The 3D shape representation has included a number of body features defined based on anatomical knowledge. We thus assign the triangle faces of 3D shape representation with a rigid part label $a_j \in \{a_1,\ldots,a_N\}$. We assign part labels based on the skeleton definition of the SCAPE model with a total of 16 rigid parts $\{a_1,\ldots,a_{16}\}$. However, it is important to note our method is flexible in term of the definition of skeleton structure, because we define using anatomical knowledge with identified body features of the 3D shape representation, which can be viewed as a simplified version of human model. In other words, we can model pose deformation using other skeleton structure definition that allow easy integration with motion data.

Given a customized human model in Figure 2(a), we can construct a 3D shape representation for the mesh model with faces labelled, as shown in Figure 2(b). Based on the labelled 3D shape representation, we calculate the correspondence relationship and assign part label for every triangle face of the customized human model with the following equations.

$$L(k) = L(a_j) \quad (2a)$$

such that

$$\arg \min_{\{a_1,\ldots,a_N\}} \left\| c_k - a_j \right\|^2 + \varphi(N(c_k), N(a_j)) \quad (2b)$$

In equation (2a), $L(k)$ denotes the centroid position c_k of triangle *k* on the human model, and a_j denotes the centroid position of the *j*-th triangle of the 3D shape representation. $L(a_j)$ records the part label $a_j \in \{a_1,\ldots,a_N\}$ that Equation (2b) is minimized. In Equation (2b), $\varphi(\cdot)$ is penalty function defined as

$$\varphi(N(c_k), N(a_j)) = \begin{cases} 0, & \text{if angle between } N(c_k) \text{ and } N(a_j) < 90^\circ; \\ \infty, & \text{else.} \end{cases} \quad (2c)$$

where function $N(c_k)$ calculates the normal vector of the triangle *k* at its centroid c_k. The penalty function $\varphi(\cdot)$ filters those triangles with different normal vectors, especially at connecting areas of two or more rigid parts, e.g. armpit. We only segment the 3D shape representation once, because the topology structure of 3D shape representation is the same for all customized human models. Figure 2(c) shows the result of segmenting input human model (a) into rigid parts.

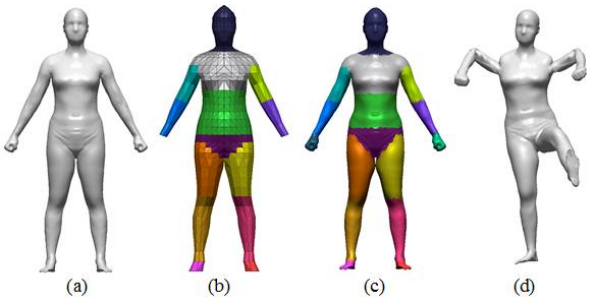


Figure 1. Part grouping and rigid part deformation.

3.3.2 Training of Non-rigid pose deformation

Since pose-induced non-rigid deformation for a customized model is unknown, we use the concept of deformation transferring [16] to copy the deformation from known pose dataset to the customized model. In addition to the given poses in the pose dataset, we train a model, using the pose dataset, to predict the non-rigid deformation of triangle *k*, namely the transformation matrix $\mathbf{T}_k^Q$ in Equation (1), using two adjacent joints' rotations, $\Delta r_{L(k),1}$ and $\Delta r_{L(k),2}$. Again, the formulation of non-rigid deformation is toward each triangle, $\tilde{\mathbf{V}}_k = \mathbf{T}_k^Q \mathbf{V}_k^0$.

The detailed formulation of non-rigid deformation is described as follows. We predict non-rigid deformation from articulated human pose, which is represented by a set of relative joint rotations. With a specific pose (*i*), the non-rigid deformation for every triangle *k* of the mesh is denoted by a 3×3 matrix

$$\mathbf{T}_k^{Q(i)} = \begin{bmatrix} q_{k,11}^i & q_{k,12}^i & q_{k,13}^i \\ q_{k,21}^i & q_{k,22}^i & q_{k,23}^i \\ q_{k,31}^i & q_{k,32}^i & q_{k,33}^i \end{bmatrix} \quad (3)$$

We reduce the dimensionality of the problem by assuming $\mathbf{T}_k^{Q(i)}$ is affected by two adjacent joints of the rigid part where triangle k is belonged to. Each entry $q_{k,mn}^i$ of matrix $\mathbf{T}_k^{Q(i)}$ is thus inner product of two vectors:

$$q_{k,mn}^i = [\Delta r_{L(k),1}^i \; \Delta r_{L(k),2}^i \; 1] \cdot \mathbf{b}_{k,mn}^i \quad m,n=1,2,3 \quad (4)$$

where $\mathbf{b}_{k,mn}^i$ is a 7×1 column vector of linear weights that map the rotations of two adjacent joint angles ($\Delta r_{L(k),1}^i$, $\Delta r_{L(k),2}^i$) and a constant bias term as the entry value of transformation matrix $\mathbf{T}_k^{Q(i)}$. Each joint rotation $\Delta r_{L(k),1}^i$ can be easily computed from the absolute rotation matrices of the two rigid parts, and be specified by three parameters in twist coordinates.

The goal is to learn the parameters $\mathbf{b}_{k,mn}^i$ in Equation (4). We train this non-rigid deformation prediction model using SCAPE pose dataset. We construct the 3D shape representation for the SCAPE template at standard pose (pose zero) of the pose dataset. We next segment this SCAPE template into rigid parts and extract skeleton structure accordingly. We then obtain the rigid deformations of the SCAPE template by rigid part rotation $\mathbf{T}^{R(i)}$ for all pose instances (*i*) given in the pose dataset, using the method described in Section 7.2.3 above.

Each mesh model in the pose dataset is the final shape integrating both rigid $\mathbf{T}^{R(i)}$ and non-rigid $\mathbf{T}^{Q(i)}$ pose deformations. Given transformation for each mesh instance $\mathbf{T}_k^{Q(i)}$ and its rigid part rotation $\mathbf{T}_{L(k)}^{R(i)}$, solving for the 9×7 regression parameters $\mathbf{b}_{k,mn}^i$ for each triangle *k* is straightforward by minimizing a quadratic cost function

$$\arg\min_{\mathbf{b}_{k,mn}^i} \sum_{i=1}^{m} ([\Delta r_{L(k),1}^i \; \Delta r_{L(k),2}^i \; 1] \cdot \mathbf{b}_{k,mn}^i - q_{k,mn}^i)^2 \quad (5)$$

To solve Equation (5), we performed PCA on the observed joint rotation angles ($\Delta r_{L(k),1}^i$, $\Delta r_{L(k),2}^i$), reducing the size of the problem because many human joints only have one or two degree of freedom instead of three. With learned parameters $\mathbf{b}_{k,mn}^i$, we can calculate using Equation (4) the non-rigid deformation matrix $\mathbf{T}_k^{Q(i)}$ based on pose/motion data, in terms of joint rotation angles. Figure 2 shows the examples of pose deformation of SCAPE template. These poses are not given in pose dataset, but new poses synthesized completely from a vector of joint rotations.

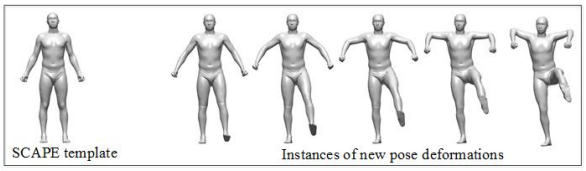


Figure 2. Pose deformation of the SCAPE template with synthesized joint rotations.

3.3.3 Pose deformation of customized human models

We model realistic skin surface deformation induced by different dynamic poses (defined in terms of joint rotation angles) using a pose dataset on a template mesh. The next question to answer is how realistic pose deformation can be obtained for a customized human model. We use the concept of deformation transfer that transfers the deformations of the SCAPE model (source) to the customized model (target).

the application of deformation transfer technique requires the availability of pairwise correspondence between source and target meshes. In general, with a set of users selected marker points, pairwise correspondence are determined by deforming target mesh into a shape of the source and then by calculating the closest Euclidean distance between the source mesh and the deformed target mesh. Such deformation is iteratively improved. Well known correspondence searching algorithms include Iterated Closet Points (ICP) [1][16] and Correlated Correspondence [2]. The selection of marker points requires some manual work at the initial stage and the involvement of manual work also causes the final results depend on users' selection or experience.

We propose a completely automatic correspondence alignment method for human models using the shape modeling method developed early. Our correspondence mapping involves three simple steps: 1) construct 3D shape representations for the source mesh (SCAPE template) and target mesh (customized human model); 2) deform the source mesh using ct-FFD [20] based on the 3D shape representation of the target; 3) for each triangle *k* of the target mesh, seek the triangle on the deformed source mesh so that centroids of the two triangles are in the closest proximity. This can be done by Equation (2b), except the different definitions of a_j, which denotes the centroid position of the *j*-th triangle face of the source mesh (SCAPE template) instead of the 3D shape representation. There are no restrictions on the correspondence mapping, and it can be injective, surjective, or bijective.

Our method in fact automatically identifies all the marker points without any manual operations. It ensures fine alignment with human body features, and the process is not iterative. By eliminating iterative deformations, our method is very efficient to register two human models. Table 1 compares the time spent by ICP method and our method for registering SCAPE template with a target mesh of 23112 triangles. It is shown that our method is much more efficient than ICP in correspondence mapping.

Table 1 Time comparison of registration methods

Registration methods	ICP (1 iteration)	ICP (6 iterations)	ct-FFD-based
Time spent (Mesh with12428 vertices and 23112 triangles)	24s (Core i7/8GB Memory/ Matlab)	229s (Core i7/8GB Memory/ Matlab)	5.1s (Core i7/8GB Memory/ Matlab)

Upon establishing correspondence between SCAPE template and a customized model, we can transfer pose-induced non-rigid deformation from the source mesh (SCAPE) to the target mesh (customized model) to finish dynamic pose deformations on customized model. This is done by setting the gradient of objective function

$$\arg\min_{\tilde{\mathbf{x}}} \left\| \mathbf{c} - \mathbf{A}\tilde{\mathbf{x}} \right\|_2^2 \qquad (5)$$

to zero, where $\tilde{\mathbf{x}}$ is a vector of unknown deformed vertex locations of customized model, **c** is a vector recording transformations of the mesh of the pose dataset, is is calculated as

$$\mathbf{c} = \mathbf{T}^R\mathbf{T}^Q = \mathbf{A}\tilde{\mathbf{x}} \qquad (6)$$

A of Equation (5) is a large, sparse matrix that the entries in **A** depends only on the target mesh's structure. In Equation (6), the rigid part transformation matrix $\mathbf{T}^R$ is known based on given motion data and once the part grouping result of the customized model is known. With the motion data, we also calculate the non-rigid deformation using the SCAPE model. For every customized model, the sparse matrix **A** in Equation (6) is known. We can then calculate dynamic pose deformation of the customized model accordingly by integrating non-rigid deformation and rigid part rotation by equation (1). Figure 3 shows the corresponding pose deformation transferred from SCAPE template in Figure 2. The resulting pose deformations on a customized model have natural and realistic skin surface deformation appearance similar to that of the source (Figure 2).

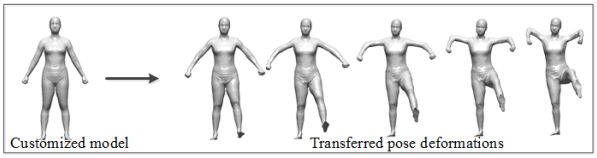


Figure 3. Resulting pose deformation on a customized model

4. RESULTS AND DISCUSSION

We evaluate the pose deformation by recruiting eight female and eight male subjects. These subjects can be classified into underweight, normal, overweight and obesity according to their BMI values. Figure 4 shows four examples of the shape customization results, which are compared to their scanned models. Figures 6 and 7 show, respectively, all female and male subjects' customized models in standard pose.

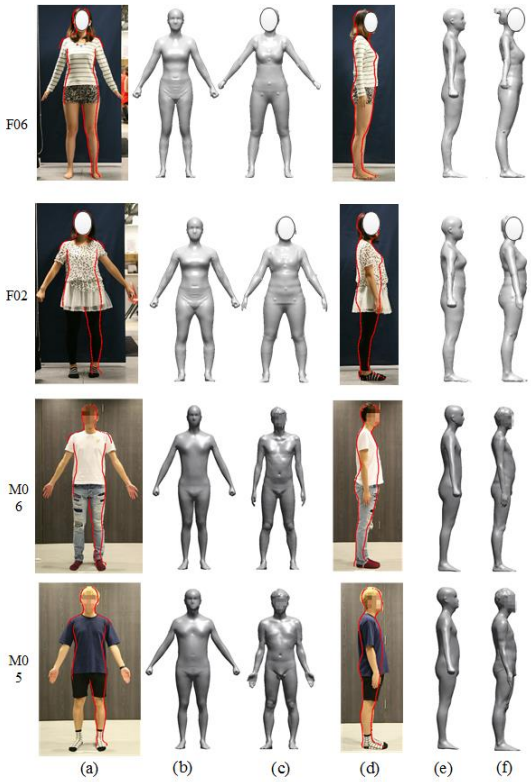


Figure 4. Four shape customization results

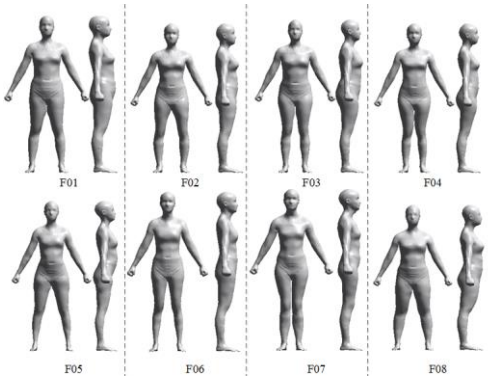


Figure 5. Female customized models in standard standing pose

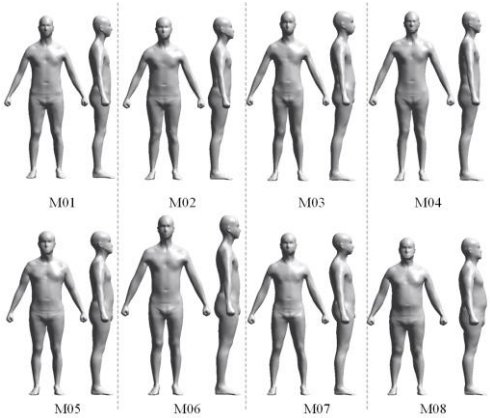


Figure 6. Male customized models in standard standing pose

We deformed all subjects' customized models into other poses dynamically. Figure 7 and Figure 8 show dynamic pose deformation examples of these subjects, and these poses are given in the pose dataset of SCAPE model. Comparing with the rigid part deformation based on the articulated models, our method generates realistic pose deformations on customized models, similar to that of the SCAPE model, even in poses with maximum bending angles.

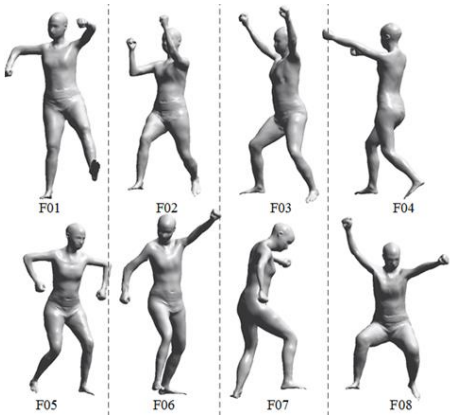


Figure 7. Dynamic deformed female models

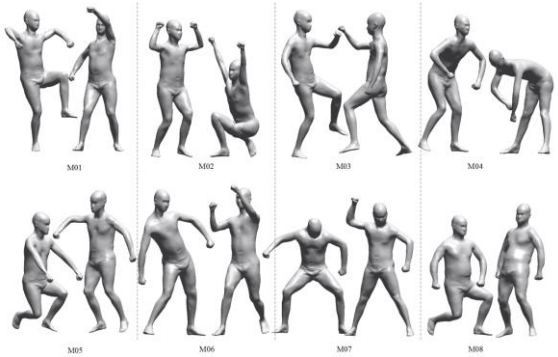


Figure 8. Dynamic deformed male models

In order to evaluate the pose deformation comprehensively, we scanned two female subjects in three different poses, as indicated on the right of Figure 9. We estimated their pose or skeleton data by manually select marker points from scanned models. Based on these pose data, we deformed the corresponding customized models as shown on the left of Figure 9.

As poses of scanned model are estimated, the estimated skeleton is not exactly equal to the real pose in the scans. Having said that, we can see the shape of the scanned models and the deformed models resembles, except some local areas, e.g. the chest areas. That is because the pose induced non-rigid deformation is transferred from the SCAPE model (a male scanned subject) to other customized models. Non-rigid deformation can be viewed as muscle deformation, yet the same muscle model is used for all people. Although the model can generate

realistic mesh models in a wide range of poses, the current method cannot capture the fact that more muscular people are likely to exhibit greater muscle deformation than others, and, conversely, that muscle deformation may be obscured in people with significant body fat.

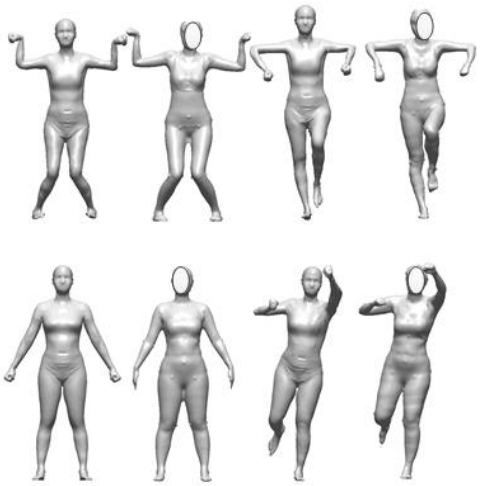


Figure 9. Comparison between pose deformation models and the corresponding scanned models

We can improve the current method using scans of different example subjects in different poses to learn the non-rigid deformation, and we transfer the deformation from one of the example subject that has the closest shape characteristic as that of the customized model.

5. CONCLUSION

We have presented human modelling method that automatically obtain 3D shape model using two orthogonal-view photographs. We also present an efficient method to deform any customized model into dynamic poses. Pose deformation is done by first grouping the triangles mesh of the customized model into articulated rigid parts with the assistance of 3D shape representation constructed in the step of shape modeling. Next, the human model is transformed in the form of rigid part rotations. Then, we learn the skin surface (non-rigid) deformations from a pose dataset to correct the defects in rigid deformation. The non-rigid deformation is transferred from pose dataset to the customized model so as to have realistic pose induced shape deformations on customized model. We optimize the procedures to realize very efficient pose deformations on customized models with diverse shapes.

6. ACKNOWLEDGMENTS

The work described in this paper was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU 5218/13E). The partial support of this work by the Innovation and Technology Commission of Hong Kong, under grant ITS/253/15, and The Hong Kong Polytechnic University, under project code: RPUC, are gratefully acknowledged.

7. REFERENCES

[1] Allen, B., Curless, B. & Popović, Z. (2002) Articulated body deformation from range scan data, ACM Transactions on Graphics, 21:1–8.

[2] Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J. & Davis, J. (2005) SCAPE, ACM Transactions on Graphics, 24(3), p. 408.

[3] Balan A., Sigal L., Black L., Davis J., and Haussecker. H. (2007). Detailed human shape and pose from images. IEEE Conference on Computer Vision and Pattern Recognition 2007 (CVPR '07), pp.1-8, 17-22 June 2007.

[4] Bogo, F., Black, M. J., Loper, M. and Romero, J. (2015). Detailed full-body reconstructions of moving people from monocular RGB-D sequences, Proceedings of the IEEE International Conference on Computer Vision, pp. 2300–2308.

[5] Guan, P.G.P., Weiss, A., Balan, A.O. and Black, M.J. (2009) Estimating human shape and pose from a single image, Computer Vision, IEEE 12th International Conference on Computer Vision.

[6] Kavan, L., & Zara, J. (2003). Real Time Skin Deformation with Bones Blending. In WSCG.

[7] Kavan, L. and Žára, J. (2005). Spherical blend skinning: a real-time deformation of articulated models, Proceedings of the 2005 symposium on Interactive 3D graphics and games, 1(212), pp. 9–16.

[8] Kavan, L., Collins, S., Žára, J. and O'Sullivan, C. (2008). Geometric skinning with approximate dual quaternion blending, ACM Transactions on Graphics, 27(4), pp. 1–23.

[9] Lander, J. (1999), Over my dead, polygonal body, Game Developer Magazine, (October), pp. 17–22.

[10] Lewis, J.P., Cordner, M. and Fong, N. (2000), Pose space deformation, Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00, pp. 165–172.

[11] Magnenat-Thalmann, N. R. Laperriere, and D. Thalmann, Joint-dependent local deformations for hand animation and object grasping, Proc. Graphics Interface'88, pp. 26–33, 1988.

[12] Mohr, A. and Gleicher, M. (2003) 'Building efficient, accurate character skins from

examples', ACM Transactions on Graphics, 22(3), p. 562.

[13] Mok P.Y. and Zhu SY. (2017), Precise shape estimation of dressed subjects from two-view image sets, In Calvin Wong (Ed.) Applications of Computer Vision in Fashion and Textiles, Elsevier.

[14] Nakamura, Y., Yamane, K., Suzuki, I. and Fujita, Y. (2003). Dynamic Computation of Musculo-Skeletal Human Model Based on Efficient Algorithm for Closed Kinematic Chains, Proceedings of the 2nd International Symposium on Adaptive Motion of Animals and Machines, p. SaP-I-2.

[15] Sloan, P.P.J., Rose, C.F. and Cohen, M.F. (2001). Shape by example, i3D - Interactive 3D Graphics and Games, pp. 135–143.

[16] Sumner, R. W. and Popović, J. (2004), Deformation transfer for triangle meshes, ACM Transactions on Graphics, 23(3), p. 399.

[17] Suzuki & Takatsu. (1997). 3D and 4D Visualization of Morphological and Functional Information from the Human Body Using Noninvasive Measurement Data, Atlas of Visualization, Vol 3..

[18] Wang, X. C. and Phillips, C. (2002) Multi-weight enveloping, Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '02, p. 129.

[19] Weiss, A., Hirshberg, D. and Black, M.J. (2011), Home 3D body scans from noisy image and range data, Proceedings of the IEEE International Conference on Computer Vision, pp. 1951–1958.

[20] Zhu, S., Mok, P. Y. and Kwok, Y. L. (2013) An efficient human model customization method based on orthogonal-view monocular photos, CAD Computer Aided Design, 45(11), pp. 1314–1332.

[21] Zhu, S. and Mok, P. Y. (2015) 'Predicting Realistic and Precise Human Body Models Under Clothing Based on Orthogonal-view Photos', Procedia Manufacturing. Elsevier B.V., 3(Ahfe), pp. 3812–3819.

[22] Güdükbay, U., Demir, İ., & Dedeoğlu, Y. (2013). Motion capture and human pose reconstruction from a single-view video sequence. Digital Signal Processing, 23(5), 1441-1450.

[23] Gültepe, U., & Güdükbay, U. (2014). Real-time virtual fitting with body measurement and motion smoothing. Computers & Graphics, 43, 31-43.

[24] Pons-Moll, G., Romero, J., Mahmood, N., & Black, M. J. (2015). Dyna: A model of dynamic human shape in motion. ACM Transactions on Graphics (TOG), 34(4), 120.

Human Action Recognition in Videos: A comparative evaluation of the classical and velocity adaptation space-time interest points techniques

Ana Paula G S de Almeida
University of Brasilia, Brazil
anapaula.gsa@gmail.com

Bruno Luiggi M. Espinoza
University of Brasilia, Brazil
bruno@cic.unb.br

Flavio de Barros Vidal
University of Brasilia, Brazil
fbvidal@unb.br

Abstract

Human action recognition is a topic widely studied over time, using numerous techniques and methods to solve a fundamental problem in automatic video analysis. Basically, a traditional human action recognition system collects video frames of human activities, extracts the desired features of each human skeleton and classify them to distinguish human gesture. However, almost all of these approaches roll out the space-time information of the recognition process. In this paper we present a novel use of an existing state-of-the-art space-time technique, the Space-Time Interest Point (STIP) detector and its velocity adaptation, to human action recognition process. Using STIPs as descriptors and a Support Vector Machine classifier, we evaluate four different public video datasets to validate our methodology and demonstrate its accuracy in real scenarios.

Keywords

human action recognition, support vector machine, space-time interest points, C-STIP, V-STIP

1 INTRODUCTION

As computer vision trends come and go, there are topics that remain widely studied, such as human action recognition. This field is being studied over the last three decades, using numerous techniques and methods to solve a common problem.

In [GBS07], human action recognition is described as a key component in many computer vision applications: video surveillance, human-computer interface, video indexing and browsing, recognition of gestures, analysis of sports events and dance choreography.

Many approaches were developed in the last years, but most of them have computational limitations, some of wich are: difficulty to estimate the motion pattern [Bla99], aperture problems, discontinuities and smooth surfaces [lLL06]; all related with the motion estimation technique used, like optical flow or more complex techniques, such as eigenshapes of foreground silhouettes, described in [GKRR05].

Some approaches, in recent successful works, use human action recognition information from video sequences as a space-time volume of intensities, gradients, optical flow, or other local features [ZMI01a, SI05].

Our methodology is similar to the proposed approach of [SLC04] in which the classical space-time interest point detector is discussed and classified using a Support Vector Machine (SVM). However, only local spatio-temporal and histograms of local features were used. This limitation shows that for small videos dataset it properly works, but for large and complex video datasets, a global spatio-temporal descriptor is required.

In this paper, we use an existing state-of-the-art detector of human action recognition as a descriptor, without the assistance of other descriptors. Section 2 describes the main related works about human action recognition. Section 3 presents a detailed explanation of the space-time interest points techniques used in this work. In Section 4, the proposed methodology is shown and in Section 5, experimental results are discussed. Section 6 is dedicated to conclusions and further work.

2 RELATED WORKS

Commonly, human action recognition is divided into two major classes, model-based recognition and appearance-based recognition [BD01, Lap04].

Many works use a human model [Roh94, GD96], generally obtained by recreating the human body using a three-dimensional model with degrees-of-freedom that allows distinct poses, representing a certain movement that corresponds to an action. With more degrees-of-freedom, a larger number of body positions can be

achieved, creating a greater number of different movements and, consequently, represented actions. A general model-based approach, as described in [Kan80], calls for a robust background and foreground segmentation to be able to distinguish between the picture domain and the scene domain. However, they are an easy way to estimate and predict the feature locations [BD01].

According to [BD01] appearance-based approaches are focused on representing an action as a motion over time and the motion recognition is achieved from appearance, since it has a space-time trajectory. In [DB97], an image template is used to recognize an action in video and it is obtained by accumulating motion images from specific key frames, in order that an image-vector is constructed and matched against previously generated ground-truth templates. This approach is used to construct a view-specific representation of action.

The approach described in [LL03] uses combined techniques based on appearance (e.g. [ZMI01b] and [MS04]) to achieve a novel motion event detector using local information and a 3D extension of Harris corner detector, named space-time interest points (STIP). One advantage of this representation is that it does not need previous segmentation or tracking [LAS08].

Over the years, numerous STIPs detectors have been proposed: the methodology shown in [DRCB05] uses a space-time cuboid to represent an interest point, achieved from the convolution of a quadrature pair of 1D Gabor filters with a 2D Gaussian smoothing kernel; an evolution of the first presented STIP is proposed by [LL04], adapting the velocity and spatio-temporal scale features, in order to obtain a stable video representation.

In [WTVG08] a Hessian-based approach is proposed, using the determinant of the Hessian as a saliency measure, being able to extract scale invariant features and densely cover the video content. And in in [CHMG12], a selective method which applies surround suppression combined with local and temporal constraints is shown, including a Bag of Videos model to build a vocabulary of visual-words for action recognition process.

3 SPACE-TIME INTEREST POINTS

According to [Lap04], the main purpose of the STIPs is to perform the event detection directly from the spatiotemporal data of the image, considering regions that have distinct locations in space-time with sufficient robustness to detect and classify. [LL03] use a 3D extension of Harris corner detector to detect these interest points.

[HS88] accentuate image areas that have maximum variation of image gradients in a local neighborhood. The goal of the Harris interest point detector is to find spatial locations where the image has significant changes in both directions.

Classical STIP

We consider the classical STIP (C-STIP) as the one proposed by [LL03] and its mathematical representation will be reviewed below.

Considering an image sequence $I(x,y,t)$ and its scale-space representation as

$$S(\cdot,\alpha^2,\beta^2) = I * G(\cdot,\alpha^2,\beta^2) \tag{1}$$

where α^2 is the spatial variation, β^2 is the temporal variance and G is a Gaussian convolution kernel

$$G(x,y,t,\alpha^2,\beta^2) = \frac{exp(-(x^2+y^2)/2\alpha^2 - t^2/2\beta^2)}{\sqrt{(2\pi)^3\alpha^4\beta^2}} \tag{2}$$

To detect interest points in I, it is necessary to search for meaningful eigenvalues $\lambda_1, \lambda_2, \lambda_3$ of a second-moment matrix γ

$$\gamma(\cdot,\alpha^2,\beta^2) = G(\cdot,s\alpha^2,s\beta^2) * (\nabla I(\nabla I)^T) \tag{3}$$

that uses spatio-temporal image gradients $\nabla I = (I_x, I_y, I_t)^T$ within a Gaussian neighborhood of every point.

And compute the local maxima of the extended Harris corner function R

$$\begin{aligned} R &= det(\gamma) - k \text{ trace}^3(\gamma) \\ &= \lambda_1\lambda_2\lambda_3 - k(\lambda_1\lambda_2\lambda_3)^3 \end{aligned} \tag{4}$$

where k is the sensitivity factor.

Thus, STIPs of I are discovered by detecting local positive spatio-temporal maxima in R.

Velocity adapted STIP

[LL04] adapted the local velocity and scales of C-STIP to compensate the relative motion between the object and the camera. To adapt the scales, a normalized Laplacian operator is calculated for each detected interest point in R (Equation 4)

$$\nabla^2 I = I_{x,norm} + I_{y,norm} + I_{t,norm} \tag{5}$$

where $I_{x,norm}, I_{y,norm}$ and $I_{t,norm}$ are the second-order derivatives of I normalized by the scale parameters. The space-time maxima of the Harris corner function and a selection of points that maximizes the Laplacian normalizes operator is a method to adapt the previous C-STIP implementation [Lap05]. Lastly, to perform the

velocity adaptation, we have to consider the Galilean transformations that affects the time domain. To cancel these effects we have to redefine the an operator of interest γ'' in terms of a velocity-adjusted descriptor, using a structure similar to that presented by [LK81]. The Equation 3 can now be redefined as

$$R_v = det(\gamma'') - k \text{ trace}^3(\gamma''). \quad (6)$$

4 MATERIALS AND METHODS

As stated earlier, our approach is based on [LL03] technique, here nominated as C-STIP, and on [LL04], named V-STIP.

The state-of-the-art STIP is used as a local detector. To use it in a recognition and classification process, an additional descriptor, such as Histogram of Oriented Gradients (HOG) [DT05] or optical flow [BFB94], is used to improve the local features. Our methodology uses STIP as a descriptor without the aid of any other descriptors.

Following the flowchart described in Figure 1, each step of our proposed method will be more detailed.

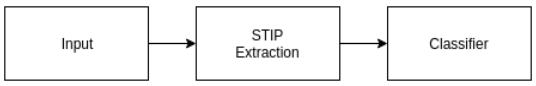


Figure 1: Our proposed methodology flowchart.

Input

For a complete evaluation of our proposed methodology, A number of specialized public video data sets were used to recognize human action. The main datasets used were:

- KTH [SLC04];
- UCF101 [SZS12];
- Weizmann [BGS05];
- YouTube [LLS09].

For each dataset, four different actions were studied in our proposed approach. An image sample of the selected actions and datasets is described in Table 4.

In the UCF101 dataset the following classes were used: *biking*, *jumping jack*, *punch* and *walking with dog*, with ten videos per action lasting about five seconds. In KTH the used classes were: *boxing*, *handwaving*, *running* and *walking*, having ten videos per action with an estimated duration of twenty seconds. In Weizmann the used classes were: *bend*, *gallop sideways*, *jump in place* and *skip*, with nine videos per action and lasting about two seconds. Finally, YouTube used the classes: *basketball*, *diving*, *soccer juggling* and *volleyball spiking*, consisting of seven videos per action with an approximate duration of seven seconds.

All selected video classes in the used datasets were randomly separated into training set (70%) and validation set (30%) and the actions were chosen in a way that it is possible to make a fair comparison between the both techniques C-STIP and V-STIP.

STIP Extraction

Since the state-of-the-art approach is C-STIP and V-STIP is its upgrade, our approach uses the methods showed in Section 3 to extract STIPs.

For the C-STIP implementation, the Equations 1 to 4 were used, generating a feature vector that is mathematically described in Section 3. Additionally, the reference structure can be found in [LL03].

The V-STIP implementation uses Equations 1 to 5, also producing a feature vector. For more details of the technique, we refer to [LL04].

Support Vector Machine

Support Vector Machines (SVMs) are state-of-the-art classifiers that produces an hyperplane based on the training data [CV95]. This achieved hyperplane has a prediction of the class labels of the validation data without further information besides its features. In [CV95] a more detailed structure of SVM is described.

Considering a training set of $(x_i, y_i), i = 1, ..., l$, where $x_i \in \Re$ is a feature vector and $y_i \in \{1,2,3,4\}$ its class labels, a Support Vector Classification (SVC) algorithm [BHHSV01] with a radial basis kernel, as shown in Equation 7, is used to train and predict the validation set class labels.

$$K(x,y) = exp(-\gamma \|x-y\|^2) \quad (7)$$

where here γ is the kernel parameter.

From the trained and validated C-STIP and V-STIP models, we evaluated all classes using a Mean Average Precision (MAP) measure, similar to [CHMG12] and [DOS15].

5 RESULTS

To train the SVM classifier, parameters were varied to evaluate which one achieve the better performance, local and global respectively. The selected parameters were: Harris corner function sensitivity factor (k), spatial variation (α) and temporal variance (β), all variables described in the Equation 4.

To evaluate our methodology, four different training scenarios (TS) for each implementation will be presented as it follows: TS1: $\alpha = 4$, $\beta = 2$ and $k = 0.01$;

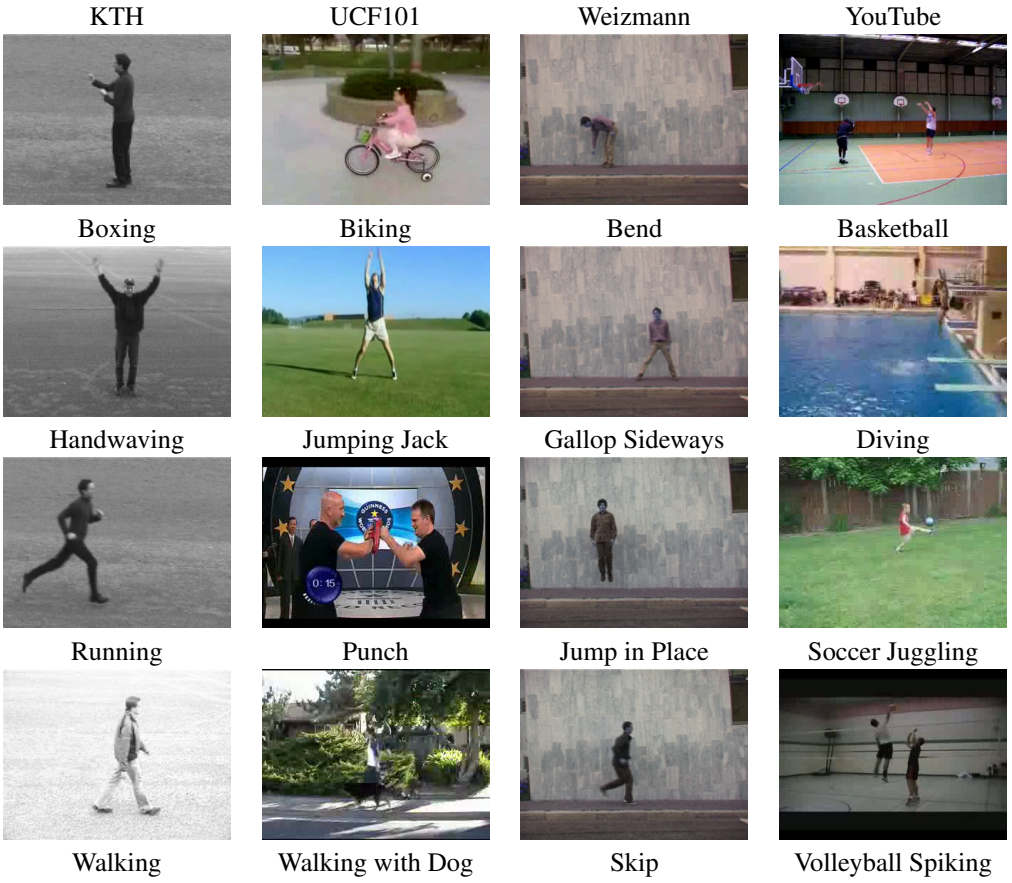


Table 1: Datasets samples.

TS2: $\alpha = 4$, $\beta = 2$ and $k = 0.05$; TS3: $\alpha = 9$, $\beta = 3$ and $k = 0.05$; TS4: $\alpha = 16$, $\beta = 4$ and $k = 0.05$.

After exhaustive evaluations, these training scenarios described above were selected from the best achieved results, improving our proposed methodology performance and denoting the direct influence of them on the MAP measure in action recognition process.

The achieved results are compiled, for each dataset and selected action, in confusion matrices found in Tables 5 to 5, where the best precision measure was highlighted.

In Tables 5 to 5, it is shown a MAP measure achieved from all classes based on the confusion matrices, summarizing the global results, presenting the results for C-STIP and V-STIP methodology, classified using the SVM schema described in Subsection 4, for each dataset.

Discussions

Due to the great amount of information that can be detailed using the confusion matrices, we opted to discuss the most relevant to evaluate the performance of the whole classification process given the proposed training scenarios.

For the first training scenario in Table 5, it is possible to notice that the process of classification of the dataset KTH, even with few false positives, was efficient to the majority of the videos in each class. This result was expected, since the KTH dataset is simpler and the human action is performed in clean environments with high contrast between the subject and the background, for all actions.

Using the C-STIP method, the classification for the actions *gallop* and *bend* was often incorrect. However, the V-STIP method has shown more suitable video classifications in this two classes, as a result of similar STIP variations in the lower body movements, generating a larger number of salience in this region. These variations occur in other scenarios with the same Weizmann dataset. In the training scenario 4, Table 5, there was a confusion between *handwave* and *box* due to lack of lateral variation by the subject, showing the influence of the spatial variation parameter (α) in classification results.

In the confusion matrices presented in Tables 5 to 5, it is possible to notice a recurrent confusion between the classes *punch* and *jumping jack* that can be explained by the upper body movement that is slightly alike between the selected videos. The V-STIP method was able to classify better than the C-STIP method, with the majority of correct classifications.

Method		C-STIP				V-STIP			
Dataset	Class	*box*	*handwave*	*run*	*walk*	*box*	*handwave*	*run*	*walk*
KTH	*box*	**42**	34	8	16	**50**	34.5	5	10.5
	handwave	30	**39**	16	15	34	**42**	13	11
	run	19	22	**47**	12	20	22	**51**	7
	walk	24	26	10	**40**	28	28	7	**37**
UCF101		*bike*	*jumpjack*	*punch*	*walkdog*	*bike*	*jumpjack*	*punch*	*walkdog*
	bike	**50**	16	21	13	**45**	31	12	12
	jumpjack	15	33	**35**	17	26	**47**	14	13
	punch	11	30	**43**	16	30	9	**44**	17
	walkdog	21	16	29	**34**	25	7	20	**48**
Weizmann		*bend*	*gallop*	*pjump*	*skip*	*bend*	*gallop*	*pjump*	*skip*
	bend	35	**35.5**	17.5	12	**52**	30	11	7
	gallop	20	**43**	20	17	29	**45.5**	20	5.5
	pjump	13	23	**37**	28	16	27	**53**	4
	skip	12	18	29	**41**	**36**	21	9	34
YouTube		*basket*	*dive*	*soccer*	*volleyb*	*basket*	*dive*	*soccer*	*volleyb*
	basket	23	31	**43**	3	14	36	**47**	3
	dive	5	**51**	41	3	6.5	**49**	43	2
	soccer	6	36	**54**	4	7	38	**52**	3
	volleyb	1	7	16	**76**	1	4	14	**81**

Table 2: Confusion matrix for TS1.

Method		C-STIP				V-STIP			
Dataset	Class	*box*	*handwave*	*run*	*walk*	*box*	*handwave*	*run*	*walk*
KTH	*box*	**43**	35	8.5	13.5	**52**	36	4	8
	handwave	26. 5	**34**	27.5	12	29	**42**	22	7
	run	16	19	**56**	9	21.5	23	**51**	5
	walk	24	27.5	7.5	**41**	30	31	4	**35**
UCF101		*bike*	*jumpjack*	*punch*	*walkdog*	*bike*	*jumpjack*	*punch*	*walkdog*
	bike	**52**	14	18	16	**49**	12	19	20
	jumpjack	14	**39**	33	14	15	33	**37**	15
	punch	10	37	**39**	14	11	29	**44**	16
	walkdog	14	17	32	**37**	13.5	14	30.5	**42**
Weizmann		*bend*	*gallop*	*pjump*	*skip*	*bend*	*gallop*	*pjump*	*skip*
	bend	23	**37**	24	16	**42**	34	13.5	10.5
	gallop	14	**34**	29	23	26	**34**	21	19
	pjump	10	22	**38**	30	14.5	20	**35.5**	30
	skip	9	19	33	**39**	16	17	29	**38**
YouTube		*basket*	*dive*	*soccer*	*volleyb*	*basket*	*dive*	*soccer*	*volleyb*
	basket	**44**	22	32	2	22	34	**42**	2
	dive	8	**49**	41	2	9	**52**	37	2
	soccer	9	36	**52**	3	12	38	**48**	2
	volleyb	2	5	18	**75**	2	3	16	**79**

Table 3: Confusion matrix for TS2.

Method		C-STIP				V-STIP			
Dataset	Class	*box*	*handwave*	*run*	*walk*	*box*	*handwave*	*run*	*walk*
KTH	*box*	**48**	38	5	9	**52**	38	3	6
	handwave	32	**46**	14	8	37	**43**	14	5
	run	18	23	**52**	7	22	23	**51**	4
	walk	26	**31**	12	30	31	28	8	**34**
UCF101		*bike*	*jumpjack*	*punch*	*walkdog*	*bike*	*jumpjack*	*punch*	*walkdog*
	bike	**65**	9	15	11	**57**	10	20	13
	jumpjack	10	34	**43**	14	16	27	**42**	15
	punch	6	31	**49**	14	8	26	**53**	14
	walkdog	12	17	29	**43**	10	14	27	**49**
Weizmann		*bend*	*gallop*	*pjump*	*skip*	*bend*	*gallop*	*pjump*	*skip*
	bend	**35**	34	17	14	**62**	22	8	9
	gallop	24	**49**	16	11	**44**	35	12	9
	pjump	14	26	**34**	26	24	23	**28**	24
	skip	10	20	31	**39**	23	16	25	**37**
YouTube		*basket*	*dive*	*soccer*	*volleyb*	*basket*	*dive*	*soccer*	*volleyb*
	basket	**63**	13	23	1	20	33	**45**	2
	dive	8	**51**	39	2	9	**53**	36	2
	soccer	11	35	**52**	2	13	37	**49**	2
	volleyb	1	5	13	**81**	2	4	12	**82**

Table 4: Confusion matrix for TS3.

Method		C-STIP				V-STIP			
Dataset	Class	*box*	*handwave*	*run*	*walk*	*box*	*handwave*	*run*	*walk*
KTH	*box*	**50**	37	5	8	**62**	27	4	7
	handwave	41	**47**	4	8	**48**	44	3	5
	run	15	20	**61**	4	29	20	**46**	5
	walk	29	**31**	10	30	35	22	7	**37**
UCF101		*bike*	*jumpjack*	*punch*	*walkdog*	*bike*	*jumpjack*	*punch*	*walkdog*
	bike	**62**	11	16	11	**53**	14	21	12
	jumpjack	15	30	**41**	13	19	29	**38**	14
	punch	13	27	**47**	13	13	22	**51**	14
	walkdog	17	17	31	**35**	14	15	32	**40**
Weizmann		*bend*	*gallop*	*pjump*	*skip*	*bend*	*gallop*	*pjump*	*skip*
	bend	**48**	35	10	7	**57**	31	6	6
	gallop	27	**40**	18	14	**42**	33	13	12
	pjump	15	20	**35**	31	28	18	**29**	25
	skip	12	16	28	**44**	14	15	32	**39**
YouTube		*basket*	*dive*	*soccer*	*volleyb*	*basket*	*dive*	*soccer*	*volleyb*
	basket	**60**	16	23	1	29	31	**39**	1
	dive	11	**47**	41	1	14	**46**	39	1
	soccer	14	34	**50**	2	16	35	**46**	2
	volleyb	3	6	9	**82**	3	6	8	**83**

Table 5: Confusion matrix for TS4.

Method	KTH	UCF101	Weizmann	YouTube
C-STIP	42%	40%	39%	51%
V-STIP	45%	46%	46%	49%

Table 6: TS1 - Mean Average Precision (MAP).

Method	KTH	UCF101	Weizmann	YouTube
C-STIP	42%	44%	33%	53%
V-STIP	54%	39%	37%	48%

Table 7: TS2 - Mean Average Precision (MAP).

Method	KTH	UCF101	Weizmann	YouTube
C-STIP	44%	47%	39%	61%
V-STIP	45%	46%	40%	51%

Table 8: TS3 - Mean Average Precision (MAP).

Method	KTH	UCF101	Weizmann	YouTube
C-STIP	47%	43%	42%	60%
V-STIP	47%	43%	39%	51%

Table 9: TS4 - Mean Average Precision (MAP).

All the datasets had videos with different duration, as presented in Subsection 4, and the proposed methodology was able to correctly classify the videos, even with the variable time, showing the robustness of the technique.

Using the MAP results in Tables 5 to 5 and the C-STIP method as first reference, the best results for KTH and Weizmann datasets were in scenario 4, with $\alpha = 16$, $\beta = 4$ and $k = 0.05$. For UCF101 and YouTube, scenario 3 have better performances, with $\alpha = 9$, $\beta = 3$ and $k = 0.05$. These were the best parameters adjustments for the aforementioned datasets.

With V-STIP as reference, the best parameter adjustment for KTH dataset is $\alpha = 16$, $\beta = 4$ and $k = 0.05$, that represents scenario 4. UCF101 and Weizmann best results were with scenario 1, where the parameters were $\alpha = 4$, $\beta = 2$ and $k = 0.01$. UCF101 can also be fit with $\alpha = 9$, $\beta = 3$ and $k = 0.05$. For YouTube dataset, there is a tie between two scenarios, TS3 and TS4, therefore it is possible to use $\alpha = 9$, $\beta = 3$ and $k = 0.05$ or $\alpha = 16$, $\beta = 4$ and $k = 0.05$.

6 CONCLUSIONS

Our goal is to evaluate state-of-the-art techniques of space-time interest points descriptors using a SVM classifier to recognize human actions in videos. Two descriptor implementations were proposed and applied in known datasets to classify actions, not taking into consideration the videos resolution and duration.

The proposal of using STIPs as descriptors in its two variations (Classic and Velocity adaptation) is valid, since the classification step was able to correctly classify the majority of the presented actions or classes, indicating that this work can be used as an alternative method to address the problem of human action recognition in videos.

Considering unclipped videos, that are videos with different actions occurring at the same time, it is possible as well to claim that the proposal can also classify the main action of the scene, once two tested datasets have, somewhat, this characteristic (UCF101 and YouTube).

The results achieved can be also be used as parameters directives for an optimization of the adjustment stage of future works.

Finally, further works will prioritize the diagonals of confusion matrices, avoiding false positives and improving global results.

7 REFERENCES

[BD01] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, Mar 2001.

[BFB94] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *Int. J. Comput. Vision*, 12(1):43–77, February 1994.

[BGS05] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *The Tenth IEEE International Conference on Computer Vision (ICCV'05)*, pages 1395–1402, 2005.

[BHHSV01] Asa Ben-Hur, David Horn, Hava T Siegelmann, and Vladimir Vapnik. Support vector clustering. *Journal of machine learning research*, 2(Dec):125–137, 2001.

[Bla99] M. J. Black. Explaining optical flow events with parameterized spatio-temporal models. In *IEEE Proc. Computer Vision and Pattern Recognition, CVPR'99*, pages 326–332, Fort Collins, CO, 1999. IEEE.

[CHMG12] Bhaskar Chakraborty, Michael B. Holte, Thomas B. Moeslund, and Jordi GonzÃ lez. Selective spatio-temporal interest points. *Computer Vision and Image Understanding*, 116(3):396 – 410, 2012. Special issue on Semantic Understanding of Human Behaviors in Image Sequences.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[DB97] James Davis and Aaron Bobick. The representation and recognition of action using temporal templates. pages 928–934, 1997.

[DOS15] Afshin Dehghan, Omar Oreifej, and Mubarak Shah. Complex event recognition using constrained low-rank representation. *Image and Vision Computing*, 42:13–21, 2015.

[DRCB05] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *Proceedings of the 14th International Conference on Computer Communications and Networks*, ICCCN '05, pages 65–72, Washington, DC, USA, 2005. IEEE Computer Society.

[DT05] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[GBS07] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, Dec 2007.

[GD96] D. M. Gavrila and L. S. Davis. 3-d model-based tracking of humans in action: a multi-view approach. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, pages 73–80, Jun 1996.

[GKRR05] Roman Goldenberg, Ron Kimmel, Ehud Rivlin, and Michael Rudzsky. Behavior classification by eigendecomposition of periodic motions. *Pattern Recogn.*, 38(7):1033–1043, July 2005.

[HS88] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Citeseer, 1988.

[Kan80] Takeo Kanade. Region segmentation: Signal vs semantics. *Computer Graphics and Image Processing*, 13(4):279 – 297, 1980.

[Lap04] Ivan Laptev. Local spatio-temporal image features for motion interpretation. 2004.

[Lap05] Ivan Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.

[LAS08] Jingen Liu, Saad Ali, and Mubarak Shah. Recognizing human actions using multiple features. 2008.

[LK81] An iterative image registration technique with an application to stereo vision. Lucas, Bruce D., and Takeo Kanade. 1981.

[LL03] Ivan Laptev and Tony Lindeberg. Space-time interest points. In *IN ICCV*, pages 432–439, 2003.

[LL04] Ivan Laptev and Tony Lindeberg. Velocity adaptation of space-time interest points. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 1, pages 52–56. IEEE, 2004.

[lLL06] Wei lwun Lu and James J. Little. Tracking and recognizing actions at a distance. In *in: ECCV Workshop on Computer Vision Based Analysis in Sport Environments*, pages 49–60, 2006.

[LLS09] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos "in the wild". In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1996–2003. IEEE, 2009.

[MS04] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[Roh94] K. Rohr. Towards model-based recognition of human movements in image sequences. *CVGIP: Image Underst.*, 59(1):94–115, January 1994.

[SI05] Eli Shechtman and Michal Irani. Space-time behavior based correlation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 405–412, June 2005.

[SLC04] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004.

[SZS12] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. 2012.

[WTVG08] Geert Willems, Tinne Tuytelaars, and Luc Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *European conference on computer vision*, pages 650–663. Springer, 2008.

[ZMI01a] L. Zelnik-Manor and M. Irani. Event-based video analysis. Technical report, Jerusalem, Israel, Israel, 2001.

[ZMI01b] Lihi Zelnik-Manor and Michal Irani. Event-based analysis of video. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–123. IEEE, 2001.

A method of core wire extraction from point cloud data of rebar

Koji Nishio
Osaka Institute of Technology
1-79-1, Kitayama
Hirakata, Osaka
JAPAN (573-0196),
Koji.a.nishio@oit.ac.jp

Noriyoshi Nakamura
Osaka Institute of Technology
1-79-1, Kitayama
Hirakata, Osaka
JAPAN (573-0196),
nakamura_noriyoshi@ggl.is.oit.ac.jp

Yuta Muraki
Osaka Institute of Technology
1-79-1, Kitayama
Hirakata, Osaka
JAPAN (573-0196),
Yuta.muraki@oit.ac.jp

Ken-ichi Kobori
Osaka Institute of Technology
1-79-1, Kitayama
Hirakata, Osaka
JAPAN (573-0196),
Kenichi.kobori@oit.ac.jp

ABSTRACT

In recent years, reinforced concrete has been widely used as a building material having high strength. However, when the arrangement of the reinforcing bars embedded in the inside are not correct, there is a problem that the strength drops greatly. Therefore, when use reinforced concrete, it is necessary to confirm whether the reinforcing bars are correctly arranged. In this paper, we propose a method of thinning of point clouds scanned by range scanner from reinforcing bars, and extracting the core wires of point clouds by introducing a concentration distribution function. This process moves points according to a gradient field of a concentration field. In addition, we conducted an experiment to confirm the effectiveness of the proposed method. As a result of the experiment, it was confirmed that the core wires can be extracted from the point clouds of the reinforcing bars.

Keywords

point cloud, thinning, core wire extraction, reinforced concrete and density distribution

1. INTRODUCTION

In recent years, reinforced concrete is widely used as a building material having high strength. However, reinforced concrete has a problem that the strength drops greatly when the number and arrangement of reinforcing bars embedded in the reinforced concrete are not correct. Therefore, in a construction using reinforced concrete, it is important to confirm whether the reinforcing bars are correctly arranged. Currently, the confirmation process is performed manually by the human eye, requiring a lot of time and possibly causing human error.

Therefore, in this paper, we propose a method to extract and reveal core wires of rebar from point cloud data scanned by a range scanner.

However, since many noise is included in the point cloud data, it is difficult to apply the previous core line extraction method. Therefore, in the proposed method, firstly, the influence of noise is reduced by thinning the point cloud data, and then the core line extraction is performed.

In the thinning process, we introduce a concentration distribution function to define a density field. This estimates the position where the core lines of the reinforcing bars exist and converges the point cloud data there.

In the process of extracting the core line, core line candidates are generated from the density field and classified into core or non-core wires using information of point cloud data thinned.

2. Extraction of core wires from the point clouds

In recent years, researches to extract core lines (skeletons) of shapes represented by point cloud data are widely performed. In Lee's method [Lee00a], a core extraction method from unorganized point cloud data is proposed. In this method, a point cloud is thinned by a simple algorithm using a local least squares method, and the result is extracted as a core line of a point cloud. In the method of least squares, there is a problem that it is necessary to appropriately set the range to which the processing is applied. However, in this method, the problem is solved by introducing a minimum spanning tree. In addition, by appling the cylindrical shape to the least squares method with the method of Luk'acs et al. [Luk98a], it is said that a good core extraction result can be obtained for a shape like a reinforcing bar. However, this method is very weak against noise and it is

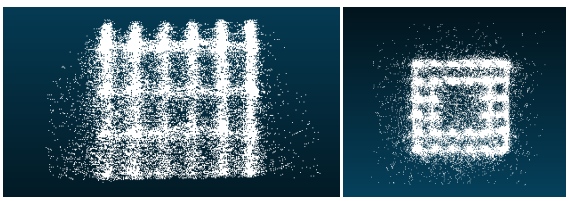
impossible to obtain good core line extraction results even with slight noise. There is also a problem that it does not correspond to a closed loop or a branched shape.

In the method of Bucksch et al. [Buc09a], the octree structure has been introduced to represent the point cloud data[Buc08a]. After that, the graph is simplified to extract the core line of the tree structure. This method is robust to noise and excellent in processing speed, however it cannot be applied to point cloud data which represents closed loop shape because it uses a tree structure.

Cao et al. [Cao10a] proposed a method to extract thin lines by shrinking the shape repeatedly using a smooth Laplacian filter. This method is applicable not only to point cloud data but also to surface data. In addition, it is possible to obtain a good extracted core line even for closed loop shape. Furthermore, it is relatively robust against noise and core line extraction is also possible from point cloud data that is partially missing if most point clouds have been acquired. However, generating the shape of the core line, it is necessary to set a plurality of weights with arbitrary values.

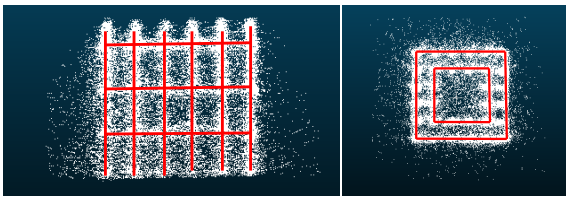
2.1 Problems of core wire extraction for reinforced bars

Since there are two problems in the point cloud data in this research, it is difficult to apply previous core line extraction methods for the point cloud data of reinforced bar. The first problem is that it is very noisy. The core line extraction processing in this research is targeted to point cloud data of rebar obtained by 3D scanner. However, as shown in Fig. 2.1, the point cloud data actually contains a lot of noise, and the surface of the shape cannot be acquired accurately. For this reason, the point cloud data handled in this study is different in shape from general point cloud data, and it is difficult to extract the core line by previous methods. The second problem is that the point cloud data of the rebar includes closed loop topology. In the point cloud data of reinforcing bars, as the reinforcing bars intersect each other, the point cloud data is distributed in a lattice pattern as shown in Fig. 2.2.



(A) Front view **(B) Plan view**

Figure 2.1 Point cloud of reinforced bars



(A) Front view **(B) Top view**

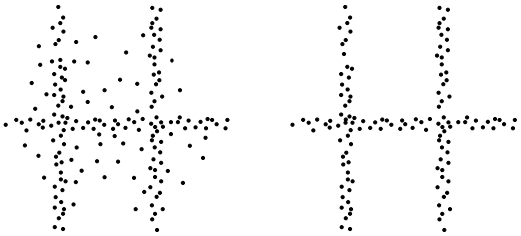
Figure 2.2 Point cloud with closed-loop distribution

3. Proposed method

In our method, a density distribution function is introduced in thinning processing to generate a density field and a vector field. Then, convergence of the point cloud according to the vector field is performed to thin the line. Thereafter, in the process of extracting the core line, intersection points of the reinforcing bars are detected from the density field and core wire candidates are generated. Then, core wires are extracted by classifying each core wire candidate into core or non-core wire from the distribution of the surrounding point cloud.

3.1 Thinning point cloud

As preprocessing for extracting the core line from the point cloud data of the reinforcing bars, thinning of the point cloud data is performed as shown in Fig.3.1.

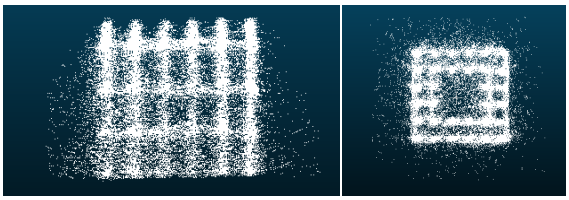


(A) Input **(B) Result**

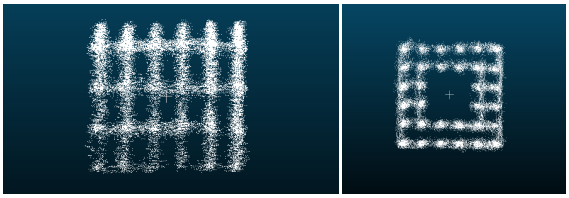
Figure 3.1 Thinning of point cloud

In this thinning process, the positions of the core lines of the reinforcing bars are estimated from the point cloud data of the reinforcing bars, and the point clouds are converged to thin lines.

In the proposed method, first, simple noise removal is performed as preprocessing. In the point cloud data scanned by the 3D scanner, points are scattered in the space, and the shape of the reinforcing bars cannot be clearly seen. However, the density of point cloud becomes high around the reinforcing bars. Therefore, it is assumed that reinforcing bars exist where the density of point clouds is relatively high, and points whose density is relatively low are removed as noise, as shown in Fig. 3.2.



(a) Before removal



(b) After removal

Figure 3.2 Noise reduction

3.1.1 Generation of density field

Because the core wire is located at the center of the reinforcing bar, in the proposed method, a concentration distribution function is introduced to estimate the position of this core line. The core lines of the reinforcing bars are near the center of the distribution of point cloud. Therefore, each point of point cloud is used as a source of the concentration distribution. The density field is generated by the density distribution function as shown in Fig. 3.3. The concentration distribution is obtained by Eq. 3.1. Where, x represents the coordinates for determining the concentration distribution value, r represents the range of the concentration distribution, and p_i represents the center coordinates of the concentration distribution, that is, the coordinates of the points of the point cloud.

$$C_m(x) = \sum_{i=1}^{n} \frac{\left(r - \|p_i - x\|\right)^2}{r^2} \quad (r \geq x)$$
$$C_m(x) = 0 \quad (r < x) \qquad (3.1)$$

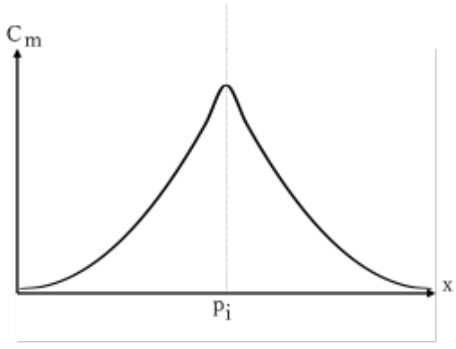
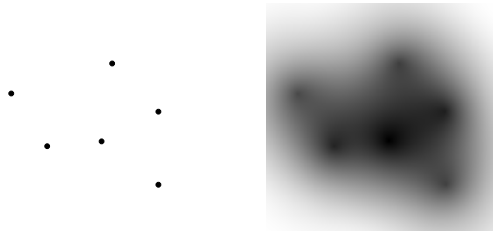


Figure 3.3 Concentration distribution function

Such concentration distribution is generated at each point of point cloud and values are added in a space where the density overlaps as shown in Fig. 3.4.



(a) Point cloud **(b) Concentration field**

Figure 3.4 Example of density field

3.1.2 Discretization of density field

Density field is discretized by using the voxel structure. At this time, normalization is performed so that the density value of the entire space becomes [0, 1]. In this paper, each voxel is referred by i, j, k sequentially allocated to the direction of each coordinate axis, and is expressed in the form of V (i, j, k). It is assumed that the density value and the gradient vector is represented by C_m (i, j, k) and G (i, j, k) at V (i, j, k) respectively.

3.1.3 Generating a vector field

Each point of the point cloud is moved according to the gradient of the density field, whereby the point cloud converges. The gradient vector G (i, j, k) at V (i, j, k) is obtained by the Eq. 3.2.

$$G_{(i,j,k)} = \left(\frac{\partial C_m(i,j,k)}{\partial i}, \frac{\partial C_m(i,j,k)}{\partial j}, \frac{\partial C_m(i,j,k)}{\partial k}\right) \qquad (3.2)$$

3.1.4 Moving points

In this process, the movement direction and the movement distance are calculated for each voxel, and the same movement is applied to all the points in each voxel. The movement direction D and the movement distance L are obtained as follows.

First, the direction of motion is determined by Eq. 3.3. Where, T is the sum of the gradient vectors stored in each voxel in a certain range centered on the current point, and σ should be set to a value greater than 1, so as not to be affected by the fine change of the gradient vector. Thereafter, the movement direction D is determined by normalizing the magnitude of T to 1.

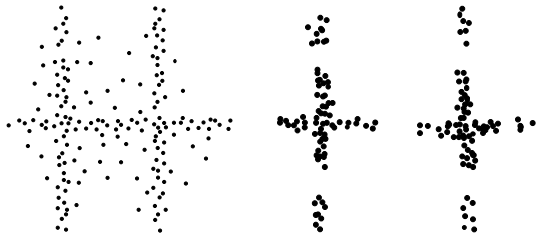
$$T = \sum_{l=i-\sigma}^{i+\sigma} \sum_{m=j-\sigma}^{j+\sigma} \sum_{n=k-\sigma}^{k+\sigma} G_{(l,m,n)}$$
$$D = \frac{T}{|T|} \qquad (3.3)$$

Next, the movement distance L is determined by Eq. 3.4. Where, w is a width of a voxel, and p is maximum distance each point can move represented by number

of voxels. The distance of movement is proportional to the magnitude of the gradient vector. Therefore, the magnitude of the gradient vector decreases at the voxel with a high density.

$$L = w\left|G_{(i,j,k)}\right|p \tag{3.4}$$

By continuing the process described above repeatedly until the maximum movement distance of the points becomes equal to or less than the threshold value, the point cloud is thinned. However, in this state, since the concentration value becomes very high at the point where the reinforcing bars cross each other, a vector field toward the intersection of the reinforcing bars is generated. Therefore, repeating the movement of points iteratively, there is a problem as shown in Fig. 3.5. Each point of point cloud concentrates at the intersection of the reinforcing bars and the point cloud becomes discontinuous in other places.



(A) input points **(B) result**

Figure 3.5 Example of the problem

As a solution to this problem, points are given spheres with radius r_s and if spheres intersect each other as shown in Fig. 3.6, a movement vector R_i which pushes back out of the radius is generated. By adding this vector to the movement process, we suppress the discontinuity of the point cloud. The vector R_i is obtained by the Eq. 3.5. Where, p is the moving point, p_i is the surrounding point, r_s is the radius of the sphere, and d is the distance between the two points.

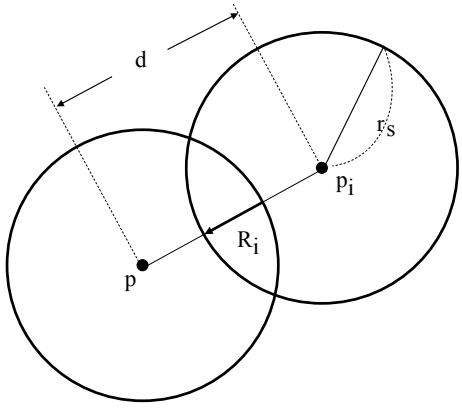


Figure 3.6 When the points are close together

$$R_i = \begin{cases} \dfrac{p-p_i}{|p-p_i|}(2r_s-d) & (d<2r_s) \\ 0 & (d \geq 2r_s) \end{cases} \tag{3.5}$$

3.2 Core wire extraction

In this process, firstly, the positions of the intersection of the reinforcing bars are detected by searching for all voxels from which voxels whose density values locally have local maximum values as shown in Fig. 3.7 and extracting them as intersection points of reinforcing bars. However, as can be seen from the figure, since the density of point clouds is nonuniform, many voxels where the density value has a maximum value exist.

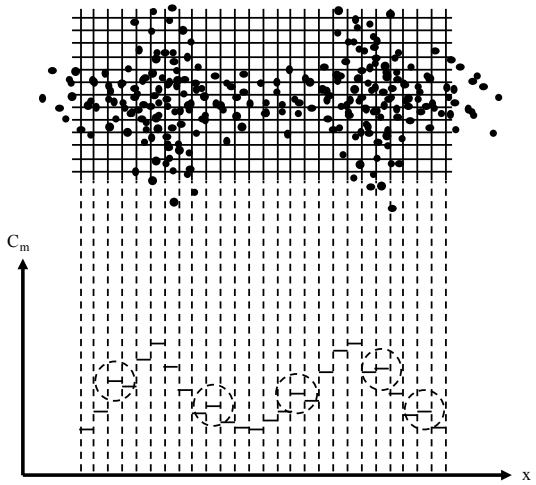
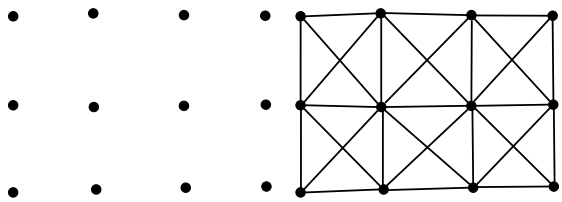


Figure 3.7 Local maximum points as an intersection points.

A voxel whose concentration value locally reaches a local maximum value is searched for and extracted as an intersection point of reinforcing bars.

Next, a core line candidate of the rebar is generated. The core line candidates are edges connecting the intersection points as shown in Fig. 3.8. However, these candidates contain wrong things. Therefore, it is necessary to classify these candidates into a core or non-core wire.



(a) Intersection points **(b) Core wire candidates**

Figure 3.8 Core wire candidate

It is determined whether or not each core wire candidate is the core wire of an actual reinforcing bar by using the position information of the point cloud.

As shown in Fig. 3.9, the points are densely distributed around the core line of the reinforcing bars, and distributed sparsely in other places. Therefore, by examining the distribution of points around each core line candidate, it is considered that classification of the core or non-core wire can be performed.

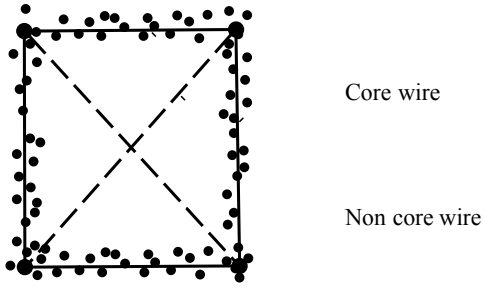
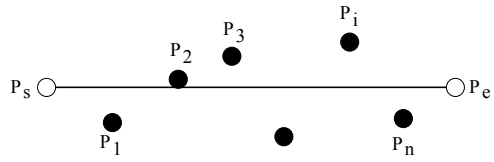
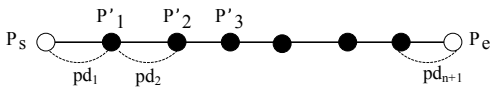


Figure. 3.9 Distribution of points along the wire

First, as shown in Fig. 3.10, each point of the point cloud is projected on the line of the nearest core line candidate. Where, P_i is the point before projection, P_i' is the point after projection, P_s-P_e is a core wire candidate, n is the number of points projected on the core line candidate and pd_i is a distance between neighboring points.



(A) A core candidate and points



(B) Points aligned on the core candidate

Figure 3.10 Projection of points

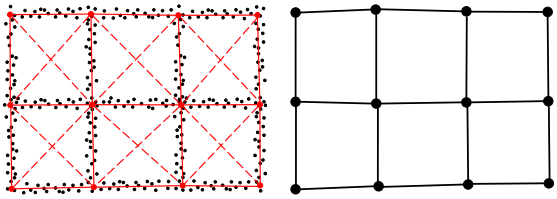
After that, the distance pd_{max} that is the maximum distance of pd_is, is obtained by the Eq. 3.7.

$$pd_{max} = max(PD)$$
$$PD = \{pd_i | i = 1,2,\cdots,n\} \quad (3.7)$$

If pd_{max} is smaller than th denoted by Eq. 3.8, the core line candidate is determined as a core line, and else it is determined as a non-core line. Where, E in the equation is the length of the core line candidate, and ρ is a parameter of [0, 1].

$$th = E \times \rho$$
$$E = |P_e - P_s| \quad (3.8)$$

The core line candidates classified by the above processing are shown in Fig. 3.11. In this figure, the core wire is indicated by a solid line, and the non-core wire is indicated by a broken line.

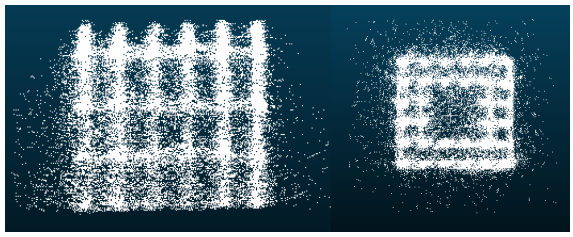


(a) Candidate classification **(b) Extracted core lines**

Figure 3.11 Classification and extraction result classification result

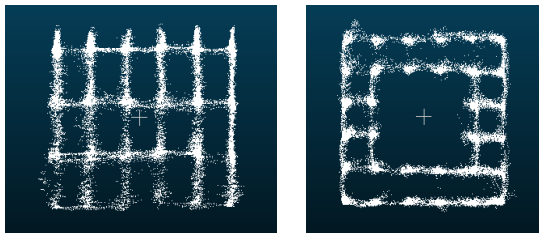
4. Results

The point cloud captured by a 3D scanner from the reinforced bars is shown in fig. 4.1 and the final thinning result generated by our method is shown in Fig. 4.2. In this experiment, we used 128×128×128 voxels, set r to 10 times the voxel width, σ to 13, p to 2.0, and r_s to 0.1.



(a) Front view **(b) Plan view**

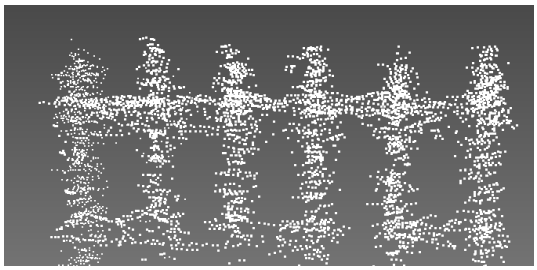
Figure 4.1 Point cloud of reinforced bars



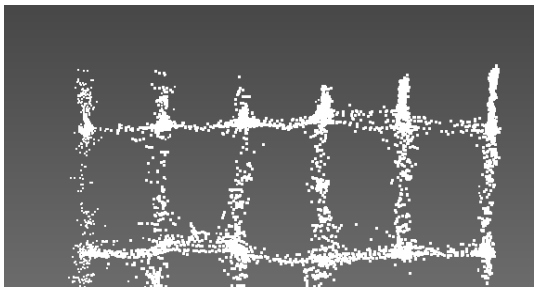
(a) Front view **(b) Plan view**

Figure 4.2 Result of Thinning Process

Fig. 4.3, 4.4 show enlarged images of the upper part and the bottom part of the result of thinning respectively.

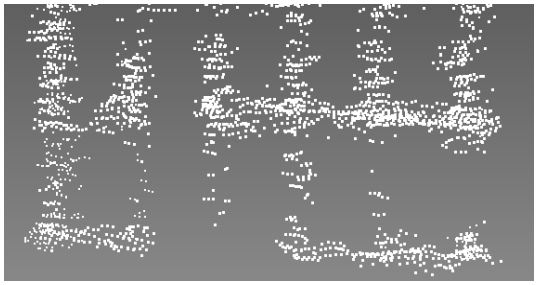


(a) Point cloud of reinforced bars



(b) Result of Thinning Process

Figure 4.3 Enlarged view of upper part

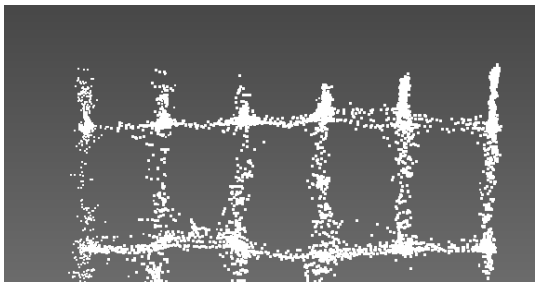


(a) Point cloud of reinforced bars

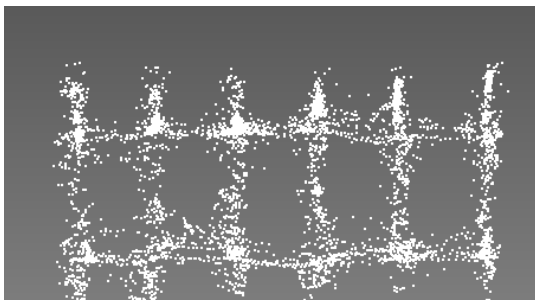
(b) Result of Thinning Process

Figure 4.4 Enlarged view of bottom part

Fig. 4.5 shows the difference between results due to the change in r.



(a) r = 10 voxel width

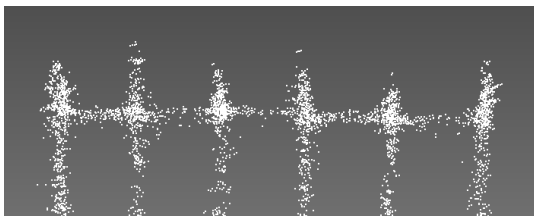


(b) r = 5 voxel width

Figure 4.5 Comparison of results by changing r

Furthermore, we verified the effectiveness of the countermeasure against the problem that the point cloud becomes discontinuous among the intersection points of reinforced bars. The result is shown in Fig. 4.6.

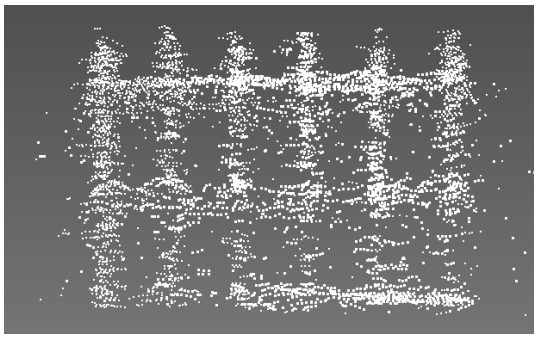
(a) Without countermeasure



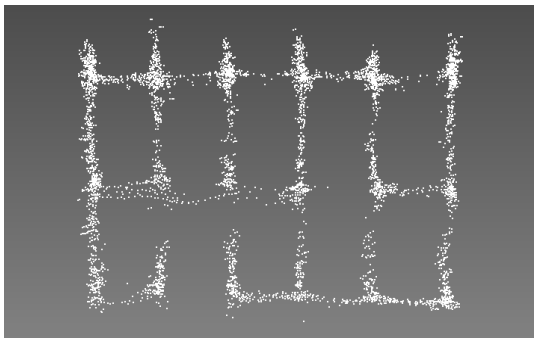
(b) With countermeasure
(r_s = 30% of voxel width)

Figure 4.6 Differences by radius r

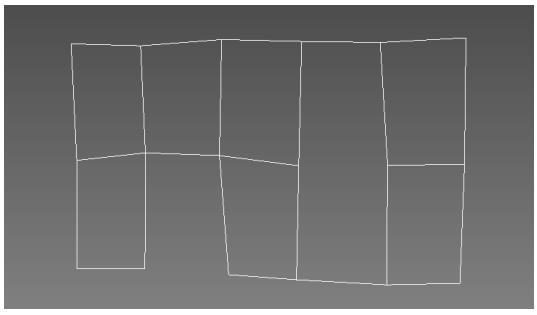
The result of core wire extraction is shown in Fig. 4.7.



(a) Point cloud of reinforced bars



(b) Result of Thinning Process



(c) Extracted core lines

Figure 4.7 Result of core line extraction

As shown in Fig. 4.2, we can see that the point cloud has been thinned. However, it can be confirmed that there is some discontinuous part. In particular, there are many discontinuities at the bottom of the data as shown in fig. 4.4. As you can see, since the distribution of point cloud is sparse, it can be seen that it is difficult to judge whether or not rebar exists by visual observation. Therefore, it is necessary to consider countermeasures by processing such as interpolation in addition to thinning.

Also, as shown in fig. 4.5, By reducing the radius of the concentration distribution, the number of convergent points increases. This is considered to be due to the fact that the influence of the density field generated from each point does not reach the surrounding points because the radius is too small, and the points does not move enough. Fig. 4.5(b) shows an example of density distribution where the size of the radius is too small.

From this result, it is considered that the size of an appropriate radius is about the radius of the reinforcing bar, but in the current method, an arbitrary value is used for the radius of the concentration distribution. Therefore, it is thought that future work is automatic parameter determination.

From Fig. 4.6, we can see the effectiveness by setting the sphere with the radius r_s. However, as introduction of this method makes it difficult for the point clusters to become thin, it is thought that it is necessary to improve so that thinning and discontinuity measures can be compatible.

As shown in Fig. 4.7, our method can extract the core wires almost correctly. However, there is some discontinuity of core wires. As a countermeasure to this problem, it is conceivable to use a different 3D scanner for scanning rebar and to change the scanning method.

5. Conclusion

In this paper, we proposed a method of extracting the core wires from the point cloud data of rebar.

In the proposed method, thinning of the point clouds of the reinforcing bars including many noises was well performed. In this thinning process, a concentration field for estimating the position of the core wire of the reinforcing bars is generated by introducing a concentration distribution function, and a vector field for converging the point cloud is constructed. Then, according to the gradient vector field, the point cloud was thinned by moving its points repeatedly. However, the convergence of the point cloud at the intersection of the reinforcing bars resulted in a problem that the point cloud where the reinforcing bars were supposed to be present occurred discontinuity. On the other hand, we considered spheres with certain radius at each point, and moved points so that spheres do not overlap and the point clouds do not concentrate to the intersection region of core wires.

In the core line extraction process, the intersection point of the reinforcing bars was detected from the generated density field, and the core line candidate of the rebar was generated from the information. When detecting the intersection point, in order to more accurately detect the intersection point, compared to the voxels existing around the current voxel, the voxel with the highest density value was taken as the

position of the intersection point of the rebar. Then, core line extraction was processed by classifying the generated core line candidates into core or non-core wires from the distribution information of the point cloud.

In the experiment, the effectiveness of core wire extraction by the proposed method was verified.

As a result, in the thinning process of the point cloud, it was possible to obtain a thinning result that suppresses the influence of noise by convergence of the point cloud.

We also confirmed how the change of each parameter influences the thinning result.

In the experiment of the core extraction process, it was confirmed that the core line can be extracted faithfully to the input data by comparing the point cloud thinning result with the core line candidates.

We think that it is necessary to perform interpolation processing when there are places where it is difficult to thin line in the input data. In addition, good processing results can be obtained by determining appropriate parameters through experiments. Therefore, it is considered that a future task is to devise a method of automatically setting parameters.

6. References

[Lee00a] Lee, In-Know, "Curve Reconstruction from Unorganized Points", Computer Aided Geometric Design, vol. 17, pp. 161-177, 2000.

[Luk98a] Lukács, G, Marshall, A. D., and Martin, R. R., "Geometric least-squares fitting of spheres, cylinders, cones, and tori", ECCV '98 Proceedings of the 5th European Conference on Computer Vision-Volume I - vol.1, pp.671-686, 1998.

[Buc09a] Bucksch, A., Lindenbergh, R. C. and Mententi, M., "SkelTre - fast skeltonization for imperfect point cloud data of botanic trees", 3DOR '09 Proceedings of the 2nd Eurographics conference on 3D Object Retrieval, pp.13-20, 2009.

[Buc08a] Bucksch, A. and Lindenbergh, R., "Campino - A skeletonization method for point cloud processing", ISPRS Journal of Photogrammetry and Remote Sensing, vol. 63, pp. 115 - 127, 2008.

[Cao10a] Cao, J. and Olson M., "Point Cloud Skeletons via Laplacian Based Contraction", SMI '10 Proceedings of the 2010 Shape Modeling International Conference, pp.187-197, 2010.

Using Genetic Algorithms to Estimate Local Shape Parameters of RBFs

Róbert Bohdal
Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics
Mlynská dolina
Slovakia, 842 48 Bratislava
robert.bohdal@fmph.uniba.sk

Mária Bohdalová
Comenius University in Bratislava
Faculty of Management
Odbojárov 10
Slovakia, 820 05 Bratislava
maria.bohdalova@fm.uniba.sk

Silvia Kohnová
Slovak University of Technology in Bratislava
Faculty of Civil Engineering
Radlinského 11
Slovakia, 810 05 Bratislava
silvia.kohnova@stuba.sk

ABSTRACT

Estimation of *design rainfall* in unobservable places is important in hydrological engineering. The aim of this paper is to use genetic algorithms to find the optimal global and local shape parameters of *radial basis functions* (RBFs) to create an interpolation model to estimate scaling exponents of short term rainfalls across selected regions of Slovakia. Scaling exponents can be used later to estimate rainfalls intensity in places without observations. In this paper, we have used interpolation methods based on RBFs to model interpolation surfaces. We investigate the properties of shape parameters in RBFs, and we test some methods for finding an optimal shape parameter. The choice of the best basis function along with the optimal shape parameter has a significant impact on the accuracy of the interpolation models which best approximate the real model. We have found that Hardy's multiquadrics interpolant with the optimal local shape parameters can be used for estimation the rainfall intensities in areas without direct observation.

Keywords

Radial basis functions, genetic algorithms, thin plate spline, shape parameters, rainfall

1 INTRODUCTION

In hydrology, engineering designers often face the problem of unreliable estimation of design short term rainfall intensities in unobservable places, or insufficiently long time series of observations. Regionalization methods are often used to solve this problem. These methods use available spatial information, and they consequently achieve reliable estimates of design values without direct observation [Koh16].

This paper gives a new method to estimate scaling exponents using interpolation methods based on *radial basis functions* (RBFs) with optimal global/local shape parameters. Our new method has not yet been applied to such extent. Our aim is to create an appropriate model for spatial estimation of rainfall intensities in places without direct observation for selected months across selected regions in Slovakia.

From the mathematical point of view, we need to construct the interpolation surface as long as we have a few of data points arranged in an irregular mesh. We attempt to achieve high accuracy of used interpolation surface for data points, in which we have no observations.

From literature, RBFs are known as a very popular interpolation tool for solving our designed problem due to their simplicity and ability to accurately approximate underlying multidimensional scattered data. This methodology is competitive, and it gives a high numerical accuracy when we compare it with other interpolating methods. Some of the most recent applications of RBFs include, for example, cartography, neural networks, medical imaging, numerical solution of partial differential equations [Flw09], [Dyn87], [Dyn89], [Isk03], [Ska13]. Interpolation methods based on RBFs are also used in BSDF interpolation [WKB12], [Ward14]. Hierarchical genetic algorithms are proposed in [TR15] to tackle the problem of automatic curve fitting. In this paper, authors have used only the Gaussian RBF, they have not compared the global shape parameter with the local

ones and as a test data they have used only analytically defined one dimensional functions. Rainfall approximation of the sparse rainfall data using RBFs is solved in [PC15], [PC16a] and [PC16b].

Many RBFs contain a free shape parameter c (see Table 1). The choice of the basis function and shape parameter has a significative impact on the accuracy of the method. In most papers authors choose this shape parameter by trial. Rippa [Rip99] has described a numerical algorithm to estimate the best value for shape parameter in radial basis interpolation using Leave One Out Cross Validation (LOOCV). Fasshauer and Zhang [FaZh07] used iterative approximative moving least squares approximation and RBFs pseudo-spectral method to estimate an appropriate shape parameter. Mongillo [Mon11] examined how to choose RBFs and shape parameters in a scattered data approximation.

Our paper introduces the idea of a global and local shape parameters estimation using genetic algorithms for scattered data. We discuss our method based on Leave Multiple Out Cross Validation (LMOCV) to estimate the best shape parameter(s).

The paper is organized as follows. Section 2 shortly describes the principle of the used interpolation methods based on RBFs. Section 3 presents our methodology and data. We discuss how an LMOCV strategy can be used in the context of finding the optimal shape parameter(s). Finally, section 4 presents our results. A comparison of the LOOCV and LMOCV method has been made. We have used root-mean-square error (RMSE) to compare selected interpolation methods based on RBFs.

2 INTERPOLATION METHODS BASED ON RBFS

The research of RBFs helps us understand how we can use these functions to solve practical problems. RBFs are preferred for image warping, geodesy, geography, digital terrain modeling, hydrology, etc. A good review of the theory of RBFs is given by Hardy [Har90], Powell [Pow91].

Let us have a set $\mathscr{X}$ of N different input points $\mathscr{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_N \mid \mathbf{x}_i \in \mathbb{R}^2\}$ with rainfall intensity values $\mathscr{F} = \{f_1,\ldots,f_N \mid f_i \in \mathbb{R}\}$. We search for such function $S : \mathbb{R}^2 \to \mathbb{R}$, for which the interpolation conditions are true:

$$S(\mathbf{x}_i) = f_i,\ i = 1,\ldots,N. \tag{1}$$

We can write the interpolation function $S(\mathbf{x})$ in the following form [Rip99]:

$$S(\mathbf{x}) = \sum_{i=1}^{N} \lambda_i \Phi(\|\mathbf{x}-\mathbf{x}_i\|) \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^2$, $\Phi(r)$ is a fixed real-valued RBF and $\|\cdot\|$ denotes the Euclidean norm.

The solution of the above interpolation conditions (1) is equivalent to the solution of a linear system of equations:

$$\mathbf{A}\cdot\boldsymbol{\lambda} = \mathbf{f},\ \mathbf{A} = \mathbf{A}_{i,j} = \Phi(\left\|\mathbf{x}_j - \mathbf{x}_i\right\|) \tag{3}$$

for the vector $\boldsymbol{\lambda} \in \mathbb{R}^2$ of unknown coefficients. This interpolation problem is solvable if and only if matrix $\mathbf{A}$ is nonsingular. The general conditions on $S(\mathbf{x})$ which guarantee nonsingularity of $\mathbf{A}$ are given in [Mic86], and they can be checked for many radial basis functions. In particular, these conditions are fulfilled for the choices of the function $\Phi(r)$ given in Table 1.

Radial basis function	$\Phi(r)$
Polyharmonic splines (PHS)	r^2
Thin plate splines (TPS)	$(1/2)r^2\log r^2$
Gauss function (GAUSS)	$e^{-r^2/2c^2}$
Hardy's multiquadric (HMQ)	$\sqrt{c^2+r^2}$
Inverse multiquadric (IMQ)	$1/\sqrt{c^2+r^2}$
Inverse quadric (IQ)	$1/(c^2+r^2)$

Table 1: Commonly used types of radial basis functions

The RBFs listed in Table 1 contain a shape parameter c that must be specified by the user. It is well known [Fra82], [Rip99], [Mon11] that the accuracy of the RBFs interpolants depends heavily on the choice of the parameter c. A smaller value of the shape parameter c corresponds to a surface with a higher curvature and a higher value of the shape parameter c corresponds to a flatter surface with a smaller curvature. The global shape parameter influences the whole surface, but local shape parameters influence the shape of the surface only in the neighborhood of each interpolated point.

2.1 Thin plate splines

One of the most commonly used interpolation methods based on RBFs is *thin plate splines* method. This method adds a polynomial term into equation (2) and does not contain any shape parameter.

We can write the TPS interpolating function $S(\mathbf{x})$ in the form [Fog96]:

$$S(\mathbf{x}) = S(x,y) = c_1 + c_2x + c_3y + \frac{1}{2}\sum_{i=1}^{N}\lambda_i r_i^2 \log(r_i^2), \tag{4}$$

where $[x,y] \in \mathbb{R}^2$, $r_i^2 = (x-x_i)^2 + (y-y_i)^2$ and $c_1,\ c_2,\ c_3,\ \lambda_i \in \mathbb{R}$ are unknown coefficients. The unknown values $\lambda_i,\ i = 1,\ldots,N$, have to satisfy the boundary conditions:

$$\sum_{i=1}^{N}\lambda_i = 0,\ \sum_{i=1}^{N}\lambda_i\mathbf{x}_i = \mathbf{0}. \tag{5}$$

Applying interpolation conditions (1) together with boundary conditions (5), we can compute the unknown values using a system of equations:

$$\mathbf{A}\cdot\mathbf{L}=\mathbf{F},$$

where

$$\mathbf{A}=\begin{pmatrix} 0 & 0 & 0 & 1 & 1 & \cdots & 1 \\ 0 & 0 & 0 & x_1 & x_2 & \cdots & x_n \\ 0 & 0 & 0 & y_1 & y_2 & \cdots & y_n \\ 1 & x_1 & y_1 & 0 & r_{21}^2\log(r_{21}^2) & \cdots & r_{n1}^2\log(r_{n1}^2) \\ 1 & x_2 & y_2 & r_{12}^2\log(r_{12}^2) & 0 & \cdots & r_{n2}^2\log(r_{n2}^2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & y_n & r_{1n}^2\log(r_{1n}^2) & r_{2n}^2\log(r_{2n}^2) & \cdots & 0 \end{pmatrix}$$

$$\mathbf{L}=\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \lambda_1/2 \\ \lambda_2/2 \\ \vdots \\ \lambda_N/2 \end{pmatrix},\ \mathbf{F}=\begin{pmatrix} 0 \\ 0 \\ 0 \\ f_1 \\ f_2 \\ \vdots \\ f_N \end{pmatrix}$$

and $r_{ij}^2=r_{ji}^2=(x_j-x_i)^2+(y_j-y_i)^2$.

2.2 Hardy's multiquadrics with local shape parameters

This method is very similar to the previous method, but it uses different RBFs, does not have a polynomial term and uses local shape parameters. For our interpolation problem, we obtain the following interpolation function:

$$S(\mathbf{x})=S(x,y)=\sum_{i=1}^{N}\lambda_i\sqrt{c_i^2+r_i^2}. \qquad (6)$$

The local shape parameters c_i significantly change the shape of the resulting interpolation surface. In general, a smaller value of the parameter c_i creates so-called "sharp extremes" at point $\mathbf{x}_i$ in the graph of the function, while its greater value "smoothes" the function. Using local shape parameters instead of one global parameter allows not only to change the shape of the interpolation surface at each interpolation point but also can increase accuracy of the created interpolation model. Introducing local shape parameters has one drawback - matrix that is created by RBFs (see equation (3)) is not symmetric anymore, which leads to the problem of finding optimal local shape parameters c_i using standard optimization methods.

3 METHODOLOGY

Different RBFs and their different shape parameters give different interpolation surfaces for the same data set. Finding the best RBF and the best estimation of the shape parameter(s) that produces the most accurate results is one of the topics of our paper. In this section we investigate *cross validation methods* for optimizing the shape parameter(s) with respect to the error in interpolation methods based on RBFs.

We decide to exclude at most K data points in one step (interpolation function calculation) from a data set with sample size N (number of interpolation points), where K corresponds to approximately 10% of the sample size N.

3.1 Cross validation methods

Cross validation methods are used for evaluating the accuracy of the created model (e.g. interpolation function) by splitting the input data set into validation and training data. The model is created from the training data so that it fits the validation data with some small error.

In practical use, there are several rounds of cross validations using different partitions of the data set. The resulting statistical measure giving the accuracy of the model is given as the average of the errors calculated in the individual rounds.

Let M (number of rounds) be the number of the sets of points indices of the excluded points. For LOOCV method, we have $M=N$ and for LMOCV method, we have M given in advance such that $M>N$.

Leave one out cross validation (LOOCV)

LOOCV method uses 1 element as validation data and $N-1$ remaining elements as training data, from which we get N singleton sets $I_p=\{p\},\ p=1,\ldots,N$ of indices of excluded points for model determination.

Leave multiple out cross validation (LMOCV)

LMOCV method uses a random number of elements for the validation data, while their number is limited by in advance given value K.

Let set $I_p=\left\{{}^pi_1,\ldots,{}^pi_{n_p}\right\}$ denote the p-th set of indices of the excluded data points for $p=1,\ldots,M$, where ${}^pi_{n_k}$ is a random integer, $1\leq{}^pi_{n_k}\leq N$, while cardinality $\left|I_p\right|$ (number of elements) of the set of indices varies from 1 to K.

3.2 Shape parameter estimation

Let $S^{(p)}(\mathbf{x})$ be the p-th interpolant of the reduced data set obtained by removing n_p points $\mathbf{x}_{{}^pi_1},\ldots,\mathbf{x}_{{}^pi_{n_p}}$ and the corresponding data values $f_{{}^pi_1},\ldots,f_{{}^pi_{n_p}}$ from the original data set. Our algorithm estimates the

shape parameter c by minimizing the error vectors ${}^{p}\boldsymbol{\varepsilon} = ({}^{p}e_1,\ldots,{}^{p}e_{n_p})^{\mathsf{T}}$, where

$$ {}^{p}e_j = S^{(p)}(\mathbf{x}_j) - f_j,\ j \in I_p,\ p = 1,\ldots,M. \qquad (7) $$

The error vectors are calculated by using standard LOOCV or our proposed LMOCV method. We take RMSE as a measure of the quality how well the interpolation function calculated from reduced data set fits the interpolation function created from all data points.

RMSE for LOOCV method is defined by:

$$ \mathrm{RMSE}(c) = \sqrt{\frac{\sum_{j=1}^{N}\left[S^{(p)}(\mathbf{x}_j) - f_j\right]^2}{N}}, \qquad (8) $$

RMSE for LMOCV method is defined by:

$$ \mathrm{RMSE}(c) = \sqrt{\frac{\sum_{p=1}^{M}\left(\sum_{j\in I_p}\left[S^{(p)}(\mathbf{x}_j) - f_j\right]^2\right)}{\sum_{p=1}^{M}|I_p|}}. \qquad (9) $$

Global shape parameter estimation

The optimal value of the global shape parameter c is defined as the value of c that minimizes RMSE(c). Any standard numerical estimation method can be used for finding the optimal shape parameter.

Local shape parameters estimation

Both previously defined RMSE measures can be used for local shape parameters estimation, with the only difference that vector $\mathbf{c} = (c_1,\ldots,c_N)$ of the local shape parameters c_i (see Section 2.2) is used instead of one global parameter c.

We use a genetic algorithm because standard optimization methods like quasi-Newton methods are not robust enough when searching the optimal vector of local shape parameters.

4 DATA AND EMPIRICAL RESULTS

In order to demonstrate the functionality of our LMOCV method for computing the interpolation function, we have used 5 sample datasets that consist of scaling exponents of maximum rainfall intensities with duration between 5 and 1440 minutes (short–term rainfall) for the warm season, from April to September in 34 rain gauge stations measured in Slovakia. The rain gauge stations are summarized in Table 2 and the area of the three regions is displayed in Figure 1.

During April, it was possible to measure rainfall only for $N = 29$ rainfall stations, while in other months we have recorded the scale exponent for $N = 34$ rainfall stations.

Region	Rain gauge stations
1	Myjava, Senica, Kuchyňa - Nový Dvor, Jaslovské Bohunice, Oravská Lesná, Čadca, Piešťany, Prievidza
2	Bratislava - Koliba, Bratislava - letisko, Nitra - Veľké Janíkovce, Telgárt, Sliač, Boľkovce, Dolné Plachtince, Bzovík, Kamenica nad Cirochou, Somotor, Rožňava, Lom nad Rimavicou, Štós - kúpele, Moldava nad Bodvou, Hurbanovo, Košice, Liptovská Osada
3	Javorina, Červený Kláštor, Poprad, Švedlár, Tatranská Lomnica, Medzilaborce, Liptovský Hrádok, Štrbské Pleso, Jakubovany

Table 2: Selected Slovakia regions with the rain gauge stations distribution

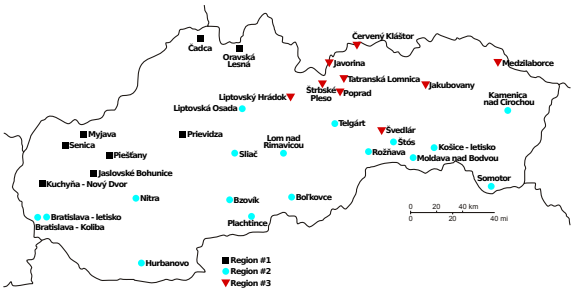


Figure 1: Map of the three selected regions and rain gauge stations

For our LMOCV method, we set $K = 4$, the maximum number of excluded data points in one step of interpolation function $S^{(p)}(\mathbf{x})$ calculation. We have created $M = 120$ sets I_p of excluded indices of rainfall stations.

We have discovered that searching the local shape parameters using classical optimization methods like quasi-Newton methods is inappropriate because these methods have found a local minimum instead of the global minimum. Therefore, we have decided to use genetic algorithms to find the global minimum. We have used the GA (see [Scr11], [Scr16]) and the GENOUD (see [Meb11]) genetic algorithm which are available as packages for **R** (**R** is a programming language and software environment for statistical computing). These packages combine the genetic algorithms approach with the standard optimization approach using BFGS optimization method and others.

We present the results of numerical experiments involving interpolation of the given datasets by polyharmonic splines, thin plate splines, Gaussian function, Hardy's multiquadric, inverse multiquadric and inverse quadric interpolants. We have compared the LOOCV and LMOCV method for excluding data points. Some selected results of the obtained interpolation surfaces for various datasets and various interpolation methods are shown in Figures 4–8.

Estimating the global shape parameter c for Hardy's multiquadrics is not suitable in general because of a very high curvature (peaks) at input points $\mathbf{x}_i$ on the created interpolation surfaces (see Figure 4). This undesirable shape is caused by the fact that the estimated shape parameter c is zero because of numerical instability of calculations in the optimization process. Consequently, we decide to find local shape parameters $\mathbf{c} = (c_1, \ldots, c_N)$ using genetic algorithms and with the LMOCV method, see Figure 5b.

In case of the inverse multiquadrics (see Figure 6) and other RBFs from Table 1, we do not face a problem with finding the optimal value of the global shape parameter. Except Hardy's multiquadrics, there is no problem with numerical instability for other RBFs. However, their accuracy is worse than that of Hardy's multiquadrics.

An example of interpolation surface created by the TPS method which does not need to estimate shape parameters is shown in Figure 7.

Table 3 and Figure 2 present RMSE calculated according to formula (8) and formula (9) for the optimal (best) global shape parameter c for interpolation methods based on RBFs from Table 1. We can see that the best accuracy is obtained for Hardy's multiquadric interpolant, the thin plate spline is second in order. The worst results have been achieved for the Gaussian RBF. As we can see, the LMOCV method gives smaller RMSE in comparison to the LOOCV method.

Figure 3 shows dependence between the RMSE and the global shape parameter c for dataset *June–July* while using Hardy's multiquadrics interpolation function. The LOOCV method gives the optimal value of the shape parameter c equal to 0.0854, and the LMOCV method gives the optimal value of the shape parameter c equal to 0.0621.

In estimating the global shape parameter for the dataset *April*, we would not be able to find its appropriate value due to numerical instability (see Figure 4). Even when we have used Q-R decomposition and SVD method for matrix equation calculation, we have experienced the same problem. In the optimization procedure, we have tried many statistical measures for the error vectors computation (MSE, MAE, MAPE, MASE and SMAPE) but we have not obtained acceptable results. We have decided to find optimal local shape parameters instead of the global parameter using the LOOCV and our proposed LMOCV method.

The LOOCV method for local shape parameter estimation gives low RMSE values (see Table 4), but the obtained function often oscilates (see Figure 5a) and has extreme values at the surface border. Using the LMOCV method, the RMSE is slightly higher, but the surface appears to be normal (see Figure 5b). Based on the above experiments, we propose that the value of the parameter c can be estimated by minimizing RMSE(c) using the LMOCV method.

5 CONCLUSION

We have found that the optimization procedure for estimating the global shape parameter of Hardy's multiquadrics interpolation function gives approximately zero value for many data sets. This zero value is unacceptable because it creates an inappropriate surface shape. Therefore, we have decided to use and estimate the local shape parameters. Classical optimization methods for the local shape parameters estimation is inapropriate because these methods rapidly converge to a local optimum. We have consequently decided to use two genetic algorithms - GA and GENOUD. Both algorithms combine genetic algorithms with the standard optimization method BFGS. We have also found that optimization process in the GENOUD package converges faster than in the GA package. Because standard LOOCV (Leave One Out Cross Validation) method for the model creation did not give good results for local shape parameters estimation, we have proposed the LMOCV (Leave Multiple Out Cross Validation) method.

Figure 5 and Table 4 show that the use of the optimal local shape parameters creates a smooth surface and gives lower RMSE values than the use of one global shape parameter. We conclude Hardy's multiquadrics interpolant with local shape parameters calculated using our proposed LMOCV method can be subsequently used in estimating the rainfall intensities in Slovakia, especially in areas without direct observation.

6 ACKNOWLEDGEMENTS

This research has been supported by the Slovak Research and Development Agency under contract No. APVV 15-0497 and by the Slovak Grant Agency VEGA under the project no 1/0710/15. The authors gratefully acknowledge to anonymous reviewers for their comments and recommendations.

7 REFERENCES

[BZ95] Billing, S.,A. and Zheng, G.,L. Radial Basis Function Network Configuration Using Genetic Algorithms. Neural Networks, Vol.8, No. 6, pp. 877–890, 1995.

[Dyn87] Dyn N. Interpolation of scattered data by radial functions, In: Topics in Multivariate Approximation, (Eds. Chui C.K., Schumaker L.L. and Utreras F.I.), Academic Press, New York, pp. 47–61, 1987

[Dyn89] Dyn N. Interpolation and approximation by radial and related functions, (Eds. Chui C.K., Schumaker L.L. and Ward J.D.), Academic Press, New York, pp.211–234, 1989

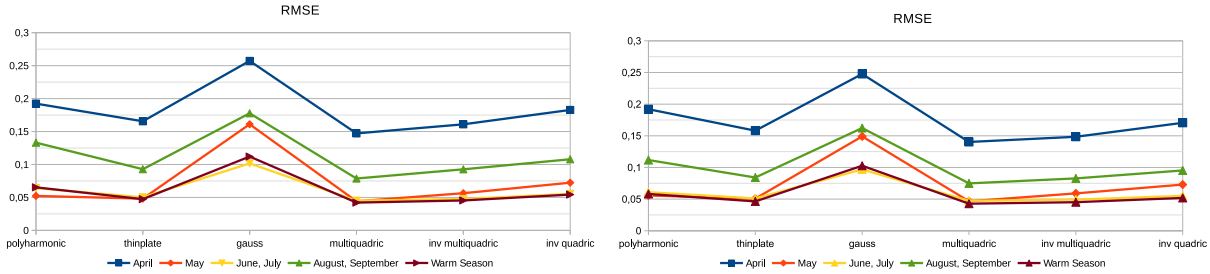
	PHS	TPS	GAUSS	HMQ	IMQ	IQ
Apr	0.192	0.166	0.257	0.147	0.161	0.183
May	0.052	0.048	0.161	0.044	0.056	0.072
Jun-Jul	0.064	0.050	0.102	0.045	0.048	0.055
Aug-Sep	0.133	0.093	0.178	0.079	0.093	0.108
Jun-Sep	0.065	0.047	0.112	0.042	0.045	0.054

(a) LOOCV method

	PHS	TPS	GAUSS	HMQ	IMQ	IQ
Apr	0.192	0.158	0.248	0.140	0.148	0.171
May	0.056	0.051	0.149	0.047	0.059	0.073
Jun-Jul	0.061	0.051	0.097	0.047	0.049	0.055
Aug-Sep	0.112	0.084	0.162	0.075	0.083	0.095
Jun-Sep	0.058	0.046	0.103	0.043	0.045	0.052

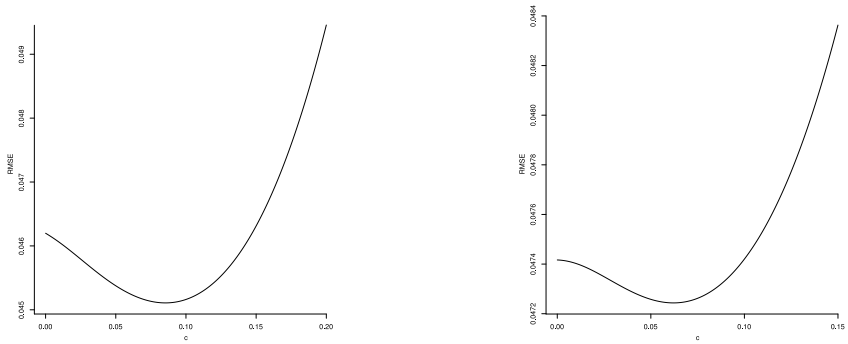
(b) LMOCV method

Table 3: RMSE comparison for various datasets and various RBFs with the global shape parameter



(a) LOOCV method (b) LMOCV method

Figure 2: RMSE comparison of two methods for points exclusion for various RBFs with the global shape parameter



(a) LOOCV method, $c = 0.0854$ (b) LMOCV method, $c = 0.0621$

Figure 3: RMSE vs. the value of the global shape parameter c of HMQ interpolation function for dataset *June–July*

	HMQ
Apr	7.546E-05
May	3.139E-09
Jun-Jul	2.229E-02
Aug-Sep	4.797E-05
Jun-Sep	9.438E-06

(a) LOOCV method

	HMQ
Apr	8.291E-02
May	2.625E-02
Jun-Jul	2.717E-02
Aug-Sep	5.015E-02
Jun-Sep	2.454E-02

(b) LMOCV method

Table 4: RMSE for HMQ interpolation function with the local shape parameters

[FaZh07] Fasshauer, G.E. and Zhang, J.G. On choosing optimal shape parameters for RBF approximation Numerical Algorithms. 2007, 45, pp. 345–368. DOI: 10.1007/s11075-007-9072-8

[Flw09] Flyer, N. and Wright, G. B. A radial basis function method for the shallow water equations on a sphere. Proc. Royal Society A. 2009, 465, pp. 1949–1976. DOI: 10.1098/rspa.2009.0033. Published 22 April 2009

[Fog96] Fogel, D. and L. Tinney. Image registration using multiquadric functions, the finite element method, bivariate mapping polynomials and thin plate spline, tech. rep., National Center for Geographic Information and Analysis, 1996.

[Fra82] Franke, R. Scattered data interpolation: tests of some methods, Math. Comp. 38, pp. 181–200, 1982.

[Har90] Hardy,R.L. Theory and applications of the multiquadric-biharmonic method, Comput. Math. Appl. 19, pp. 163–208, 1990.

[Isk03] Iske, A. Radial basis functions: basics, advanced topics and mesh free methods for Transport Problem. Seminar of Mathematics, pp. 247–274, 2003

[Koh16] Kohnová, S., Bohdalová, M., Bohdal, R. and Ochabová, K. To the Applicability of Radial Basics Functions for the Interpolation of short Term Rainfall Scaling Exponents in Slovakia. Acta Hydrologica Slovaca, Vol.17 (2), pp. 243–251, 2016.

[Meb11] Mebane, W.,Jr. and Sekhon, J. S. Genetic Optimization Using Derivatives: The rgenoud package for R. Journal of Statistical Software, 42(11), pp.1–26, 2011.

[Mic86] Micchelli, C.A. Interpolation of scattered data: distance matrices and conditionally positive definite functions, Constr. Approx. 2, pp. 11–22, 1986.

[Mon11] Mongillo, M. Choosing Basis Functions and Shape Parameters for Radial Basis Function Methods. SIAM Undergraduate Research Online (SIURO), Volume 4. http://dx.doi.org/10.1137/11S010840, 2011

[PC15] Patané, G., Cerri, A., Skytt, V., Pittaluga, S., Biasotti, S., Sobrero, D., Dokken, T. and Spagnuolo, M. A comparison of methods for the approximation and analysis of rainfall fields in environmental applications. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 2015. Volum II.(3) Suppl. W5 pp. 523–530

[PC16a] Patané, G., Cerri, A., Skytt, V., Pittaluga, S., Biasotti, S., Sobrero, D., Dokken, T. and Spagnuolo, M. Applications to Surface Approximation and Rainfall Analysis. I: Heterogenous Spatial Data Fusion, Modeling, and Analysis for GIS Applications. Morgan & Claypool Publishers. pp. 79-98, 2016a.

[PC16b] Patané, G., Cerri, A., Skytt, V., Pittaluga, S., Biasotti, S., Sobrero, D., Dokken, T. and Spagnuolo, M. Comparing Methods for the Approximation of Rainfall Fields in Environmental Applications. ISPRS journal of photogrammetry and remote sensing (Print), 2016.

[Pow91] Powell, M.J.D. The theory of radial basis function approximation in 1990, in: Advances in Numerical Analysis, Vol. II: Wavelets, Subdivision Algorithms and Radial Functions, ed. W. Light (Oxford University Press, Oxford, UK), pp. 105–210, 1991.

[Rip99] Rippa, S. An algorithm for selecting a good value for the parameter c in radial basis function interpolation. In: Advances in Computational Mathematics 11, pp. 193–210, 1999.

[Ska13] Skala, V. Fast Interpolation and Approximation of Scattered Multidimensional and Dynamic Data Using Radial Basis Functions. WSEAS Transactions on Mathematics. Issue 5, Volume 12, May 2013

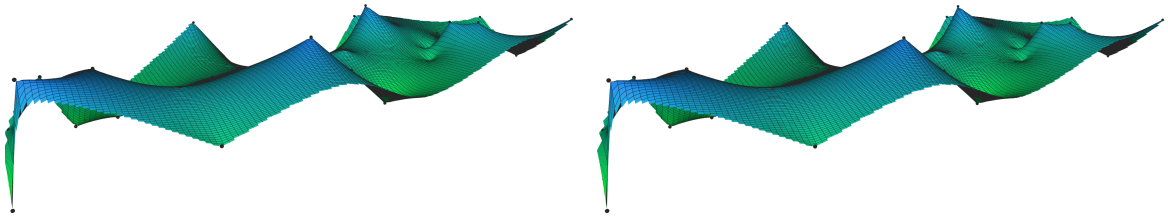
[Scr11] Scrucca, L. GA: A Package for Genetic Algorithms in R. Journal of Statistical Software, 53(4), pp.1–37. URL http://www.jstatsoft.org/v53/i04/, 2013.

[Scr16] Scrucca, L. On some extensions to GA package: hybrid optimisation, parallelisation and islands evolution. Submitted to R Journal. Pre-print available at arXiv URL http://arxiv.org/abs/1605.01931, 2016.

[TR15] Trejo–Caballero, G., Rostro–Gonzalez, H., Garcia–Capulin, V., Ibarra–Manzano, O. G., Avina–Cervantes,J. G. and Torres–Huitzil, C. Automatic Curve Fitting Based on Radial Basis Functions and a Hierarchical Genetic Algorithm. Mathematical Problems in Engineering. 14 pp., http://dx.doi.org/10.1155/2015/731207, 2015.

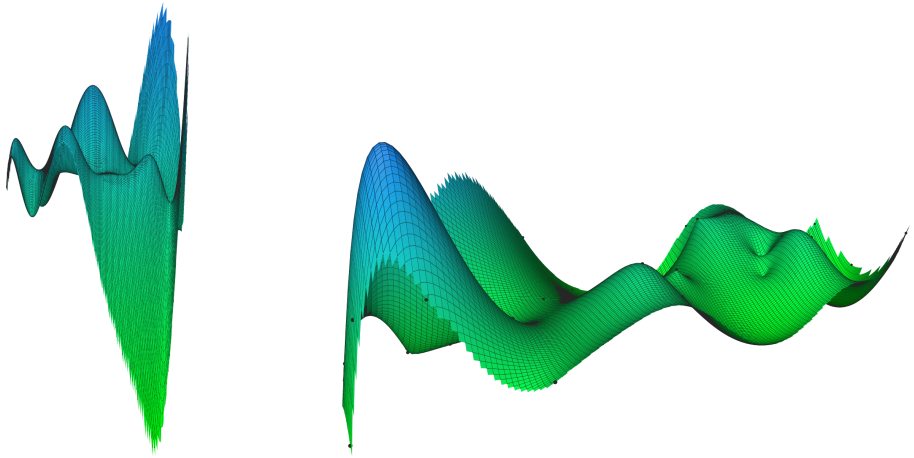
[WKB12] Ward, G., Kurt, M. and Bonneel, N. A Practical Framework for Sharing and Rendering Real-World Bidirectional Scattering Distribution Functions. Lawrence Berkeley National Laboratory, LBNL5954E, September, 2012.

[Ward14] Ward,G., Kurt, M. and Bonneel, N. Reducing Anisotropic BSDF Measurement to Common Practice. Proceedings of the 2nd Eurographics Workshop on Material Appearance Modeling: Issues and Acquisition, MAM '14, 2014. pp. 5–8, http://diglib.eg.org/EG/DL/WS/MAM/MAM2014/005–008.pdf, http://dx.doi.org/10.2312/mam.20141292, edit. Reinhard K. and Holly R., Lyon, France, Eurographics Association



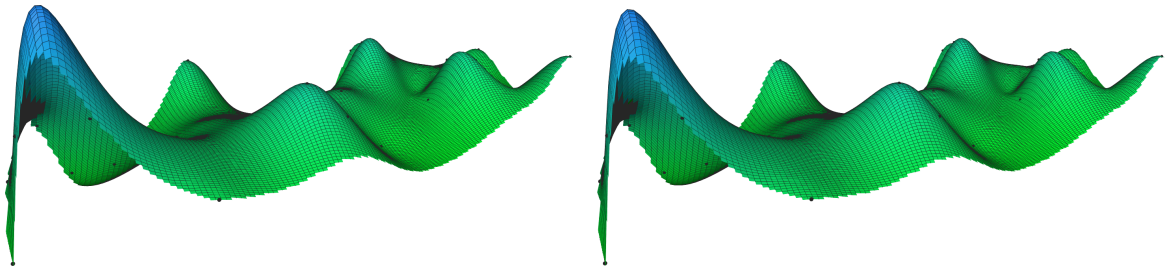
(a) LOOCV method, $c = 1.0\text{e}{-10}$ (b) LMOCV method, $c = 1.0\text{e}{-10}$

Figure 4: Dataset *April*: HMQ interpolation function with the optimal global shape parameter



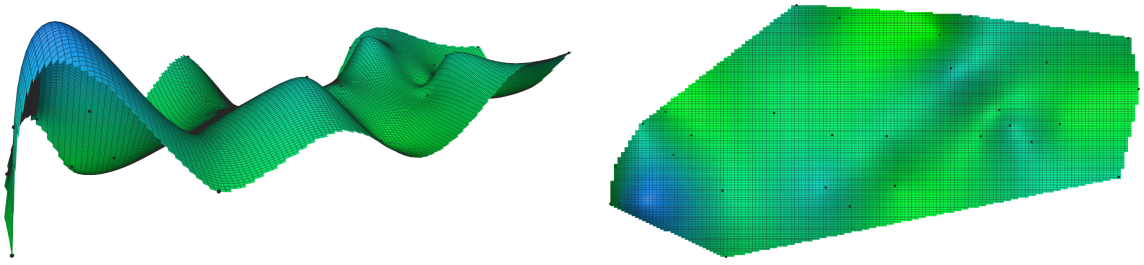
(a) LOOCV method (b) LMOCV method

Figure 5: Dataset *April*: HMQ interpolation function with the optimal local shape parameters



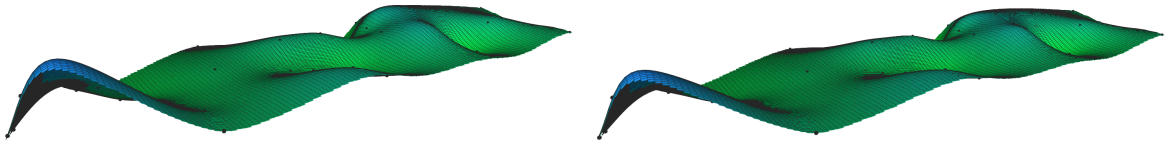
(a) LOOCV method, $c = 0.148$ (b) LMOCV method, $c = 0.143$

Figure 6: Dataset *April*: IMQ interpolation function with the optimal global shape parameter



(a) LOOCV method (b) Convex hull

Figure 7: Dataset *April*: TPS interpolation function with its top view



(a) LOOCV method, $c = 0.0854$ (b) LMOCV method, $c = 0.0621$

Figure 8: Dataset *June–July*: HMQ interpolation function with the optimal global shape parameter

Application of Vision-based Particle Filter and Visual Odometry for UAV Localization

Rokas Jurevičius
Vilnius University Institute of Mathematics and Informatics
Akademijos str. 4
LT-08663 Vilnius, Lithuania
rokas.jurevicius@mii.vu.lt

Virginijus Marcinkevičius
Vilnius University Institute of Mathematics and Informatics
Akademijos str. 4
LT-08663 Vilnius, Lithuania
virginijus.marcinkevicius@mii.vu.lt

Abstract

Conventional UAV (abbr. Unmanned Air Vehicle) auto-pilot systems uses GPS signal for navigation. While the GPS signal is lost, jammed or the UAV is navigating in GPS-denied environment conventional autopilot systems fail to navigate safely. UAV should estimate it's own position without the need of external signals. Localization, the process of pose estimation relatively to known environment, may solve the problem of navigation without GPS signal. Downward looking camera on a UAV may be used to solve pose estimation problem in combination with visual odometry and other sensor data. In this paper a vision-based particle filter application is proposed to solve GPS-denied UAV localization. The application uses visual odometry for motion estimation, correlation coefficient for apriori known map image matching with aerial imagery, KLD (abbr. Kueller-Leiblach distance) sampling for particle filtering. Research using data collected during real UAV flight is performed to investigate: UAV heading influence on correlation coefficient values when matching aerial imagery with the map and measure localization accuracy compared to conventional GPS system and state-of-the-art odometry.

Keywords

Particle Filter Localization, GPS-Denied Navigation, Visual Odometry, KLD Sampling, Correlation Coefficient

1 INTRODUCTION

GPS signal used for UAV navigation is vulnerable to signal jamming and spoofing [8]. Encoded military standard GPS signals are safe against spoofing, although they are still vulnerable to jamming and are not publicly available. Conventional autopilot systems fail to navigate safely since there is no available alternatives to GPS positioning. UAV should estimate it's own position without the need of external signals. Localization, the process of pose estimation relatively to known environment, may solve the problem of navigation without GPS signal. Particle filters have solved localization problem for autonomous robots [11] using laser scanners and panoramic vision [1]. Mixed particle filter algorithm may address robot localization issue with better precision over long distance and long duration flights. Algorithm enables navigation in GPS-denied environment and contributes to UAV safety as a GPS backup system.

Downward looking camera on a UAV may be used to solve pose estimation problem [5, 9] in combination with visual odometry and other sensor data. Such solution may deal with this problem to certain limitations: the visible area of the camera must contain enough visual features for tracking throughout the flight. Errors are accumulating and errors add up to infinity, within infinite flight time. Similar problem of mobile robot localization was solved using Monte Carlo localization [13]. Particle filter mixed with wheel odometry approach was used in [4], where a mobile robot used ceiling mosaic and a upward facing camera to localize it‘s position using particle filters. Research by H. Andreasson [1] demonstrates successful application of particle filter with panoramic vision for robot localization. Stereo vision systems have been successfully applied to low/medium size UAVs due to it‘s low weight and versatility. The problem of two cameras is the rigid distance between them, which limits the useful altitude range [3]. Computer vision techniques were demonstrated to be able to solve "kidnapped robot problem" (or global localization problem) using visual odometry and Extended Kalman-filter based SLAM (abbr. Synchronous Localization and Mapping) in [3]. This solution recovers flight trajectory from homography calculated from image feature matching. This method is vulnerable to bad results if there is not enough landmarks in the image. In this paper a particle filter algorithm is proposed that integrates data from visual odometry and uses KLD sampling technique for UAV localization in a previously known map. The results are compared with conventional GPS system and position calculated from visual odometry only.

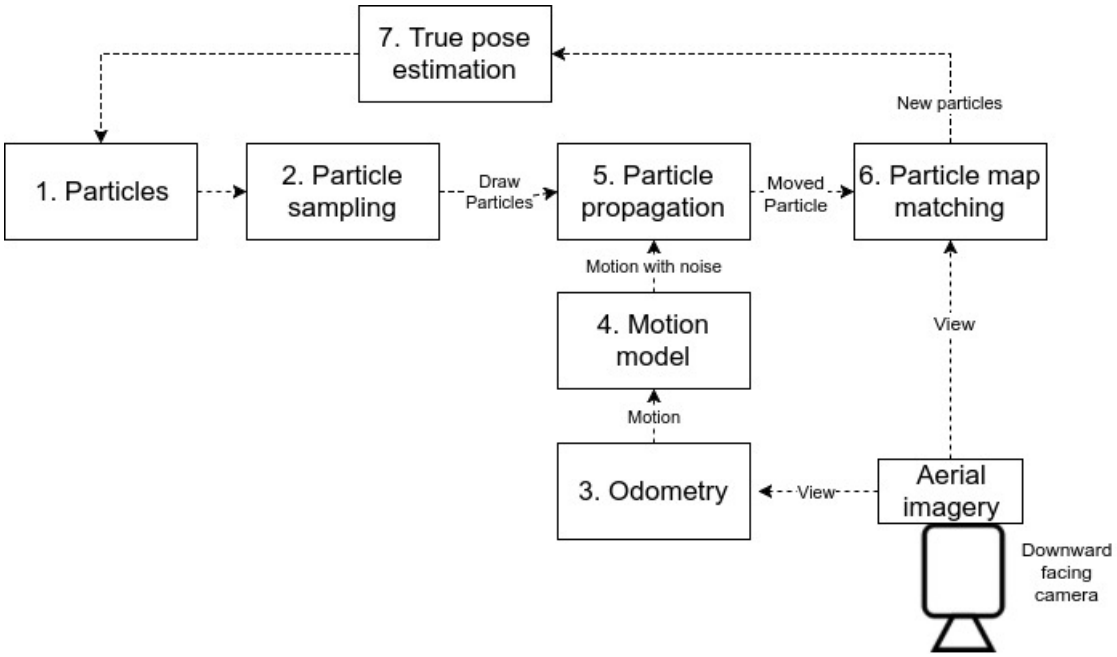


Figure 1: Proposed particle filtering algorithm including visual odometry

2 VISION-BASED PARTICLE FILTER LOCALIZATION

This paragraph shows the theory behind particle filter localization and algorithms used in particular steps particle filtering. Fig. 1 shows the schema of the proposed algorithm. Each step is described in subsections in details.

2.1 Particles

Particle is a hypothesis for the aircraft's possible position in map. A number of particles is maintained in the algorithm to evaluate more than one possible location of the aircraft and propagate the possibilities over time. Each particle is assigned an image similarity value on time t

$$b_t = b_{t-1}\frac{R_t + 1}{2}$$

that is calculated using UAV image similarity value R_t with the map image on the particle location. Initial particle density value is assigned $b_0 = 1$. Particles are also assigned weight value which is used during sampling. The particle weight w_i is calculated by normalizing all probabilities $w_{i,t} = \frac{b_{i,t}}{\sum_{j=0}^{n} b_{t,j}}$, where n is the number of particles, i is single particle index, t is time of current iteration.

2.2 Particle sampling

Sampling is the stage of the Particle Filter when particles are re-sampled according to their weight. Each iteration re-samples particles to find the most plausible UAV location over time. KLD-sampling technique was selected due it's to ability to dynamically adjust particle count thus reducing computational costs when it is not necessary. Sampling uses Kueller-Leiblach distance [6] to calculate minimal number of particles that keeps particle probability distribution the same. The technique has shown good results against other sampling techniques on simulated flight data - it provides the same localization accuracy, but dynamic particle count allows to decrease computational times up to 1.7 times [7].

2.3 Odometry

Visual odometry is the process of calculating aircraft (or robot) motion from camera images. In this setup monocular SVO [5] (abbr. Semi-direct Visual Odometry) with downward facing camera is used to calculate motion. SVO algorithm was selected due to high accuracy compared with other algorithms and real-time execution on embedded is possible due to semi-dense algorithm implementation.

SVO algorithm was selected because of more accurate positioning compared to other algorithms and real-time execution on embedded platforms. SVO algorithm was shown to run 55 frames per second on an embedded flight computer.

2.4 Motion model

Motion model is used for dead-reckoning of the UAV pose from odometry data (visual and movement speed sensors). The UAV pose may be described using six parameters
$\langle x, y, z, \theta_{roll}, \theta_{pitch}, \theta_{yaw} \rangle$ in space relative to the known environment. Parameters x, y and z are the coordinate

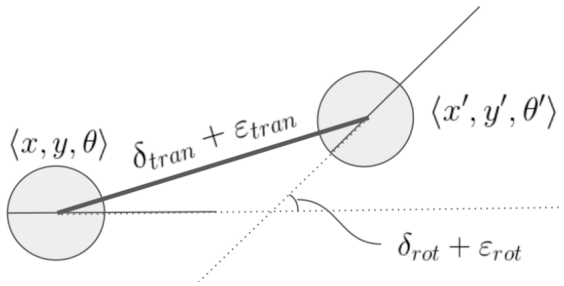


Figure 2: Simplified planar motion model for UAV

locations in 3D environment. Parameter z is equivalent to altitude, which can be measured using sensors (barometer, laser) with relatively high precision.
In the case of localization in orthophoto map the altitude is only required for image pixel scaling, so it can be ignored during localization. Roll and pitch angles are required for the calculation of camera relative elevation angle and image center on the map. Those parameters can be ignored by using camera gimbal hardware in the case if camera is configured to always look downward. The search space thus is narrowed down to pseudo-planar movement using only three parameters (see fig. 2), where aircraft pose $P_t = \langle x, y, \theta_{yaw} \rangle$, where θ_{yaw} is UAV heading angle (UAV platform angle to North). Figure 2 shows planar movement of the particle with appriori position $\langle x, y, \theta \rangle$ and posteriori position $\langle x', y', \theta' \rangle$. Particle movement is described as translational movement $\hat{\delta}_{tran} = \delta_{tran} + \varepsilon_{tran}$, where

- $\hat{\delta}_{tran}$ is planar movement with an extra noise
- δ_{tran} is measured movement change measured by sensors (usually odometry)
- ε_{tran} is additional random noise value

Rotational movement is described as $\hat{\delta}_{rot} = \delta_{rot} + \varepsilon_{rot}$, equation explanation is analogues to translational movement.

The pose update can be calculated after new sensor data using these equations [12]:

$$x' = x + \alpha_1 \hat{\delta}_{tran} cos(\theta_{yaw} + \alpha_3 \hat{\delta}_{rot})$$

$$y' = y + \alpha_2 \hat{\delta}_{tran} sin(\theta_{yaw} + \alpha_3 \hat{\delta}_{rot})$$

$$\theta'_{yaw} = \theta_{yaw} + \alpha_3 \hat{\delta}_{rot}$$

, where:

- x', y' and θ'_{yaw} are posterior UAV location relative to the orthophoto map
- α_n - measurement noise scale coefficients, selected manually
- $\hat{\delta}_{tran}$ - translational (movement speed) measurement with measurement noise ε_{tran}, obtained:
$$\hat{\delta}_{tran} = \delta_{tran} + sample\_normal(\varepsilon_{tran})$$
- $\hat{\delta}_{rot}$ - rotational (heading angle) measurement with measurement noise ε_{rot}, obtained:
$$\hat{\delta}_{rot} = \delta_{rot} + sample\_normal(\varepsilon_{rot})$$
- *sample_normal* is Gaussian distribution sampling function:
$$sample\_normal(\varepsilon) = \varepsilon \cdot gaussian(0, \tfrac{1}{3})$$

2.5 Particle propagation

This step of the Particle Filter uses sensor data, odometry and motion model to propagate the particles after re-sampling. Propagation moves the old re-sampled particles into their current locations according to movement that happened since last Particle Filter iteration.

2.6 Particle map matching

Template matching technique is used to match image viewed by the camera and cropped image from map on the particle pose. Matching is done using monochrome gray scale images to make the matching faster. The pixel gray scale value y is calculated using formula from OpenCV library [2]:

$$y_{x,y} = 0.299 r_{x,y} + 0.587 g_{x,y} + 0.114 b_{x,y}$$

, where $r_{x,y}$, $g_{x,y}$ and $b_{x,y}$ are the respective red, green and blue pixel values on image x and y coordinates. Normalized correlation coefficient (CCOEFF) from OpenCV [2] library was used to calculate image similarity R_t on time t between camera image T and cropped map image I:

$$R_t = \frac{\sum_{x=0}^{w} \sum_{y=0}^{h} (T'(x,y) \cdot I'(x,y))}{\sqrt{\sum_{x=0}^{w} \sum_{y=0}^{h} T'(x,y)^2 \cdot \sum_{x=0}^{w} \sum_{y=0}^{h} I'(x,y)^2}}$$

, where

- $T'(x,y) = T(x,y) - \frac{\sum_{x'=0}^{w} \sum_{y'=0}^{h} T(x',y')}{w \cdot h}$
- $I'(x,y) = I(x,y) - \frac{\sum_{x'=0}^{w} \sum_{y'=0}^{h} I(x',y')}{w \cdot h}$
- w, h are the image dimensions (width and height).

2.7 UAV Pose Estimation

True location of the UAV is calculated by recursively estimating particle belief density values as described in [12]. Belief is calculated for each particle as conditional probability

$$bel(P_t) = p(P_t | P_{0:t-1}, m_{1:t-1}, b_{1:t-1})$$

, where

- P_t - predicted aircraft pose on time t
- m_t - sensor data (may be IMU, barometer, wind speed and other data used for dead-reckoning) on time t

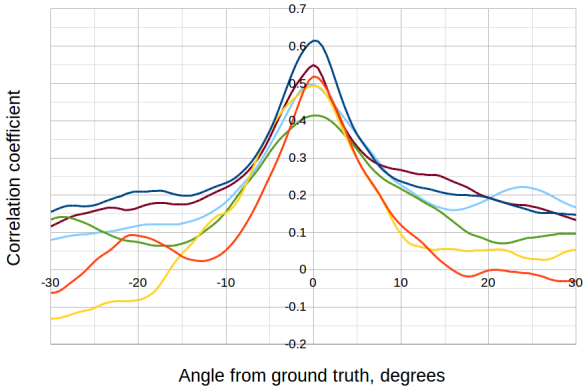


Figure 3: Correlation coefficient values when matching map with aerial imagery on all heading angles.

- b_t - previously described particle probability value on time t

The belief is a probability on location P_t, conditioned on all previous sensor data and all particle probability density values. The particle with highest belief is considered to be the true UAV pose for current iteration.

3 EXPERIMENTAL RESULTS

This section describes experimental environment, hardware components used for data collection and obtained results

3.1 Experimental setup

A fixed-wing UAV was used to collect aerial imagery and sensor data during 1 km flight. Basler acA640-120uc industrial camera with global shutter was used to collect aerial imagery alongside with other sensor data provided from UAV flight controller. Images was recorded in 640x480 resolution at 90 FPS. Data was recorded using MPEG2-TS video format and sensor data was recorded as meta-data alongside the video stream. The video playback allows data to be read with the same timing as it was recorded on UAV. Map used for matching was downloaded from Google Maps [10] using highest available zoom level. Initial particle count for the particle filter was set to 500.

3.2 UAV Heading impact on correlation coefficient

This section investigates magnetometer error impact on correlation coefficient since it was noticed during localization experiment. Typical magnetometers used in UAV's may contain noise in measuring heading direction. Fig. 3 presents correlation coefficient values for 6 images captured during real flight and matched with according orthophoto map images. Table 1 contains average similarity change values versus heading change. Data from table 1 suggests that +/- 2 degrees of heading

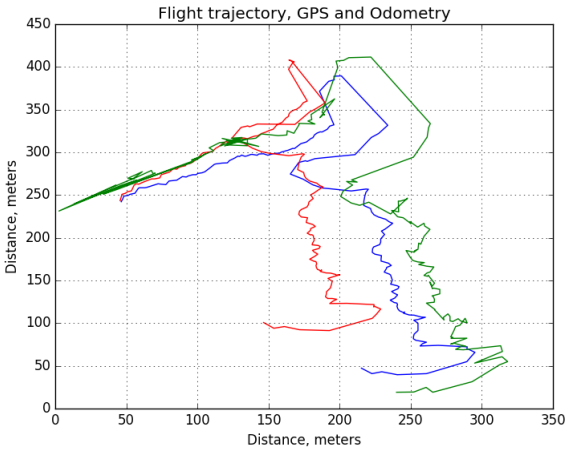


Figure 4: Flight trajectory reconstruction using odometry (red), particle filter (green) and conventional GPS sensor (blue).

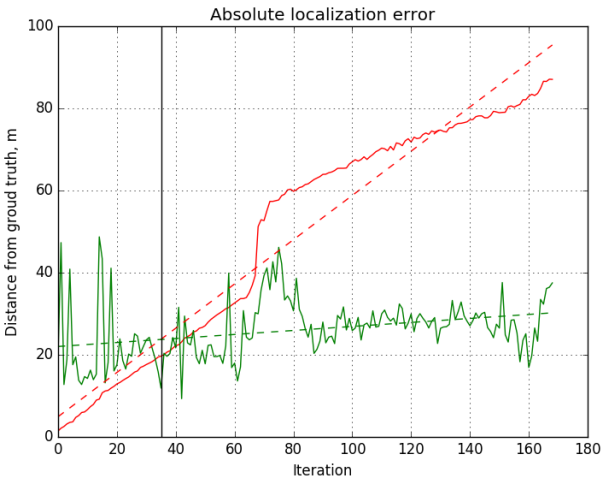


Figure 5: Particle filter localization and visual odometry absolute errors.

angle error can be ignored, because it affects correlation coefficient only up to 10% on average.

Table 1: Heading change impact on similarity coefficient

Heading change, °	Average similarity change, %
+10	67.78
+5	34.59
+2	9.38
-2	10.15
-5	35.49
-10	62.08

3.3 Localization accuracy

In this section we will evaluate localization accuracy compared with conventional GPS positioning system

and pure visual odometry positioning. Accumulated odometry error correction is expected when using particle filter in combination with odometry. Flight trajectory reconstruction using odometry and particle filter localization is presented in fig. 4. Trajectory errors in meters are presented in fig. 5, the plot shows that odometry suffers from cumulative errors as it was introduced in the introduction of the paper. The dashed lines are the trend-lines of the errors, the vertical line show the breaking point of the trend-lines at 35 seconds flight time. Since the breaking point of accuracies shows that proposed algorithm adds a lot less errors during long time flights. Additional experiments are required to validate whether errors won't add up after longer flights. Particle filter localization was able to keep error values in around 50 meter range. After the 1 kilometer flight the final error was reduced by a factor of 2 compared to localization from visual odometry only. Proposed algorithm error trend-line slope is reduced by a factor of 11 times compared with visual odometry.

4 CONCLUSIONS AND FUTURE WORK

This paper analyses an application of particle filter for UAV localization in previously known orthophoto map using images from downward facing camera on the UAV platform. The obtained results concludes:

- Flight heading error +/- 2 degrees causes correlation coefficients errors in up to 10% range. Particle filter execution was done with +/- 5 degree uncertainty in UAV heading.
- Experiment on real flight data shows that particle filter is able to reduce the slope of accumulating errors with a factor of 11 times compared to visual odometry.
- At the end of experimental flight, particle filter localization allowed to improve position precision by a factor of 2 compared with position from odometry data only.

Future work in the field of this paper would include additional experiments comparing proposed algorithm with geo-referencing based localization to benchmark localization accuracy. An alternative image similarity coefficient based on deep learning can be proposed in replacement of correlation coefficient to improve localization accuracy of the algorithm.

5 ACKNOWLEDGEMENT

The results of this research was obtained during development of project "Development of hybrid unmanned air vehicle for homeland defense purposes (No. KAM-01-08)" coordinated by Vilnius University

REFERENCES

[1] Henrik Andreasson, André Treptow, and Tom Duckett. 'Localization for mobile robots using panoramic vision, local features and particle filter'. In: *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE. 2005, pp. 3348–3353.

[2] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

[3] Fernando Caballero et al. 'Vision-based odometry and SLAM for medium and high altitude flying UAVs'. In: *Journal of Intelligent and Robotic Systems* 54.1-3 (2009), pp. 137–161.

[4] Frank Dellaert, Sebastian Thrun, and Charles E Thorpe. *Mosaicing a large number of widely dispersed, noisy, and distorted images: A bayesian approach*. Carnegie Mellon University, The Robotics Institute, 1999.

[5] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. 'SVO: Fast semi-direct monocular visual odometry'. In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 15–22.

[6] Dieter Fox. 'KLD-sampling: Adaptive particle filters'. In: *Advances in neural information processing systems*. 2001, pp. 713–720.

[7] R Jurevicius, V Marcinkevicius, and V Taujanskas. 'Comparison of image similarity functions and sampling algorithms in vision-based particle filter for UAV localization'. In: (2016).

[8] Andrew J Kerns et al. 'Unmanned aircraft capture and control via GPS spoofing'. In: *Journal of Field Robotics* 31.4 (2014), pp. 617–636.

[9] Georg Klein and David Murray. 'Parallel tracking and mapping for small AR workspaces'. In: *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*. IEEE. 2007, pp. 225–234.

[10] Gabriel Svennerberg. *Beginning Google Maps API 3*. Apress, 2010.

[11] Sebastian Thrun. 'Particle filters in robotics'. In: *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc. 2002, pp. 511–518.

[12] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. 2005.

[13] Sebastian Thrun et al. 'Robust Monte Carlo localization for mobile robots'. In: *Artificial intelligence* 128.1 (2001), pp. 99–141.

Line segment similarity criterion for vector images

Ales Jelinek
Brno University
of Technology
FEKT, UAMT
Technicka 12
61600, Brno,
Czech Republic
xjelin08@stud.feec.vutbr.cz

Ludek Zalud
Brno University
of Technology
FEKT, UAMT
Technicka 12
61600, Brno,
Czech Republic
zalud@feec.vutbr.cz

ABSTRACT

Vector representation of the images, maps, schematics and other information is widely used, and in computer processing of these data, comparison and similarity evaluation of two sets of line segments is often necessary. Various techniques are already in use, but these mostly rely on the algorithmic functions such as minimum/maximum of two or more variables, which limits their applicability for many optimization algorithms. In this paper we propose a novel area based criterion function for line segment similarity evaluation, which is easily differentiable and the derivatives are continuous in the whole domain of definition. The second important feature is the possibility of preprocessing of the input data. Once finished, it takes constant time to evaluate the criterion for different transformations of one of the input sets of line segments. This has potential to greatly speed up iterative matching algorithms. In such case, the computational complexity is reduced from $O(pt)$ to $O(p+t)$, where p is the number of line segment pairs being examined and t is the number of transformations performed.

Keywords

Vector, Line Segment, Similarity, Distance, Criterion.

1 INTRODUCTION

Evaluating similarity of two scenes is a frequent task to deal with in many technical applications. Due to inevitable error of the measuring devices and dynamic nature of the world around us, we can rarely (if ever) get two exactly same results and test them for equality. Instead, we usually get similar results in similar situations, and this is where some metrics of similarity becomes essential. A human observer usually spots the similarity unconsciously due to our evolutionarily developed sense for pattern matching, but a machine, which did not have millions of years for its development, is reliant on numerically evaluable algorithms.

Although the notion of similarity may look straightforward at first, it is a quite complex task to solve in general. Some situations require invariance to all affine transformations [1]. Other applications require invariance only to a subset of possible transformations as described in [2] and [3], where invariance only to rigid transformations and scaling is demanded. On the other hand, all transformations are important in motion estimation applications [4], [5], path planing [6] and simultaneous localization and mapping (SLAM) problem in robotics [7]. Many applications exist for 3D object detection [8] and 3D scene matching [9], where incomplete line segments often appear and the similarity criterion should take this into account. Another set of algorithms is used for polygons [10] or polyline curves [11]. Mathematically well described is the Frechet distance [12], but its domain of operation are polylines and in case of isolated line segments it simplifies to bare comparison of distances, similar to algorithms above.

Wideness of requirements being put on the similarity criterion leads to a set of algorithms with different properties for different situations, rather than a single overcomplicated method for everything. Specialization also enables deeper optimization, which is necessary in time and resource critical applications. Our research is focused on mobile robotics and SLAM, which requires 2D similarity criterion variant to all transformations with zero output for any two line segments lying on the same line. This behaviour is of a great importance, because the robot, due to an obstructed view, rarely observes an object as a whole. Partial information about the edges of the surrounding objects results into uncertainty about their real dimensions, because we do not know, which

part of the real edge has the robot actually sensed. This uncertainty is expressed as a perfect match anywhere along a line defined by an infinite extension of the static line segment. Only two or more skew line segments can solve this ambiguity and define a single transformation between the new and the static data.

Solutions mentioned above represent possible approaches, but are hard to describe mathematically because of inbuilt conditional statements and require repeated recomputation in the iterative fitting algorithms, which limits performance significantly. The method described further in this paper satisfies the requirements and solves the issues described.

2 STATE OF THE ART

The most relevant criteria to our demands are based on the Hausdorff distance. Valuable comparison of the established line segment distance functions [13] evaluates three of these similarity metrics.

Basic Hausdorff distance function for line segments, as described in [10], is defined as:

$$\begin{aligned} d_{l_1,l_2}(l_1,l_2) &= \sup_{\boldsymbol{P}\in l_1}\inf_{\boldsymbol{Q}\in l_2} d(\boldsymbol{P},\boldsymbol{Q}), \\ d_{l_2,l_1}(l_1,l_2) &= \sup_{\boldsymbol{P}\in l_2}\inf_{\boldsymbol{Q}\in l_1} d(\boldsymbol{P},\boldsymbol{Q}), \end{aligned} \tag{1}$$

$$d_{crit}(l_1,l_2) = \max(d_{l_1,l_2}, d_{l_2,l_1}), \tag{2}$$

where $d(\mathbf{P},\mathbf{Q})$ is some distance metric (Euclidean in most cases), d_{l_1,l_2} is longest perpendicular distance from l_2 to l_1 and vice versa in the second case. The criterion distance is the higher value of these two.

Modified Hausdorff line segment distance originates from [14] and the definition is as follows:

$$d_{crit}(l_1,l_2) = \min(\| l_1 \|, \| l_2 \|)\sin(\alpha), \tag{3}$$

where $\| l_x \|$ denotes length of the particular line segment and α is the angle formed by l_1 and l_2.

Modified perpendicular line segment Hausdorff distance [11] relies on the perpendicular distance between an end point of one line segment and the corresponding line segment as a whole. If a line segment is described as $l_x = \{\boldsymbol{P}_{x1}, \boldsymbol{P}_{x2}\}$ then the perpendicular distance can be written as $d_\perp(l_x, \boldsymbol{P}_{yz})$, where $x,y,z \in \{1,2\}$ are appropriate indices. The criterion is then defined by the equations:

$$\begin{aligned} d_{\perp 1} &= \min(\max(d_\perp(l_1,\boldsymbol{P}_{21}), d_\perp(l_1,\boldsymbol{P}_{22})), \\ &\qquad \max(d_\perp(l_2,\boldsymbol{P}_{11}), d_\perp(l_2,\boldsymbol{P}_{12}))), \\ d_{\perp 2} &= \min(\min(d_\perp(l_1,\boldsymbol{P}_{21}), d_\perp(l_1,\boldsymbol{P}_{22})), \\ &\qquad \min(d_\perp(l_2,\boldsymbol{P}_{11}), d_\perp(l_2,\boldsymbol{P}_{12}))), \end{aligned} \tag{4}$$

$$w_i = \frac{d_{\perp i}}{d_{\perp 1} + d_{\perp 2}} \quad \text{for } i = \{1,2\}, \tag{5}$$

$$d_{crit}(l_1,l_2) = \frac{1}{2}\left(w_1 d_{\perp 1} + w_2 d_{\perp 2}\right). \tag{6}$$

Many other criteria can be found in the literature, but although they fulfil slightly different requirements, in general, the concept is quite similar - usage of distances between points, rarely the angle of the examined line segment pair, all of them combined using min()/max() functions. Basic Hausdorff distance (2), and many others not mentioned here, does not even give zero results for line segments lying on the same line. On the other hand, modified version of this criterion (3) gives zero results for every collinear pair of line segments, which is not desirable as well.

The mentioned criteria definitely fulfil the task they were designed for, but the properties do not meet our requirements and their formulation prevents possible optimizations for better performance. The next section describes a novel criterion, which overcomes these limitations.

3 AREA BASED CRITERION FOR LINE SEGMENT SIMILARITY

The main thought behind the design of the presented criterion is following: If the similarity of two points (zero dimensional objects) in higher dimensional spaces is a distance, then the similarity of two line segments (1D objects) should be defined by an area (in more than one dimensional spaces). For the purpose of the criterion, which should return zero as an extremum for a certain input, we are going to work with the square of the area. This approach deals with a possible negative sign of the area without need to employ an absolute value and ensures, that a zero is the minimal possible output of the computation.

Let us have two arbitrary line segments $\boldsymbol{AB}$ and $\boldsymbol{CD}$, as depicted in the following Figure 1.

One of the requirements stated above demands zero output, if both examined line segments belong to the same line. This is satisfied by the squared area of the parallelogram defined by the vectors $\mathbf{x}$ and $\boldsymbol{y}$ (as depicted in Figure 1):

$$S^2_{ABCD}(l_1,l_2) = \| (\boldsymbol{B}-\boldsymbol{A}) \times (\boldsymbol{D}-\boldsymbol{C}) \|^2 = \| \boldsymbol{x} \times \boldsymbol{y} \|^2 . \tag{7}$$

To keep the notation lucid, we use a magnitude of a cross product even for a 2D problem. For every cross-product in this paper a condition of zero z coordinate for any vector involved is applied. A brief intuitive examination of the equation (7) reveals the same weakness as has the Modified Hausdorff distance criterion (3): The

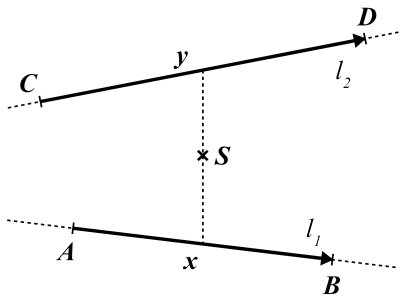


Figure 1: An example of two line segments l_1 defined by points $\boldsymbol{AB}$ and l_2 defined by points $\boldsymbol{CD}$ and the middle point $\boldsymbol{S}$. Further equations rely on a substitution: $\mathbf{x} = \boldsymbol{B} - \boldsymbol{A}$ and $\mathbf{y} = \boldsymbol{D} - \boldsymbol{C}$.

result is zero for any collinear pair $\{l_1, l_2\}$. To obtain fully functional criterion, this rule is further narrowed by introduction of the squared area of the triangle $\boldsymbol{ABS}$, which is defined by:

$$S^2_{ABS}(l_1, l_2) = \frac{1}{4} \parallel (\boldsymbol{B} - \boldsymbol{A}) \times (\boldsymbol{S} - \boldsymbol{A}) \parallel^2, \qquad (8)$$

where $\boldsymbol{S} = (\boldsymbol{A} + \boldsymbol{B} + \boldsymbol{C} + \boldsymbol{D})/4$. This function is zero for any pair $\{l_1, l_2\}$, where $((\boldsymbol{C} + \boldsymbol{D})/2) \in (\boldsymbol{A} + t\boldsymbol{x})$ and $t \in \mathbb{R}$.

By summation of the equations (7) and (8) we get the final criterion:

$$S_{crit}(l_1, l_2) = S^2_{ABCD} + 64 S^2_{ABS}. \qquad (9)$$

Coefficient 64 balances the influence of both parts of the criterion, because S_{ABCD} is significantly larger than S_{ABS}. The exact value of the constant is justified in the following subsection, where the criterion is examined deeper.

3.1 Properties of the criterion

So far, the criterion was developed using intuitive understanding of the geometrical properties of the cross product, but this can hardly prove the concept to be working at all conditions.

To provide a mathematical proof of the existence and shape of the minimum of the criterion function, we need to examine every possible mutual position and length of both line segments l_1 and l_2, which means eight variables in total. Situation can be simplified by understanding the geometrical properties of the criterion. Both area functions (7) and (8) are independent of the position of the origin, because all vectors resulting form the subtractions become invariant to translation. Invariance of the area functions with respect to rotation of both line segments is evident from an alternative form of equation (7):

$$S^2_{ABCD}(l_1, l_2) = \parallel \boldsymbol{x} \times \boldsymbol{y} \parallel^2 = \parallel \boldsymbol{x} \parallel^2 \parallel \boldsymbol{y} \parallel^2 \sin^2(\alpha). \quad (10)$$

Since lengths of the vectors and the angle between them are not affected by the rotation of the whole pair and formula (8) can be rewritten in the exact same way, we can claim, that the criterion provides results independent on that kind of transformation.

The value of the criterion function (9) is definitely dependent on the scale, but for the purpose of minimum search, that dependency is irrelevant. Multiplying a parabolic function by a constant does not affect the location of its minimum.

Combination of the previous findings implies a remarkable simplification of the minimum search task. Thanks to the invariance of the criterion to the translation and rotation and omission of the scale factor, we can fix one line segment at a constant position and examine only the remaining four independent variables defining position and length of the second one. Minimum of the criterion function (9) is then given by the solution of the following system of equations:

$$\begin{aligned}
&\frac{\partial S_{crit}}{\partial C_x} = 0, \\
&\frac{\partial S_{crit}}{\partial C_y} = 2(C_y - D_y) + 2(C_y + D_y) = 0, \\
&\frac{\partial S_{crit}}{\partial D_x} = 0, \\
&\frac{\partial S_{crit}}{\partial D_y} = -2(C_y - D_y) + 2(C_y + D_y) = 0, \\
&\text{all for } l_1 = \{\boldsymbol{A}, \boldsymbol{B}\} = \{\{0,0\}, \{1,0\}\}.
\end{aligned} \qquad (11)$$

The equations are not simplified, because at this stage we can easily compare magnitudes of the first derivatives of both components of the criterion function (9). The coefficient 64 was chosen to equalize these magnitudes, which are now 2 for both components of the non-zero equations.

System of equations (11) directly provides conditions for critical points of the criterion function:

$$\begin{aligned}
&C_y = D_y = 0, \\
&C_x, D_x \in \mathbb{R},
\end{aligned} \qquad (12)$$

which means any line segment lying on the x axis.

To reveal nature of the critical points, we compute the Hessian of the criterion function (9). Generally it is defined as:

$$\mathbf{H}_{i,j} = \frac{\partial f(x_1, \ldots, x_n)}{\partial x_i \partial x_j} \qquad \text{for } 1 \le i, j \le n. \qquad (13)$$

As l_1 is set constant at the beginning of this examination, S_{crit} is a function of l_2 only, therefore we can write it as $S_{crit}(C_x, C_y, D_x, D_y)$. The Hessian is then:

$$\mathbf{H} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 4 \\ 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 4 \end{bmatrix}. \tag{14}$$

H is positive semi-definite, which implies, that only minima or saddle points can exist in the critical points defined by the conditions (12). Since $S_{crit}(C_x, C_y, D_x, D_y)$ under conditions (12) is always zero, the continuous subspace of the critical points can only be the minimum of the function (9).

Now we can claim, that the criterion (9) truly satisfies requirements formulated in the semifinal paragraph of the Introduction. The only remaining feature to be described is an optimized procedure for similarity evaluation of multiple line segment pairs at once.

3.2 Expansion for a set of line segment pairs

In practice, there are a lot of situations, where two sets of line segments are being tested for similarity. An overall similarity is then given by a sum of similarities of all corresponding pairs from both sets. During scan to map matching in robotics, or image to image registration in computer vision applications, one set is often considered static (remains constant during computation) and the other is dynamic (i.e. manipulated using rigid transformation). The transformation is iteratively adjusted to minimize the overall similarity criterion. The established similarity criteria require transformation of the second set of line segments and recalculation of the output value any time, the transformation changes. Our criterion allows to precompute the result and then transform it in constant time, regardless the number of pairs being examined.

Let the line segment $l_1 = \{\boldsymbol{A}, \boldsymbol{B}\}$ be static and the $l_2 = \{\boldsymbol{C}, \boldsymbol{D}\}$ belong to the transformed set. The transformation is described by a rotation matrix **R** and translation vector $\boldsymbol{t}$:

$$\mathbf{R} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}, \qquad \boldsymbol{t} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}, \tag{15}$$

where θ is the angle of rotation and t_x and t_y are translations in the direction of the x and y axes and affects an arbitrary point in accordance with equation $\boldsymbol{P'} = \mathbf{R}\boldsymbol{P} + \boldsymbol{t}$. Equations (7) and (8), with transformation of l_2 included, look as follows:

$$S^2_{ABCD}(l_1, l_2) = \| (\boldsymbol{B} - \boldsymbol{A}) \times (\mathbf{R}(\boldsymbol{D} - \boldsymbol{C})) \|^2, \tag{16}$$

$$S^2_{ABS}(l_1, l_2) = \frac{1}{64} \| (\boldsymbol{B} - \boldsymbol{A}) \times (\boldsymbol{A} + \boldsymbol{B} + \mathbf{R}(\boldsymbol{C} + \boldsymbol{D}) + 2\boldsymbol{t} - 4\boldsymbol{A}) \|^2 . \tag{17}$$

The overall similarity for a set of N line segment pairs is then given by:

$$\begin{aligned} S_{tot} &= \sum_{i=1}^{N} S_{crit,i} \\ &= s^2 \sum (\boldsymbol{x}_i \cdot \boldsymbol{y}_i)^2 \\ &+ c^2 \sum \| \boldsymbol{x}_i \times \boldsymbol{y}_i \|^2 \\ &+ 2cs \sum (\boldsymbol{x}_i \cdot \boldsymbol{y}_i) \| \boldsymbol{x}_i \times \boldsymbol{y}_i \| \\ &+ \sum \| \boldsymbol{x}_i \times \boldsymbol{v}_i \|^2 \\ &+ 2s \sum (\boldsymbol{x}_i \cdot \boldsymbol{z}_i) \| \boldsymbol{x}_i \times \boldsymbol{v}_i \| \\ &+ 2c \sum \| \boldsymbol{x}_i \times \boldsymbol{z}_i \| \| \boldsymbol{x}_i \times \boldsymbol{v}_i \| \\ &+ 4t_y \sum x_{x,i} \| \boldsymbol{x}_i \times \boldsymbol{v}_i \| \\ &- 4t_x \sum x_{y,i} \| \boldsymbol{x}_i \times \boldsymbol{v}_i \| \\ &+ s^2 \sum (\boldsymbol{x}_i \cdot \boldsymbol{z}_i)^2 \\ &+ 2cs \sum (\boldsymbol{x}_i \cdot \boldsymbol{z}_i) \| \boldsymbol{x}_i \times \boldsymbol{z}_i \| \\ &+ c^2 \sum \| \boldsymbol{x}_i \times \boldsymbol{z}_i \|^2 \\ &+ 4st_y \sum x_{x,i} (\boldsymbol{x}_i \cdot \boldsymbol{z}_i) \\ &- 4st_x \sum x_{y,i} (\boldsymbol{x}_i \cdot \boldsymbol{z}_i) \\ &+ 4ct_y \sum x_{x,i} \| \boldsymbol{x}_i \times \boldsymbol{z}_i \| \\ &- 4ct_x \sum x_{y,i} \| \boldsymbol{x}_i \times \boldsymbol{z}_i \| \\ &+ 4t_y^2 \sum x_{x,i}^2 \\ &- 8t_x t_y \sum x_{x,i} x_{y,i} \\ &+ 4t_x^2 \sum x_{y,i}^2, \end{aligned} \tag{18}$$

where $\boldsymbol{x} = \boldsymbol{B} - \boldsymbol{A}$, $\boldsymbol{y} = \boldsymbol{D} - \boldsymbol{C}$, $\boldsymbol{z} = \boldsymbol{D} + \boldsymbol{C}$, $\boldsymbol{v} = \boldsymbol{B} - 3\boldsymbol{A}$ and $c = \cos(\theta)$, $s = \sin(\theta)$, both from the rotation matrix **R**. Separation of the variables related to the rigid transformation and the sums originating from the initial description of every examined line segment is a crucial result. It allows us to precompute the sums only once for the whole set and then evaluate the cumulative similarity for any transformation in constant time. Assuming p is the number of line segment pairs being examined and t is the number of transformations performed, the computational complexity of the whole task is reduced from $O(pt)$ for established criteria to $O(p+t)$ for our criterion. This could lead to significant performance improvement of iterative matching algorithms, which rely on completely static, or partially updated data set. The examples are iterative closest line algorithms by [15] and [7].

4 EXPERIMENTS AND RESULTS

Empirical verification is essential, when any new method is being released for practical applications. In this

section, we are going to present synthetic tests proving features theoretically described in Sec. 3.1 and 3.2 and show some performance tests demonstrating advantages of our method, when similarity for a set of line segment pairs under different rigid transformations is to be computed.

All experiments were implemented in the C++ programming language and compiled with Microsoft Visual Studio 2015, using the -O2 optimization setting. No other optimizations were made to keep the tests as general as possible. The machine used for running the experiments had a four-core / eight threads, 64-bit Intel Core-i7-4790K CPU, running on 4.0 GHz. Processor cache memory is large enough to hold all of the data of every test performed. Visualization of the results was done using MATLAB 2015 computing environment.

4.1 Feature verification

For feature verification of our criterion, we have decided to adopt the methodics introduced in [13]. It clearly visualizes properties of the criterion under wide variety of conditions and an interested reader may find results for other line segment distance functions in the cited paper in the given format. We believe, that unification of testing methods will help the readers to compare the methods and choose the right criterion function for their needs.

Initial arrangement of the experiment is depicted in Figure 2. There is a static line segment $l_1 = \{\boldsymbol{A},\boldsymbol{B}\} = \{\{0,0\},\{100,0\}\}$ and a dynamic l_2 transformed by various transformations according to the particular test. Figure 2 shows the positive direction of the rotation and the x axis. Rotations are always performed about the origin of the coordinates.

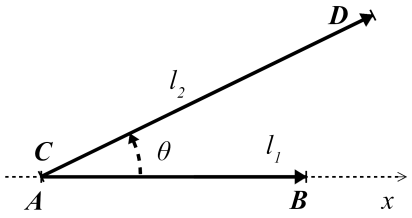


Figure 2: Schematic depiction of the positions of the line segments during the verification process as introduced in [13].

The first two tests directly follow [13]. Figure 3 depicts a situation, where l_2 is first translated in the direction of the x axis and then rotated by a given angle. The graph clearly shows, that for l_2 lying on the x axis, the criterion returns zero and as the translation increases and the angle closes to $\frac{\pi}{2}$ and $\frac{3}{2}\pi$ the output value grows rapidly.

Figure 4 shows another important case, where the length of l_2 is varied and then the line segment is

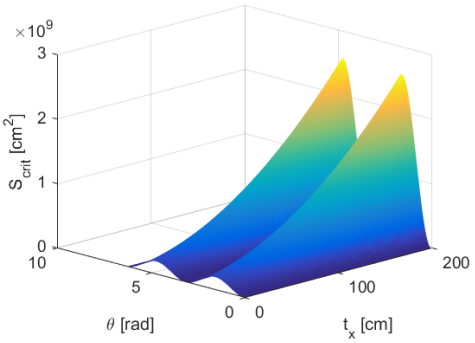


Figure 3: Criterion verification: Translation of l_2 by $t_x \in [0;200]$ centimetres followed by a rotation by $\theta \in [0;2\pi]$ radians.

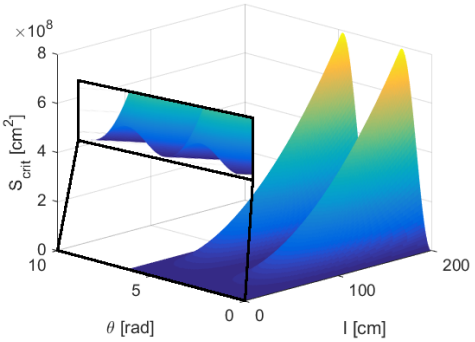


Figure 4: Criterion verification: Length scaling of l_2 in the interval $[1;200]$ centimetres followed by a rotation by $\theta \in [0;2\pi]$ radians. Sub-plot shows the detail of the graph for very small lengths of l_2.

rotated by the angle θ. Again, the criterion gives zero output for any l_2 coincident with the x axis and increases as the length grows and θ closes to $\frac{\pi}{2}$ or $\frac{3}{2}\pi$, which is desirable. Even for very short l_2 the nature of the criterion is consistent and the detail magnified in the sub-plot in Figure 4 corresponds to the harmonic shape observed in Figure 3.

The two following tests are meant to demonstrate, that for any l_2 moving along the l_1 (i.e. x axis in these experiments), the value of the criterion remains the same. Figure 5 shows a situation, where l_2 is first rotated and then translated along the x axis. The graph clearly demonstrates, that the translation has no effect and the enumerated similarity remains the same.

Similar behaviour appears, when l_2 is scaled at first, then rotated by the fixed angle $\frac{\pi}{4}$ and translated along l_1. The output of the criterion (see Figure 6) is quadratically dependant on the length of l_2, but the translation does not affect it in any way.

All four experiments prove the theoretical findings from Section 3. If l_2 and l_1 lie on the same line, the criterion returns zero, which corresponds to conditions (12). The tests also illustrate the fact, that the value of the crite-

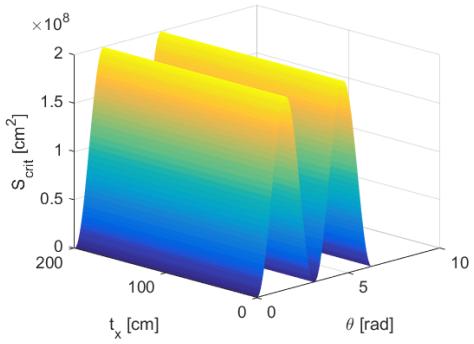


Figure 5: Criterion verification: Rotation of l_2 by $\theta \in [0;2\pi]$ radians followed by a translation by $t_x \in [0;200]$ centimetres.

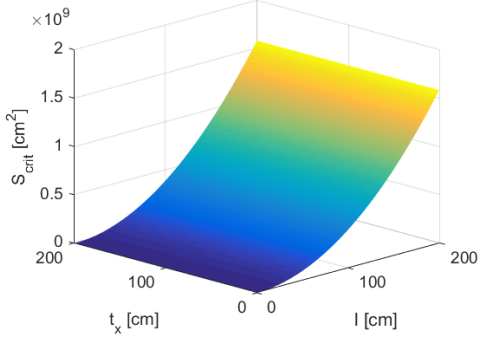


Figure 6: Criterion verification: Length scaling of l_2 in the interval $[1;200]$ centimetres followed by a rotation by a fixed angle $\pi/4$ radians and translated by $t_x \in [0;200]$ centimetres at the end.

rion is not affected by translation of l_2 in the direction of l_1, which is mathematically proved in the system of equations (11).

4.2 Performance verification

Though the equation (18) might seem enormous at first, there are many repeating terms, so the sums can be precomputed with reasonable amount of additions and multiplications. The same applies to later evaluation for various transformations. In fact, careful examination of equations (16) and (17) reveals, that the number of basic floating point operations is roughly the same in both cases.

In this set of tests the performance of a naive and the optimized algorithms is compared. The naive implementation of the criterion function for a set of line segment pairs stems from the equations (16) and (17). First, it computes the transformation of the dynamic line segments and then the errors. Contrary, the optimized algorithm first computes sums of the equation (18) and then evaluates the criterion for each particular transformation.

Figures 7 and 8 show the timings of the proposed criterion function. Both exhibit the predicted behaviour sta-

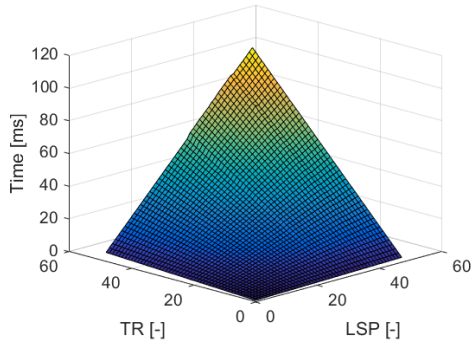


Figure 7: Processing time of a naive implementation of the criterion for various number of line segment pairs being processed (LSP) and transformations performed (TR).

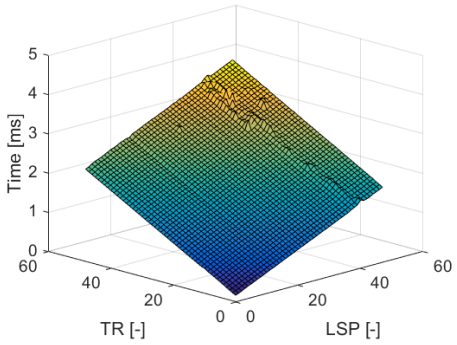


Figure 8: Processing time of the optimized implementation criterion for various number of line segment pairs being processed (LSP) and transformations performed (TR).

ted in Section 3.2. Naive implementation corresponds to $Q(pt)$ computation complexity, while the optimized algorithm is $O(p+t)$. To emphasize the benefits of the optimization Figure 9 shows the speed-up over the basic version. The performance gains are present even for the lowest numbers of transformations and line segment pairs, but the small percentage of improvement is highly implementation dependant and redeemed by a larger memory footprint of the optimized algorithm. The most significant improvements can be observed, when both p and t grow up, which is a frequent case. In such situation, the large performance gains are doubtless.

5 CONCLUSION

This paper presents a novel area-based line segment similarity criterion. Contrary to usual alternatives, the criterion function is fully differentiable and the derivatives are continuous in the whole domain of definition. The criterion is designed to give zero output for any two line segments lying on the same line, which makes it well applicable, wherever a small vector image is to

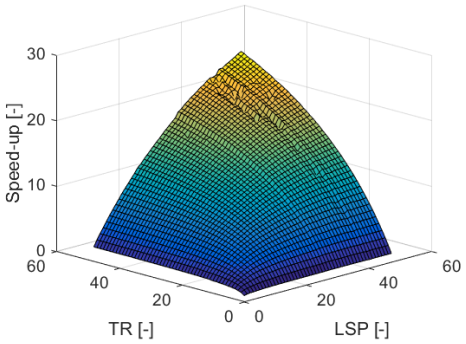


Figure 9: Seed-up of the optimized criterion over a naive implementation for various number of line segment pairs being processed (LSP) and transformations performed (TR).

be fitted into a larger one (e.g. scan to map matching algorithms in robotics).

The criterion also supports precomputation. Many algorithms iteratively transform a line segment set by a rigid transformation and compare it to a static set. Precomputation reduces computational complexity of such algorithms from $O(pt)$ to $O(p+t)$ (p is a number of line segment pairs being examined and t a number of transformation performed).

All of these features were theoretically derived and practically tested in the appropriate parts of the paper. Testing procedure was selected to correspond with other publications to provide the reader with consistent information. We hope, that this decision will help him to compare various approaches and choose the most relevant for his needs.

Future work is going to be focused on application of the presented approach in a mapping system for mobile robots and measurement of the practical performance gains. An interested user of the algorithm may benefit from multithreaded implementation, SIMD instructions, loop unrolling and many other techniques, which can improve the performance significantly, but are highly platform dependant. We hope, that features of the described criterion will open new possibilities not only in robotics, but in shape registration tasks in general.

6 ACKNOWLEDGEMENTS

The completion of this paper was made possible by the grant No. FEKT-S-17-4234 - "Industry 4.0 in automation and cybernetics" financially supported by the Internal science fund of Brno University of Technology, by the Department of Control and Instrumentation (BUT - FEEC) and by the project CEITEC (CZ.1.05/1.1.00/02.0068), financed from the European Regional Development Fund and by the TACR under the project TE01020197 - CAK3.

REFERENCES

[1] Y. Li and R. L. Stevenson, "Multimodal Image Registration With Line Segments by Selective Search," *IEEE Transactions on Cybernetics*, pp. 1–14, 2016.

[2] Z. Wang, J. M. Esturo, H. Seidel, and T. Weinkauf, "Pattern Search in Flows based on Similarity of Stream Line Segments Additional Material," in *Vision, Modeling and Visualization*, (Darmstadt, Germany), The Eurographics Association, 2014.

[3] Y. Gao and M. K. Leung, "Line segment Hausdorff distance on face matching," *Pattern Recognition*, vol. 35, pp. 361–371, 2002.

[4] Liang Huang, Qing Chang, and Shangfeng Chen, "Line segment matching of space target image sequence based on optical flow prediction," in *2015 IEEE International Conference on Progress in Informatics and Computing (PIC)*, pp. 148–152, IEEE, 2015.

[5] G. Yammine, E. Wige, F. Simmet, D. Niederkorn, and A. Kaup, "Novel Similarity-Invariant Line Descriptor and Matching Algorithm for Global Motion Estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, pp. 1323–1335, 2014.

[6] K. Z. Haigh and J. R. Shewchuk, "Geometric Similarity Metrics for Case-Based Reasoning," *Case-Based Reasoning: Working Notes from the AAAI-94 Workshop*, pp. 182–187, 1994.

[7] J. Witt and U. Weltin, "Robust stereo visual odometry using iterative closest multiple lines," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4164–4171, IEEE, 2013.

[8] W. Xiao-yu, H. Bing, S. Fang, and C. Xi, "3-D object detection based on line sets matching," in *2015 IEEE Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 270–275, IEEE, 2015.

[9] M. Poreba and F. Goulette, "Line segment based approach for accuracy assessment of MLS point cloud in urban areas," *International Symposium on Mobile Mapping Technology (MMT)*, 2013.

[10] H. Alt, B. Behrends, and J. Blömer, "Approximate matching of polygonal shapes," *Annals of Mathematics and Artificial Intelligence*, vol. 13, pp. 251–265, 1995.

[11] X. Yu, M. Leung, and Y. Gao, "Hausdorff distance for shape matching," in *The 4th LASTED International Conference on visualization, image, and image processing*, 2004.

[12] K. Shahbaz, "Applied Similarity Problems Using Frechet Distance," *arXiv:1307.6628 [cs]*, no. June, 2013.

[13] S. Wirtz and D. Paulus, "Evaluation of established line segment distance functions," *Pattern Recognition and Image Analysis*, vol. 26, pp. 354–359, 2016.

[14] J. Chen, M. K. Leung, and Y. Gao, "Noisy logo recognition using line segment Hausdorff distance," *Pattern Recognition*, vol. 36, pp. 943–955, 2003.

[15] M. Alshawa, "ICL, Iterative Closest Line: A novel point cloud registration algorithm based on linear features.," *ISPRS 2nd summer school*, no. 10, pp. 53–59, 2007.

Automatic Plant Recognition System for Challenging Natural Plant Species

Masoud Fathi Kazerouni
Institute of Real-time Learning Systems
Hölderlinstr. 3
Germany D-57076, Siegen, NRW
masoud.fathi@uni-siegen.de

Jens Schlemper
Institute of Real-time Learning Systems
Hölderlinstr. 3
Germany D-57076, Siegen, NRW
schlemper@fb12.uni-siegen.de

Klaus-Dieter Kuhnert
Institute of Real-time Learning Systems
Hölderlinstr. 3
Germany D-57076, Siegen, NRW
kuhnert@fb12.uni-siegen.de

ABSTRACT

Photosynthesis is one of turning points to shape the world. Plants use this process to convert light energy into chemical energy. Some of the early microorganisms evolved a way to use the energy from sunlight to make sugar out of simpler molecules, but unlike green plants today, the first photosynthesizing organisms did not release oxygen as waste product, so there was no oxygen in the air. Plants are very busy factories and leaves are the main place for production. A useful plant recognition system is capable of identification of different species in natural environment. In natural environment, plants and leaves grow in different regions and climates. During day, variation of light intensity can be considered as an important factor. Thus, recognition of species in different conditions is a real need as plants are ubiquitous in human life. A dataset of natural images has been utilized. The dataset contains four different plant species of Siegerland, Germany. Modern combined description algorithms, SURF, FAST-SURF, and HARRIS-SURF, have been carried out to implement a reliable system for plants species recognition and classification in natural environment. One of well known methods in machine learning community, Support Vector Machine, has been applied in the implemented systems. All steps of system's implementation are described in related sections. The highest obtained accuracy belongs to the implemented system by means of SURF algorithm and equals to 93.9575.

Keywords

Component, SURF, combination, FAST, HARRIS, FAST-SURF, HARRIS-SURF, feature extraction, feature detection, natural images, natural plant species recognition, weather condition, light intensity

1. INTRODUCTION

Ever since early man rubbed two sticks together to make fire, plants have played a vital role without any doubt in the history of mankind. In past years, some plants have been become extinct and risk of extinction always exists for plants. Collection of their information is essential to have a complete database of different plants species [1, 2, 3, 4].

Developing systems that assist botanists and professionals to recognize and identify types of plants species is really necessary in modern life. Plant species recognition relies on computational methods to extract discriminative features from images like other image recognition tasks. Tendency to have automatic systems has been increased recently. Thus a set of techniques that learn features automatically has priority. It is a goal to transform raw data to correct representation which can be effectively exploited in machine learning and pattern recognition tasks. Two main advantages of feature learning are automatic analysis of images and efficient use of features in image classification. Real-world data is usually complex, redundant, variable, and even noisy. Adaption of present strategies is very important to automate and generalize plant recognition system.

To build a useful system, different aspects or components of natural environment should be taken into consideration. More precisely, climate, weather, and wilderness can affect on performance of a plant recognition system. Therefore, weather condition is one of added factors. Different types of natural light can produce a wide variety of appearances when an image is taken from a plant, even though the light source, sun, is the same. Position and distance of leaves from camera are different, and point of view has influence on the image too. Time of day is another factor which can be considered in developing a helpful and accurate system. The system should be independent from the camera which is used to take plants images. Consideration of these continuum factors contributes to have a reliable and general system for various agricultural and medicine applications. The implemented system should tackle all challenges according to mentioned factors.

The prepared and used dataset in this work is more challenging than other present datasets as it does not contain scanned or pseudo-scanned images [5], [6]. It is one of the most realistic references of the current state of art in this area. Figure 1 shows some sample images of the dataset which belong to one plant species.

Figure 1. Four different sample images of Cornus.

Previous works have some disadvantages. Some of them [7, 8] are only applicable to certain species and they cannot be applied on other species, therefore they lack the factor of being useful for general applications. In [9], the implemented system is called a semi-automatic system, experiment is performed on 1000 images, and the obtained accuracy is 85%. Being semi-automatic is one of its weaknesses. In [10] and [11], modern combined detection and description methods have been used for classification of 32 different plant species. The used dataset in [10] and [11] is not natural, although it is a common dataset.

In this work three combined methods, SURF [12], FAST-SURF [11], and HARRIS-SURF [11], has been utilized to extract features and classify images with machine learning methods automatically. The main purpose of this work is to consider real parameters and natural factors and invent a system according to them. This paper presents some contributions. First, the proposed systems are full automatic. Second, combined methods have been used to do modern detection and description phases. Third, natural images have been utilized for the whole work. Fourth, some environmental parameters such as light intensity, and weather condition are not kept fixed.

The rest of this paper is organized as follows: section 2 presents the plant identification task; section 3 describes the details of proposed approaches; section 4 presents conducted experiments and obtained results. Finally, conclusions are drawn in section 5.

2. GENERAL REVIEW

Plant recognition task is a huge problem that had been escaped into neglect for years. In recent years, more attention has been paid to develop automatic plant recognition systems which have been demanded in different fields of scientific activities and industry. Development of such a system is important for exploration and protection of plant species to have essential information and relationship between plants.

In developing such a useful system, different parameters should be considered in all designing and implementation steps. Some of parameters are compatibility of the system with different environmental and illumination conditions, weather, distance, and etc. In order to achieve the desired goals and have a system which has generality, the first step is determination of an appropriate dataset. The dataset must include natural images to be completely close to real world. Natural images are photographs of the surrounding environment where people live. Rich covariance structure is one of natural images' properties. Characteristics of information related to redundancy are very important in classification tasks. In natural images, background of each image is different from the other one. Although it makes the recognition procedure harder, it helps to have a general dataset. According to mentioned facts, selection of an efficient method to extract features is vital and affects other classification steps. To have sufficient information, modern and engineered methods can be applied to extract effective features.

Input raw images to algorithms and systems are too large to be processed. In feature detection step, significant locations of image are produced as output. For instance, corner detectors find locations of corners in one image. To get useful information of detected parts, encoding interesting information into a series of numbers will be done by feature description methods. In computer vision field, there are various feature detection methods such as HARRIS [13] and FAST [14] which can be applied to relax the detection step and complexity of initial images. In [8] and [15], SIFT, FAST, and HARRIS methods have been utilized to construct combined feature detection and description methods. In [10], one common component of proposed combined methods is SIFT algorithm, and the system can recognize 32 different plant species and the highest obtained accuracy is 89.3519%. SURF method also can be used as a high performance scale and rotation-invariant interest point detector and descriptor [11]. This method will be used in this work to implement an automatic plant recognition system.

In order to compensate disadvantages of current systems [10, 15], another modern algorithm, SURF, is applied to implement the desired automatic system. Its speed is higher than SIFT algorithm and has good performance. Application concerns are very important for choosing an algorithm. Due to execution time of the system by SIFT in [15] for

plant recognition, it is decided to change the system by using SURF algorithm as the core of methods. Multi resolution pyramid technique is utilized in SURF algorithm to make a copy of the original image with Pyramidal Gaussian or Laplacian Pyramid shape to obtain an image with the same size but with reduced bandwidth. Thus a special blurring effect on the original image, called Scale-Space, is achieved [16]. This technique ensures that the points of interest are scale invariant [16]. SURF algorithm shows good behavior in view of slight perspective changes. In [11], SURF method has been used to distinguish 32 different plant species, where the used dataset is artificial and the images have been recorded against a white background, and the obtained accuracy is 92.28395%. This accuracy is higher than other systems which have been proposed in [10] and [17]. In order to classify plant species, Probabilistic Neural Network (PNN) has been utilized in [17] and the obtained accuracy of this semi-automated system is 91.41%. Being semi-automatic is one its disadvantages and, it is tested on only artificial images.

By using Bag of Visual Words, an image can be represented in one feature vector. However this technique is originally often used in natural language processing, it can be applied to images and a word can be considered as amount of local features. When images are expressed by vectors, it is possible to use support vector machines (SVMs) [18] for classification step.

The root of SVMs is Statistical Learning Theory and they have been mostly used in classification problems [19] or regression challenges [20]. They are robust, accurate, and effective even when a small training dataset is used. Also, they have been adopted to handle multiple classification tasks. In [21], SVM has been applied to classify leaves efficiently where the input vector of SVM is 5 variables which have been composed of 12 leaf features. In [15], SVMs have been applied in the proposed systems to do classification of natural images of plant species.

The final step is the test part. This part is performed over 336 images of the dataset. All experiments have been performed and investigated on these natural images for 4 different plant species.

3. Plant Recognition System Investigation

Automatic plant recognition systems are required to operate at a high level of reliability and accuracy in natural environment and real world. The system explanation is simplified by dividing the system into 4 different steps. The first step is the preprocessing part, which deals with conversion of RGB image. The feature detection and description parts are considered as the second step. The third step is the training part of the system in detail. The forth step consists of the test procedure of implemented systems.

3.1.1 Preprocessing

Inputs are natural images in RGB model in the prepared dataset. There are different conversion methods to convert images to grayscale format. This step is performed as the preprocessing part of the proposed system. As dataset includes natural images, each image has different numbers of objects, background, and etc. The following equation, Equation 1, has been used for conversion from RGB format to grayscale format [22].

$$Y = 0.299\,R + 0.587\,G + 0.114\,B \quad \text{(Equation 1)}$$

where R, G, and B correspond to color of each pixel.

3.1.2 Detection and Extraction of Features

The next step is detection and extraction of features. Undoubtedly human eye can extract all information of raw image, but all information is not useful for computer algorithms. In real world, changes of illumination, viewpoints, and scales are irrefutable, therefore local representations have priority to be used in proposed system. The important point is that global representations are not invariant to transformations and they are sensitive to changes. On other words, local methods represent images based on some salient regions, which remain invariant to viewpoints and illumination changes.

Due to superior performance of local features [23], these types of methods have been selected for the current systems. Also local structures are very helpful for object recognition and classification applications. Meanwhile they lead to achieve high accuracy. Due to large numbers of local features, the amount of memory increases in comparison to other methods, and it is one disadvantage of local representation. The solution is to aggregate local image descriptors into compact vector representation [24].

A reliable method in presence of changes and deformations is essential, and it can affect the whole procedure. Three various categories of feature detectors which are single scale detectors, multi scale detectors, and affine invariant detectors can be considered. In 1977, Hans P. Moravec defined the concept of interest points as distinct regions [25]. He was interested in finding distinct regions that could be used to register consecutive image frames [25]. This method is not invariant to rotation, and also it has a low repeatability rate. After 11 years, Harris and Stephens proposed a new method, Harris detector, to improve limitations of the Moravec's algorithm. Harris detector is a good combined corner and edge detector which has roughly acceptable detection results and repetition rate. It is based on intensity variation over all directions. Corners are the regions

in the image with large variation in intensity in all directions. Auto-correlation matrix is a very popular mathematical technique which is utilized in features detection methods. One 2×2 symmetric auto-correlation matrix is used in Harris method as follows:

$$M(x,y) = \sum_{u,v} w(u,v) * \begin{bmatrix} I_x^2(x,y) & I_x I_y(x,y) \\ I_x I_y(x,y) & I_y^2(x,y) \end{bmatrix}$$

(Equation 2)

where I_x is local image derivative in the x direction, nd I_y is local image derivative in the y direction, and $w(u, v)$ indicates a weighting window, rectangular or circular window, over the area (u,v). Moreover, window function gives weights to pixels.

To apply a circular window, a Gaussian one should be used. In this case, the response is isotropic, and the values will be weighted more heavily near the center. Finding interest points will be done by computation of eigenvalues of the mentioned matrix for each pixel. When both eigenvaules are large, it means that it is the location of a corner. In order to measure corner $C(x,y)$ for each pixel (x,y), the following equation will be used:

$$C(x, y) = det(M) - K(trace(M))^2 \quad \text{(Equation 3)}$$

where

$$det(M) = \lambda_1 * \lambda_2, \text{ and } trace(M) = \lambda_1 + \lambda_2 \quad (6)$$

(Equation 4)

K is used as an adjustment parameter and the eigenvalues of the auto-correlation matrix are λ_1, λ_2. Harris proposed to combine the eigenvalues in a single measure instead of two measures. Furthermore, the obtained eigenvalues make decision on status of a region. When λ_1, and λ_2 are small, $|C|$ is small too, then the region is flat, and the windowed region has constant intensity approximately. For example, the region is flat when there is little change in C in any direction. When one eigenvalue is high and the other one is low ($\lambda_2 >> \lambda_1$ or vice versa), C will be less than zero and the region is edge. In other words, the local auto correlation function is ridge shaped and little change in C has been caused by local shifts in one direction along the ridge, in addition to, significant change happens in the orthogonal direction. The last condition happens when λ_1 and λ_2 are large, so the local auto correlation function is peaked sharply and shifts in any direction will cause an increase. In this condition, C is large and the region is one corner. Harris has been used as one of detection algorithms.

Another algorithm has been applied to detect keypoints as it is an important to identify correspondence of keypoints in images. A FAST algorithm is very attractive for this purpose to get keypoints in a high speed level. It is an efficient corner detector based on comparing pixels intensities. In this algorithm, a circle of 16 pixels surrounding the central pixel has been considered to identify corners. In classification part, at least 12 continuous pixels must be darker or lighter than the central pixel, and then the central pixel will be corner. When central the pixel is classified as a corner pixel, it is not necessary to test all 16 pixels in cases of a non-corner pixel. Thus the algorithm is quick and applicable when high speed is needed, but it detects a large number of corners which is a disadvantage. This disadvantage arises from the basis of the algorithm as it is based on intensity information of 16 surrounding pixels. Furthermore, natural images are complex and a large number of corners will be detected by FAST. One pixel, p, is selected in the image. To know the status of this pixel, whether it is a corner or not, the algorithm works as below.

On threshold is set due to the selected pixel's intensity value. It is 20% of this pixel, and is considered as T. A circular area, which is consisted of 16 pixels, is considered as a search region to find out the pixel's status. 12 pixels out of these 16 pixels should be higher or lower than T (value of I_p). To have a fast algorithm, intensity of pixels 1, 5, 9, 13 are compared with Ip. Three of these four pixels should satisfy the threshold condition, so the selected pixel will exist and remain. When at least three of the four pixel values - I_1 ,I_5 ,I_9, I_{13} are not above or below $I_P + T$, p is not a corner. In this case, the pixel p is rejected. All 16 pixels are checked when at least three of the considered pixels are above or below $Ip + T$. In this case, if 12 contiguous pixels have the required condition, the selected pixel is a corner. In this algorithm all pixels will be tested and corners will be found finally.

The last used algorithm is SURF, which is applied for both feature detection and extraction. The algorithm speeds up the SIFT algorithm without scarifying the quality of detected points. Thus this method is widely used in different computer vision applications due to its efficiency, distinctiveness, and robustness in invariant feature localization. Also it has been applied to extract features as a component of the used combined methods. In SURF algorithm, an intermediate image representation, called image integral [26], is used to increase the calculation speed of the algorithm. The integral image is obtained by computation of an input image. The input image is I, and integral image is IM where a point is (x,y).

$$IM(x,y) = \sum_{i=0}^{i \le x} \sum_{j=0}^{j \le y} I(x,y) \quad \text{(Equation 5)}$$

When the integral image is used, calculating the area of an upright rectangular will lead to reduction of four operations. In addition to, change of size does not affect computation time and the algorithm is useful, even though large areas are required.

SURF entails computation of Hessian matrix and its detector is basically based on the determinant of this matrix. A 2-dimensional Hessian matrix consists of a *2*2* matrix containing the second order partials derivatives of a scalar valued function (image pixel intensities) as shown in below:

$$H(f(x,y)) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \quad \text{(Equation 6)}$$

It is a symmetric matrix and its determinant is the product of eigenvalues.

Calculation of derivatives is performed by convolution with a suitable kernel. The determinant can be calculated in different scales. Gaussians are optimal for scale space considerations. SURF approximates LoG with box filter. A parallel procedure is also possible due to two usages, box filters and integral images. The purpose is to increase computational efficiency. Integral images help to do fast computation of box convolutions and have a fast way to compute intensities for any rectangle within the image, which is independent of rectangle size. Also, computation time is not sensitive to the size of filter. A scale space is divided into octaves. These octaves show a series of filter response maps obtained by convolving the same input with a filter of increasing size [27].

The construction of scale space begins with a *9*9* filter. It calculates blob response of the image for the smallest scale. After that, the sizes of filters increase to continue the procedure, *15*15*, *21*21*, *27*27*, and etc. Blob response is shown at location *x = (x, y, σ)* as the following:

$$det\,(H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2 \quad \text{(Equation 7)}$$

When the used filter is a *9*9* matrix, σ equals to 1.2. In general, the following equation exists:

σ = (current filtersize/base filter size)(base filter scale)* (Equation 8)

where base filter size is 9 and base filter scale is 1.2.

An important step is to localize and find major keypoints in scale space. To achieve this goal, a non-maximum suppression in a *3*3*3* neighborhood is applied. The maxima of the determinant of the Hessian matrix are then interpolated in scale and image space with the method proposed by Brown et al. [27, 28]. One considerable point is the difference in scale between the first layers of every octave which is large, thus scale space interpolation is an important issue.

The next part is feature description, which should be robust and unique for a feature. Other points are the direct impact computational complexity, robustness, and accuracy. In description phase, the bases are on Haar wavelet responses in horizontal and vertical directions, x and y. The integral images will be used to do efficient calculations at any scale. Finding orientation of interest points contributes to have rotational invariance algorithm. Gaussian weights are applied to the interest point obtain robustness against deformations and translations. SURF provides a functionality which is called Upright-SURF or U-SURF and it contributes to robustness up to $\pm 15^{\circ}$ [27]. This version of SURF improves the speed of algorithm as one of advantages.

An interest area is defined by a window size of *20s*20s*. This area is divided into *4*4* square regions as subareas, and they contribute to keep spatial information. Then, Haar wavelet responses are computed for each subarea in x and y directions. A vector is created after this procedure. In the following, the formed vector is shown:

$$v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|) \quad \text{(Equation 9)}$$

In general, there are two different dimensions for SURF feature descriptor, 64 and 128. 64 dimension version has higher speed and 128 dimension version provides better distinctiveness of features. It can be considered as another functionality of the SURF algorithm. Another improvement of this algorithm helps in matching stage and speed up it. It is sign of Laplacian and the purpose is to distinguish bright blobs on dark backgrounds and vice versa:

$$\nabla^2 L = tr(H) = L_{xx}(x,\sigma) + L_{yy}(y,\sigma) \quad \text{(Equation 10)}$$

As Laplacian is the trace of Hessian matrix, the values have been calculated for determinant of the Hessian matrix before. Also, it is possible to use the sign of Laplacian to have faster matching, and it does not have any impact on performance of description and other stages. This algorithm, SURF, is used as description component of FAST-SURF and HARRIS-SURF methods. The next step of the system is training model, which includes two main parts, BoW and using SVMs for training.

3.1.3 Training

A useful and successful approach for recognition tasks is to quantize local visual features and then apply BoW method. Then a classifier, such as SVM, is applied to do classification task. By using this method, each image will be described as collection of words. In this step, obtained local descriptor will be proceeded to build a codebook and its final output will be utilized in next step. Also, this method has been widely applied for image representation [29], [30], and [31]. In [10], [11], and [15], BoW technique has been selected to classify different plant species, because it is unaffected by position and orientation of object in image and it applies fixed length vector irrespective of number of detections. The used dataset contains natural images in this system. For instance, illumination, light intensity, time, scale, and distance are not the same in all

images and they vary. Due to the characteristics of the dataset, this method contributes to represent and use images in right way and format, hence it is the right choice.

In order to create a vocabulary of visual words, quantization and grouping of local descriptors are essential. Each group will be one specific word of the vocabulary (dictionary). Therefore, K-means clustering [32] has been applied in this part due to its efficiency in clustering applications. Also it is a usual quantization approach due to its simplicity and high convergence speed. It is possible to reduce the size of feature space and compute histograms related to visual words. A finite number of clusters will be available to form a visual vocabulary. K-means uses Euclidean distance. This method finds the positions y_i, $i= 1, 2, 3, ..., k$, of the clusters that minimize the following equation, the square of the distance from the obtained feature descriptors to the cluster.

$$\arg\min_{c}\sum_{i=1}^{k}\sum_{x\in i} d(x,\mu_i)^2 = \arg\min_{c}\sum_{i=1}^{k}\sum_{x\in i}\| x-\mu_i \|_2^2$$

(Equation. 11)

where k is the number of clusters, and μ_i is the centroid of all points in c_j.

Euclidean distance is a base of this step, and local descriptors are assigned to the nearest word. As the distance between camera and plants varies, constructed vocabulary is different for each distance.

3.1.4 SVM Classification

A classifier has been designed and built in this challenging step. It has been proven that SVMs are useful tools in real world applications based on statistical learning theory. In image recognition, supervised machine learning models have been considered as efficient methods in many cases. SVM involves in finding separating optimal hyper-plane in higher dimensions efficiently, which maximizes the margin of the training data. When number of dimensions is greater than number of samples, SVM stays efficient and it is one of its benefits in addition to efficient memory. In [10], [11], [15], and [33], SVMs have been used as a part of implemented systems and the results show that SVMs are effective and efficient. The key features of SVMs are the use of kernels, the absence of local minima, the sparseness of the solution and the capacity control obtained by optimizing the margin. Therefore, SVMs have been applied in three implemented systems.

4. RESULTS AND DISCUSSIONS

Investigation of each implemented system has been performed in this section individually. A new plant dataset which consists of 4 classes, collected at the Learning Real-time System Institute, Siegen, Germany, is employed in the experiments. These images are sampled in the campus of Hölderlin, University of Siegen, Germany. Evaluation experiments have been performed to test effectiveness of the proposed systems in different aspects. The dataset includes 1000 images, and divided into two sub-datasets, training and test datasets, and corresponding numbers of sub-datasets are 664 and 336 images. Table 1 shows the number of images for each sub-dataset and relevant distances. Images' variations of the dataset are large in the prepared dataset as it has been pointed out before.

Dataset	25 cm	50 cm	75 cm	100 cm	150 cm	200 cm
Number of images for Training	160	160	160	160	12	12
Number of images for Test	80	80	80	80	8	8

Table 1. Number of images in different distances

The hardware used for the experiments is a personal computer (PC) equipped with an Intel® Core™ i7-4790K, CPU @ 4.00 GHz, and installed memory (RAM) 16.0 GB.

The first experiment is finding accuracy of the proposed systems in each distance. In order to obtain accuracy, total number of correct predictions is divided by total number of predictions [34], then it is multiplied by 100 and percentage of accuracy is calculated. While distance between camera and plants is 25 cm, the results show that implemented methods which are SURF method and HARRIS-SURF method, have the highest accuracy between all of them. Their accuracy is equal to 95%, and FAST-SURF has 93.75 percentage of accuracy.

Applied Method	Correct Predictions	Wrong Predictions	Percentage of Accuracy
SURF	76	4	95.0000
FAST-SURF	75	5	93.7500
HARRIS-SURF	76	4	95.0000

Table 2. Accuracy of classification (Distance 25 cm- K= 1000)

The accuracy of SURF method remains constant, when the distance increases to 50 cm. In this case, decrease of accuracy is high by using HARRIS-SURF method in comparison to previous distance and it has the lowest accuracy between three implemented methods. The accuracy of the system with FAST-SURF is 91.25% and increase of distance from 25 cm to 50 cm has less effect on it.

Applied Method	Correct Predictions	Wrong Predictions	Percentage of Accuracy
SURF	76	4	95.0000
FAST-SURF	73	7	91.2500
HARRIS-SURF	69	11	86.2500

Table 3. Accuracy of classification (Distance 50 cm- K= 1000)

Increase of distance affects on accuracy of implemented systems. Implemented SURF system has the highest accuracy in 75 cm, where two other systems have the same accuracy and it is 85%.

Applied Method	Correct Predictions	Wrong Predictions	Percentage of Accuracy
SURF	72	8	90.0000
FAST-SURF	68	12	85.0000
HARRIS-SURF	68	12	85.0000

Table 4. Accuracy of classification (Distance 75 cm- K= 1000)

The last accuracy measurement is done for 100 cm, 150 cm, and 200 cm distances. In this case, the system with SURF method has highest accuracy, 95.83%, and accuracy of the other systems is equal to 93.75%. The results are shown in Table 5.

Used Method	Correct Predictions	Wrong Predictions	Percentage of Accuracy
SURF	92	4	95.8300
FAST-SURF	90	6	93.7500
HARRIS-SURF	90	6	93.7500

Table 5. Accuracy of classification (Distance 100 cm, 150 cm, 200 cm- K= 1000)

Table 6 includes the information of total number of correct and wrong predictions for three implemented systems. Results of all distances have been used to achieve the following table. Highest accuracy has been achieved by SURF method, where correct predictions are 316 out of 336 tested images.

Applied Method	Correct Predictions	Wrong Predictions	Percentage of Accuracy
SURF	316	20	93.9575
FAST-SURF	306	30	90.9375
HARRIS-SURF	303	33	90.0000

Table 6. Accuracy of classification (K= 1000)

In order to understand, investigate, and describe performance and quality of an automatic classification system, confusion matrix is often used, which is an easy understanding table. It is a popular tool for performance evaluation and predictive capability of a classification system. Additionally, nature of the classification errors can be identified and compared. Each table represents one square matrix 4-by-4 for each method and distance. The confusion matrixes are shown in Table 7, Table 8, and Table 9.

SURF Method	Hydrangea	Amelanchier Canadensis	Acer Pseudoplatanus	Cornus
Hydrangea	78	0	5	1
Amelanchier Canadensis	5	76	1	2
Acer Pseudoplatanus	1	0	82	1
Cornus	2	2	0	80

Table 7. Confusion matrix- SURF

FAST-SURF Method	Hydrangea	Amelanchier Canadensis	Acer Pseudoplatanus	Cornus
Hydrangea	74	0	8	2
Amelanchier Canadensis	4	73	2	5
Acer Pseudoplatanus	1	1	81	1
Cornus	2	4	0	78

Table 8. Confusion matrix- FAST-SURF

HARRIS-SURF Method	Hydrangea	Amelanchier Canadensis	Acer Pseudoplatanus	Cornus
Hydrangea	72	0	7	5
Amelanchier Canadensis	5	72	1	6
Acer Pseudoplatanus	1	1	81	1
Cornus	3	1	1	79

Table 9. Confusion matrix- HARRIS-SURF

Precision and recall results have been derived from confusion matrix. Obtained results of precision and recall calculations show that variations of the HARRIS-SURF method's results are less than two other methods in 25 cm distance. In Figure 2, precision and recall calculations are shown for this distance. FAST-SURF method has the lowest area under its curve in precision case. It could be expected as the lowest accuracy also belongs to this method. Comparison of results can be done by investigation of area under the curves. A high area under one curve represents both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate.

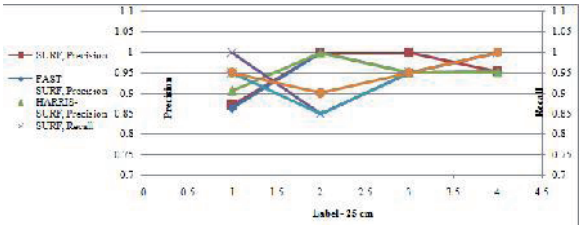


Figure 2. Precision and recall- 25 cm.

In Figure 3, precision and recall measurements are shown. SURF method has the highest areas under precision and recall curves between three used methods. The minimum values of precision and recall measurements belong to HARRIS-SURF method.

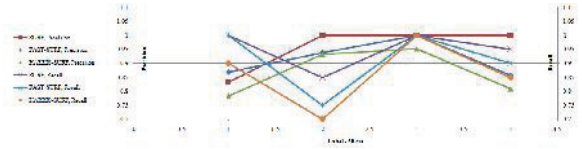


Figure 3. Precision and recall- 50 cm.

Increase of distance affects on performance of applied methods in proposed automatic classification system. As a result, measures of precision and recall calculations will be changed. In distance 75 cm, SURF method has the highest accuracy. Hence, surrounded areas under precision and recall curves are more than other two methods. In Figure 4 precision and recall results of three used methods are shown. The shape of FAST-SURF method's curve is more similar to SURF method's curve than HARRIS-SURF method's curve in precision measurements.

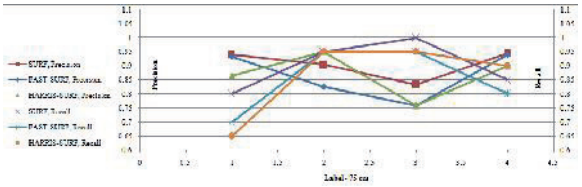


Figure 4. Precision and recall- 75 cm.

When the distance increases from 75 cm to 100 cm, 150 cm, and 200 cm, recall and precision results have been changed. The SURF method has the highest accuracy, and the recall values have been in a range, where the range is [0.91667, 1]. Precision results' range is [0.92308, 1]. These two different ranges show that the obtained values of precision and recall measurements are large and prove that system can return many correct labeled results. In this special case, obtained results of HARRIS-SIFT method are comparable to the results of FAST-SIFT method, because they have the same accuracies. The difference between accuracy of SURF method and HARRIS-SURF method and FAST-SURF method is 2% and it is small, therefore they can be applied according to desired usages and applications. Figure 5 shows the mentioned points.

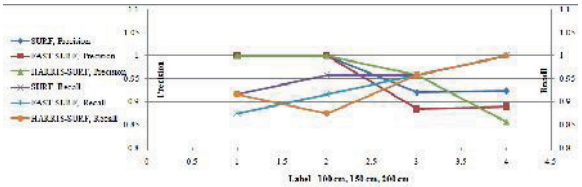


Figure 5. Precision and recall- 100 cm, 150 cm, 200 cm.

SURF method is helpful in detection and description steps as discussed in previous sections. The number of detected keypoints has been calculated which is shown for different distances in Table 10. Moreover, representation of keypoints is shown in Figure 6. Also Table 11, Table 12, and Table 13 represent needed time for implemented system.

Number of keypoints according to method and distance	SURF method
50 cm	9035
75 cm	4894
100 cm	7099

Table 10. Number of keypoints

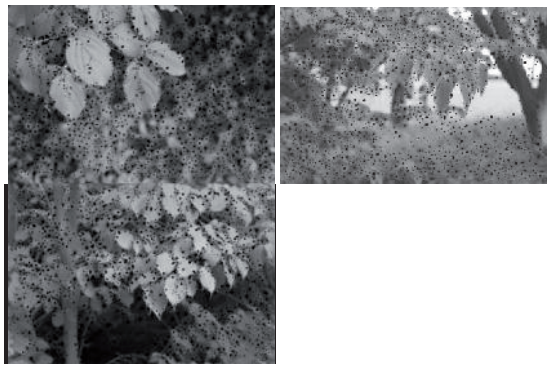


Figure 6. (Top left) Representation of keypoints for SURF method in distance 50 cm, (Top right) Representation of keypoints for SURF method in distance 75 cm, (Down left) Representation of keypoints for SURF method in distance 100 cm.

Distance	SURF Detector	SURF Descriptor	Operator SURF	Rest of Test Step
25 cm (80 Images)	301.683	582.2428	883.9258	0.009433
50 cm (80 Images)	317.5301	867.7071	1185.2372	0.008696
75 cm (80 Images)	328.998	1066.009	1395.007	0.0088608
100, 150, 200 cm (96 Images)	427.2995	1591.538	2018.8375	0.012067

Table 11. Needed time for the system used SURF

Distance	FAST Detector	SURF Descriptor	Operator FAST-SURF	Rest of Test Step
25 cm (80 Images)	0.483436	183.4367	183.920136	0.008029
50 cm (80 Images)	0.604643	269.0538	269.658443	0.008125
75 cm (80 Images)	0.677999	266.051	266.728999	0.007488
100, 150, 200 cm (96 Images)	1.059732	287.218	288.277732	0.011386

Table 12. Needed time for the system used FAST-SURF

Distance	HARRIS Detector	SURF Descriptor	Operator HARRIS-SURF	Rest of Test Step
25 cm (80 Images)	39.98336	182.3574	222.34076	0.007809
50 cm (80	40.11504	268.3261	308.44114	0.007712

Images)				
75 cm (80 Images)	40.11931	267.249	307.36831	0.006547
100, 150, 200 cm (96 Images)	48.08637	285.847	333.93337	0.009801

Table 13. Needed time for the system used HARRIS-SURF

When detection of keypoints has been done by using FAST method, the result is increase of computational speed. Although the combined FAST-SURF method has a good performance, it is robust in comparison to FAST-SIFT method when the distance is 25 cm. Its accuracy is also good enough to be applied in different systems. The needed time for the system with FAST-SURF is less than the system [15] which used SIFT as its component. The other used method is HARRIS-SURF. In this method, keypoints detection has been performed by using HARRIS method. In this case, number of detected keypoints is less than other two methods. The needed time is fair when this method has been applied. Another point is its accuracy. Its accuracy is the lowest one among all methods.

The system has the highest accuracy when it is implemented by SURF method. The SURF implemented system works well and has a good performance. As mentioned, SURF uses fast second order box filters to detect keypoints. In comparison to SIFT method in [15], SURF method detects more keypoints and the needed time has been increased. In addition to, SIFT uses smaller descriptors in comparison to SURF method which can lead to the mentioned fact. In detection step, SURF uses fast computation by means of box filters to have approximate Laplacian of Gaussian images if a fixed number of keypoints has been detected. If the number of keypoints is fixed for both SURF and SIFT algorithms, execution time is lower by using SURF, hence the needed time is comparable to the SIFT's needed time in [15]. There is also a good trade-off between time and accuracy in three implemented systems when they are compared to three other implemented systems in [15]. Additionally, proposed systems in [10] and [11] have been tested on artificial dataset while the used dataset in this work is complex and natural. Performance of the implemented systems is better than the proposed system in [9].

5. CONCLUSION

In this work, three different systems have been implemented by using combined methods of SURF, FAST, and HARRIS algorithms. In order to test the implemented systems, a very challenging dataset has been used. The dataset is composed of natural images with different point of views, angles, illumination, distances, and even weather conditions. Due to characteristics and properties of the dataset's images, the implemented systems have generality for plant recognition system. The results have been explained in details by conducting some experiments. Due to the performance of the implemented systems, they can be applied by considering the desired parameters. When the needed time should be lower, the FAST-SURF system can be applied. Obtained accuracy of the system is higher when SURF algorithm has been applied in the developed system. The next step of this work can be implementation of deep neural network to recognize plant species.

6. ACKNOWLEDGMENTS

The used dataset belongs to the Institute of Real-time Learning Systems, University of Siegen, Germany. It has been created and prepared by Masoud Fathi Kazerouni.

7. REFERENCES

[1] Du, J. X., Wang, X. F., & Zhang, G. J. (2007). Leaf shape based plant species recognition. Applied mathematics and computation, 185(2), 883-893.

[2] Ye, Y., Chen, C., Li, C. T., Fu, H., & Chi, Z. (2004, October). A computerized plant species recognition system. In Intelligent Multimedia, Video and Speech Processing, 2004. Proceedings of 2004 International Symposium on (pp. 723-726). IEEE.

[3] Zhenjiang, M., Gandelin, M. H., & Baozong, Y. (2006). An OOPR-based rose variety recognition system. Engineering Applications of Artificial Intelligence, 19(1), 79-101.

[4] Plotze, R. D. O., Falvo, M., Pádua, J. G., Bernacci, L. C., Vieira, M. L. C., Oliveira, G. C. X., & Bruno, O. M. (2005). Leaf shape analysis using the multiscale Minkowski fractal dimension, a new morphometric method: a study with Passiflora (Passifloraceae). Canadian Journal of Botany, 83(3), 287-301.

[5] Wu, S. G., Bao, F. S., Xu, E. Y., Wang, Y. X., Chang, Y. F., & Xiang, Q. L. (2007, December). A leaf recognition algorithm for plant classification using probabilistic neural network. In Signal Processing and Information Technology, 2007 IEEE International Symposium on (pp. 11-16). IEEE.

[6] Mallah, C., Cope, J., & Orwell, J. (2013). Plant leaf classification using probabilistic integration of shape, texture and margin features. Signal Processing, Pattern Recognition and Applications, 5, 1.

[7] Gouveia, F., Filipe, V., Reis, M., Couto, C., & Bulas-Cruz, J. (1997, July). Biometry: the characterisation of chestnut-tree leaves using computer vision. In Industrial Electronics, 1997. ISIE'97., Proceedings of the IEEE International Symposium on (pp. 757-760). IEEE.

[8] Saitoh, T., & Kaneko, T. (2000). Automatic recognition of wild flowers. In Pattern Recognition, 2000. Proceedings. 15th International Conference on (Vol. 2, pp. 507-510). IEEE.

[9] Mishra, P. K., Maurya, S. K., Singh, R. K., & Misra, A. K. (2012, March). A semi automatic plant identification based on digital leaf and flower images. In Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on (pp. 68-73). IEEE.

[10] Kazerouni, M. F., Schlemper, J., & Kuhnert, K. D. (2015). Comparison of modern description methods for the recognition of 32 plant species. Signal & Image Processing, 6(2), 1.

[11] Kazerouni, M. F., Schlemperz, J., & Kuhnert, K. D. (2015). Efficient Modern Description Methods by Using SURF Algorithm for Recognition of Plant Species. Advances in Image and Video Processing, 3(2), 10.

[12] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. Computer vision–ECCV 2006, 404-417.

[13] Harris, C., & Stephens, M. (1988, August). A combined corner and edge detector. In Alvey vision conference (Vol. 15, No. 50, pp. 10-5244).

[14] Rosten, E. & Drummond, T. (May 2006). Machine Learning for High-Speed Corner Detection. In European Conference on Computer Vision (vol. 1, pp. 430–443).

[15] Kazerouni, M. F., Schlemper, J., & Kuhnert, K. D. Modern Detection and Description Methods for Natural Plants Recognition (Periodical style—Accepted), ICMLA 2017: 19th International Conference on Machine Learning and Applications, Copenhagen, Denmark.

[16] https://en.wikipedia.org/wiki/Speeded_up_robust_features

[17] Hossain, J., & Amin, M. A. (2010, December). Leaf shape identification based plant biometrics. In Computer and Information Technology (ICCIT), 2010 13th International Conference on (pp. 458-463). IEEE.

[18] Vapnik, V. (1995). The natural of statistical theory.

[19] Osuna, E., Freund, R., & Girosi, F. (1997, September). An improved training algorithm for support vector machines. In Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop (pp. 276-285). IEEE.

[20] Drucker, H., Burges, C. J., Kaufman, L., Smola, A., & Vapnik, V. (1997). Support vector regression machines. Advances in neural information processing systems, 9, 155-161.

[21] Priya, C. A., Balasaravanan, T., & Thanamani, A. S. (2012, March). An efficient leaf recognition algorithm for plant classification using support vector machine. In Pattern Recognition, Informatics and Medical Engineering (PRIME), 2012 International Conference on (pp. 428-432). IEEE.

[22] https://en.wikipedia.org/wiki/Grayscale#Converting_color_to_grayscale

[23] Bianco, S., Mazzini, D., Pau, D. P., & Schettini, R. (2015). Local detectors and compact descriptors for visual search: a quantitative comparison. Digital Signal Processing, 44, 1-13.

[24] Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Perez, P., & Schmid, C. (2012). Aggregating local image descriptors into compact codes. IEEE transactions on pattern analysis and machine intelligence, 34(9), 1704-1716.

[25] Moravec, H. P. (1977). Towards automatic visual bbstacle avoidance. In International Conference on Artificial Intelligence (5th: 1977: Massachusetts Institute of Technology).

[26] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on (Vol. 1, pp. I-I). IEEE.

[27] Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008). Speeded-up robust features (SURF). Computer vision and image understanding, 110(3), 346-359.

[28] Brown, M., & Lowe, D. G. (2002, September). Invariant Features from Interest Point Groups. In BMVC (Vol. 4).

[29] Caicedo, J. C., Cruz, A., & Gonzalez, F. A. (2009, July). Histopathology image classification using bag of features and kernel functions. In Conference on Artificial Intelligence in Medicine in Europe (pp. 126-135). Springer Berlin Heidelberg.

[30] Jégou, H., Douze, M., & Schmid, C. (2010). Improving bag-of-features for large scale image search. International journal of computer vision, 87(3), 316-336.

[31] Rahman, M. M., Antani, S. K., & Thoma, G. R. (2011, June). Biomedical CBIR using “bag of keypoints” in a modified inverted index. In Computer-Based Medical Systems (CBMS), 2011 24th International Symposium on (pp. 1-6). IEEE.

[32] MacQueen, J. (1967, June). Some methods for classification and analysis of multivariate observations. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability (Vol. 1, No. 14, pp. 281-297).

[33] Purohit, S., Viroja, R., Gandhi, S., & Chaudhary, N. (2015, December). Automatic plant species recognition technique using machine learning approaches. In Computing and Network Communications (CoCoNet), 2015 International Conference on (pp. 710-719). IEEE.

[34] http://scikit-learn.org/stable/modules/model_evaluation.html#accuracy-score

Unlimited Object Instancing in real-time

Szymon Jabłoński
Institute of Computer Science
Warsaw University of Technology
ul. Nowowiejska 15/19
00-665 Warsaw, Poland
s.jablonski@ii.pw.edu.pl

Tomasz Martyn
Institute of Computer Science
Warsaw University of Technology
ul. Nowowiejska 15/19
00-665 Warsaw, Poland
martyn@ii.pw.edu.pl

ABSTRACT

In this paper, we propose a novel approach to efficient rendering of an unlimited number of 3D objects in real-time. We present a rendering pipeline that is based on a new computer graphics programming paradigm implementing a holistic approach to the virtual scene definition. Using Signed Distance Functions (SDF) for a virtual scene representation, we managed to control the content and complexity of the virtual scene with the use of mathematical equations. In order to solve the limited hardware problem, especially the limited capacity of the GPU memory, we propose a scene element repository which extends the idea of the data based amplification. The content of the repository strongly depends on a 3D object visualization method. One of the most important requirements of the developed pipeline is the possibility to render 3D objects created by artists. In order to achieve that, the object visualization method uses Sparse Voxel Octree (SVO) ray casting. The developed rendering pipeline is fully compatible with the available SVO algorithms. We show how to avoid occlusion errors which can occur in the SDF and SVO integration single-pass rendering pipeline. Finally, in order to control the content and complexity of the virtual scenes in an unlimited way, we propose a collection of global operators applicable to the virtual scene distance function. Developed Unlimited Object Instancing rendering pipeline can be easily integrated with traditional visualization methods, e.g. the triangle rasterization. The only hardware requirement for our approach is the support for compute shaders or any GPGPU API.

Keywords

Computer graphics, voxel rendering, sparse voxel octree, signed distance function, instancing, data based amplification, holistic programming paradigm, procedural graphics, level of detail

1 INTRODUCTION

Signed Distance Functions (SDF) derive from fractal theory and their application to computer graphics originated with a method of ray-tracing quaternion Julia sets [Hart89]. Among others, the paper showed that SDF relatively simple equations can represent highly detailed, complex geometry. This opened the door to modeling and rendering very complex virtual scenes. Unfortunately, the limitations of the then hardware did not allow for the full use of the new approach. Except for the Julia sets, SDF-based modeling and visualization methods were usually limited to virtual scenes consisting of only basic primitives and relatively simple isosurfaces. In this paper, we present an algorithm that significantly extends the idea of SDF based visualization in combination with the Sparse Voxel Octree 3D object representation.

One of the most common indicators used to evaluate the quality of the computer graphics is the virtual scene complexity. In order to achieve realistic rendering results, the virtual scene must be represented as a collection of high-resolution 3D objects with detailed geometries and materials.

In order to render complex scenes containing numerous high-resolution 3D objects in real-time, the rendering pipeline should be based on the three important types of algorithms. The first one refers to the virtual scene management. In order to process culling operations (e.g. the frustum inclusion test) in an efficient way, we need to organize the scene using some sort of a hierarchical spatial structure [Cao10]. The second type of an algorithm deals with the level of detail (LOD) management [Lueb02]. Using defined LOD evaluation functions, we can select which visible objects should be rendered with higher LOD and which could be rendered without some details. Finally, the third type of an algorithm is an instancing algorithm that is used to render many instances of objects stored in the GPU memory

usually along with an affine transformation and material parameters associated with each instance [Suther63].

Thanks to the constantly increasing computation power and memory capacity of today's GPUs, we can process and render more and more polygons per frame. At the same time, however, the complexity of objects in virtual scenes is also increased. It means we must improve the algorithms, which are used in computer graphics engines, so as to increase the complexity of virtual scenes.

In this paper, we present a novel approach to an efficient real-time rendering of a potentially unlimited number of 3D objects. By using Signed Distance Functions for the virtual scene representation, we extend the idea of the object instancing. Thanks to the SDF representation integrated with Sparse Voxel Octrees (SVO), we are able to render as many 3D objects created by artists as we want in real-time. The foundation of the developed algorithm is a novel computer graphics paradigm which we called *Holistic Graphics Programming*. By redefining the notion of the virtual scene, we developed a method for the virtual scene LOD management for an unlimited number of 3D object—Unlimited Object Instancing rendering pipeline (UOI).

2 RELATED WORK

There is a wide selection of literature about each mentioned component of the developed UOI rendering pipeline.

Over the years, many methods of the scene representation, LOD management and object instancing have been developed. Also, there are many papers on the Signed Distance Functions and Sparse Voxel Octrees. However, there is no related work in the context of UOI rendering pipeline algorithms that can be created by integrating this idea with the holistic approach to the scene representation, particularly in the context of visualizing 3D objects created by artists. Therefore, below we focus mainly on papers that are directly related to each component of our approach.

As mentioned in the introduction, the most common method to increase the complexity of the virtual scene is the object instancing [Suther63]. The object instancing is quite an old approach that can be placed in the context of various visualization methods. The main idea of the instancing is a compression of the virtual scene description. Instead of storing an information about the geometries of all instances of a given object in the scene separately (e.g., of each tree in a forest scene), it is possible to store only the geometry data of a single object and an additional buffer for the data that individualizes the instances once they placed in the scene (e.g., transformations that define the localization, size and orientation of an instance in the world coordinates).

Deussen et al. presented a great example of how to exploit the instancing approach to create realistic plant ecosystems in non-real-time graphics engines [Deussen98]. The geometry instancing approach is very popular for creating realistic botanical scenes due to the nature of plant structure with numerous similar elements [Snyder87, Hart91, Hart92, Kay86], and is commonly utilized in computer games. Using the instancing approach it is possible to render many instances of a given source object. However, in order to create a realistic, highly detailed virtual scene, each instance should have unique attributes. For this goal the data based amplification approach can be used to procedurally generate a variation of instances. Procedural noises and random functions can also be used to create unique, detailed variations of instances objects on the fly.

The modern GPU APIs offer a hardware-accelerated functionality for instancing geometry in real-time. Martyn showed how it can be utilized in the context of the self-affine geometry of IFS for real-time visualization of fractals [Martyn10]. Nevertheless, virtual scene rendering with the geometry instancing is limited to low-poly objects in real-time. It can still be used to render botanical scenes in an efficient way, but there is a problem to process many instances of high-resolution objects in real-time even by means of today's GPUs. The reason is that the geometry instancing with the triangle rasterization pipeline is limited by the object-space computation complexity.

Signed Distance Functions are widely used in the computer graphics from modeling and visualization of fractals [Reiner11], soft shadow generation [Wright15, Keinert14] to the font rendering [Green07]. One of the first paper that utilized this method of an object representation in the context of visualization was [Hart89]. It presented the idea of unbouding volumes which were used to ray-trace quaternion Julia sets. The idea was later extended by Hart et al. into the so-called Sphere Tracing [Hart94, Hart97].

Given an object represented by the distance function, sphere tracing relies on an iterative traversing a ray from the eye through the projection plane towards the object. For each iteration, we calculate an SDF estimated distance to the object, and if the estimation is smaller than a predefined value, the ray is considered to hit the object.

For a single primitive object like AABB, the classic ray–AABB intersection test will be much faster, because there is no need to perform many distance estimations presented in sphere tracing. However, SDF functions can be used to create highly detailed procedural objects using SDF primitives with boolean operators. Reiner et al. presented an introduction to an interactive SDF ray marching pipeline with a procedural object generation based on domain operations [Reiner11].

In the context of our work, the a most interesting operation that can be applied to distance functions is the object repetition generated by the modulo function in either controlled or unlimited manner.

Thanks to the increased computation power of today's GPUs and newly developed Sparse Voxel Octree algorithms, the high-resolution voxel-based representation is now ready for real-time applications. Due to the screen-space computation complexity of the SVO rendering pipeline, numerous high-resolution 3D objects can be processed in real-time using instancing approach. Cyril Crassin was able to perform visualization of the global illumination using SVO and voxel cone tracing [Crassin11]. There are also a few promising implementations of efficient ray tracing of SVO [Laine10] and even object animation, deformation and fracturing in real-time [Bau11, Wil13, Domaradzki16]. For that reason, the utilization of the SVO-based representation seems to be a very promising solution for modeling and visualization of 3D high-resolution objects. Moreover, SVO offers a continuous and symmetrical method of the LOD transition without any visible transition artifacts [Jab16].

3 UNLIMITED OBJECT INSTANCING

In this section, we describe requirements of the UOI rendering pipeline in real-time. Keeping in mind the hardware limitations like the limited capacity of the memory or the computation precision, we need to start with the proper definition of the UOI rendering pipeline.

3.1 Unlimited Object Instancing definition

For the purposes of this work, the requirements and features of the UOI rendering pipeline are defined as follows:

- A real-time rendering pipeline that is able to process and render an unlimited number of objects in the virtual scene. If it is possible to store a 3D object in the GPU memory and render it in real-time, it should be possible to store and render potentially unlimited instances of this object with unique variations without any noticeable performance hit or memory requirements increase.
- The possibility of visualizing 3D objects which were created by artists.
- A continuous and symmetrical LOD management of the virtual scene.

Considering the requirements above, it may seem that the most serious development obstacle is the hardware limitation. However, in our opinion, this is not the main problem. Computer hardware is and, presumably, will always be limited. In order to develop the UOI rendering pipeline, it's necessary to change the current computer graphics programming paradigm, for example, by applying the holistic approach to the definition of the virtual scene.

However, before we present the idea of the UOI rendering pipeline, we analyze some possible, naive designs that could be created using the available algorithms. Doing so, one can exclude algorithms and structures that cannot be used in the context of the proper implementation of the Unlimited Object Instancing pipeline.

3.2 Naive Unlimited Object Instancing

For the polygonal representation of geometry, the hardware instancing functionality, which is implemented in all modern GPUs, can be used. On the other hand, for a rendering pipeline that offers screen-space computation complexity like e.g. ray-casting, the software instancing approach can be used to render numerous instances of 3D objects. Both methods can be used to render many instances of a given collection of 3D objects. However, memory requirements for the virtual scene description would be increasing significantly because of the need of storing a unique data for each instance (e.g. model-to-world space transformations).

The second possibility is to store the whole scene in a single spatial structure like e.g. Sparse Voxel Octree. Using the DAG algorithm it is possible to store and render a scene object represented by a high-resolution grid (even of 128^3 resolution) [Kampe13] in real-time. The high-resolution scene grid could be also used as a virtual scene description. Then, 3D objects instancing could be used. However, none of them could be directly used to develop the proper UOI Instancing rendering pipeline.

3.3 Issues with classic paradigm

In all modern games and virtual simulations taking place in a 3D world, the virtual scene is defined as a collection of objects. In order to process and render a complex scene based on such a paradigm, it is necessary to check out which 3D objects are visible from the current point of view. Moreover, to execute the frustum visibility and occlusion tests in an efficient way, it is necessary to organize the objects implementing some sort of a hierarchical spatial structure.

This standard approach to the scene representation defines the classic real-time computer graphics programming paradigm. For the purpose of our work, we called it the *Object-Based Graphics Programming*. The main features of this programming paradigm are:

- A virtual scene is defined as a collection of 3D objects.

- The visibility and occlusion tests are executed to classify objects to render.
- Hierarchical spatial structures are used to organize the virtual scene and to increase the performance of visibility tests.

Due to hardware limitations, it is neither possible to store an unlimited collection of objects in the memory nor to perform the visibility tests for all of them. In order to develop the UOI rendering pipeline, we need to change the computer graphics programming paradigm to the paradigm which we called the *Holistic Graphics Programming*.

4 HOLISTIC UNLIMITED OBJECT INSTANCING

The foundation of the UOI rendering pipeline is the holistic approach applied to the virtual scene definition. In the holistic architecture, we replace a collection of objects with a single object—just the virtual scene object. By controlling the LOD of the virtual scene object, we will control the content and complexity of the scene regarded as a single entity, processing potentially an unlimited number of objects.

The example of the holistic approach can be found in a few popular computer graphics algorithms. A great example is the procedural terrain rendering, especially when considered in the context of its local LOD management. Using tessellation shaders, the geometrical complexity of the terrain geometry can be locally increased or decreased by controlling a tessellation factor for each patch of the terrain independently. The similar approach is used in the holistic UOI rendering pipeline to control the LOD of a scene.

The UOI rendering pipeline we developed is based on the following four components:

1. **Graphic asset repository**
2. **Virtual Scene representation**
3. **3D object visualization**
4. **Global operators collection**

4.1 Graphic asset repository

The *Graphic asset repository* contains a collection of all visual ingredients which the scene is composed of. In order to render the scene in real-time without using the data streaming functionality, all visual assets like 3D geometries and materials need to be stored in the GPU memory. The type and format of this data depend on the visualization method. In our implementation, by default, it is a collection of SVOs and textures.

Due to the limited memory capacity of GPUs, the repository contains a finite number of elements. Then, using the instancing approach with a data amplification method, the elements are rendered with unique variations, creating a complex, detailed virtual scene.

4.2 Virtual scene representation

The second component of the UOI rendering pipeline is the way we apply the holistic approach to the scene representation. In the proposed solution, we have implemented it by means of Signed Distance Function—the whole scene is represented by a signed distance function.

Since we use SDF, it is possible to create a highly detailed complex scene from primitive distance functions along with boolean operators. For the purpose of this work, we do not use complex distance functions, and the basic scene element is defined as the distance function for the cube primitive:

$$d = length(max(abs(p+o)-b),0) \qquad (1)$$

where:

d = distance to the object
p = point on ray from the eye
o = offset from scene origin
b = cube size

Using the SDF ray marching, we can generate cube primitives efficiently. After that, each cube primitive could be replaced with the 3D object represented by the SVO. On the basis of the resulting nearest ray-cube intersection, we can easily calculate a ray stop position for the SVO ray casting.

Fig. 1 presents rendering results of a virtual scene represented by a distance equation with a single SDF component.

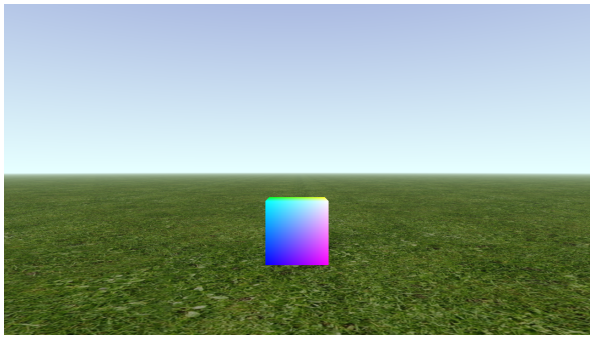


Figure 1: Ray marched SDF based virtual scene with a single SDF component. Rendered cube is shaded with calculated ray cast start position. 590 FPS with Nvidia GeForce GTX 660.

In order to add other elements to the scene, we can extend the global scene distance function by adding next SDF components to the virtual scene SDF equation.

For each component of the virtual scene equation, we have to calculate the value the component's SDF at the current iteration and process a minimum function per the ray marching iteration.

In the case of the virtual scene with three SDFs representing smaller and smaller objects in the scene, the distance equation can be expressed as follows:

$$\begin{aligned} d0 &= length(max(abs(p+o0)-b0),0) \\ d1 &= length(max(abs(p+o1)-b1),0) \\ d2 &= length(max(abs(p+o2)-b2),0) \\ d &= min(min(d0,d1),d2) \end{aligned} \quad (2)$$

where:

d	= distance to the object
$d0, d1, d2$	= distance to SDF component 0,1,2
p	= point on ray from the eye
$o0, o1, o2$	= SDF component offset from scene origin
$b0, b1, b2$	= scene SDF component 0,1,2 cube size

Fig. 2 presents rendering results of a virtual scene represented by a distance equation with three SDFs.

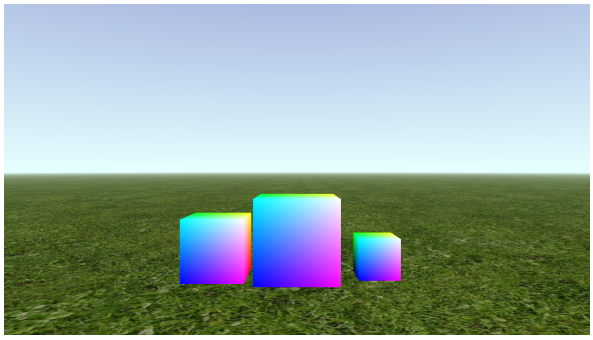


Figure 2: Ray marched SDF based virtual scene with the three SDF components. Rendered cubes are shaded with calculated ray cast start positions. 490 FPS with Nvidia GeForce GTX 660.

Obviously, the calculation of the unlimited number of minimum functions is impossible on the limited hardware. However, thanks to the simplicity and flexibility of the function-based scene representation, we are able, in theory, to handle the unlimited number of scene SDFs. It is possible because there is no need to store a large scene data in the GPU memory. That means, that the whole scene description of the highly complex virtual scene can be saved as a simple equation which makes the UOI rendering pipeline possible.

4.3 3D object visualization

Thanks to the use of distance functions as virtual scene equation components it is possible to create complex procedural objects. However, one of the main requirements of the developed rendering pipeline is the ability to visualize 3D objects created by artists. To achieve this goal, the cube primitives acquired from the SDF are replaced with the SVO-based 3D object. Moreover, we can use any available SVO algorithm for shading, object deformation and LOD management. In the implemented Unlimited Object Instancing rendering pipeline framework, we used a simple SVO ray casting.

Fig. 3 presents rendering results of the virtual scene represented by distance equation with the three SDF components with SVO based object ray casting.

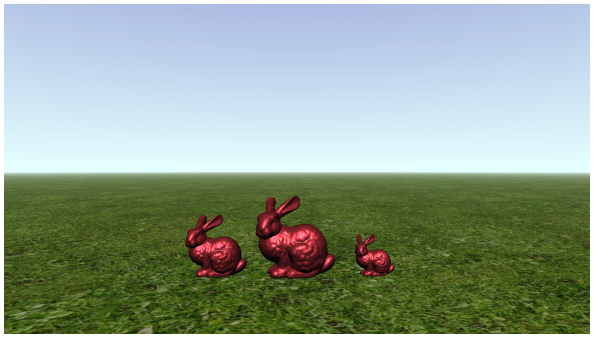


Figure 3: Ray marched SDF based virtual scene with the three SDF components integrated with SVO based 3D object ray casting. 220 FPS with Nvidia GeForce GTX 660.

The main problem related to the integration of SDF with SVO are potential occlusion errors. For SDF-based virtual scenes utilized in the "standard" way, ray-object intersection and occlusion errors would not happen.

If we want to replace SDF-based AABB boxes with SVO 3D objects, we may face the situation that a ray hits an AABB box but it misses an included SVO object. The details of our occlusion fixing solution are described in Sec. 5.

4.4 Global operator collection

The last component of our UOI rendering pipeline is the *Global operator* collection which is used to control the content and complexity of the scene in the spirit of the holistic paradigm. In order to create a complex scene by using a finite collection of the scene elements, global functions are applied to the SDF scene function. As mentioned before, one of the main features of the SDF functions is the possibility of multiplying objects by modifying primitives' distance functions.

The base operator that is used to obtain an unlimited number of instances per SDF component is the *Instancing Operator*. With the SDF-based object representation, using the modulo function it is possible to create objects in a controlled or unlimited way. The cube distance function 1 can be extended with *Instancing operator* as:

$$\begin{aligned} cell &= floor((p+size*0.5)/size) \\ p &= mod(p+size*0.5,size)-size \\ d &= length(max(abs(p)-b),0) \end{aligned} \quad (3)$$

where:

cell = object instance grid cell
size = repeat interval
d = distance to the object
p = point on the ray
b = cube size

Instancing operator can be applied in any dimension, creating a 1D/2D/3D grid with the instanced cube objects. An important feature of the repetition function is that we acquire the cell id of every generated object. This information is then used by the all remaining operators as a unique input for the noise algorithms. In addition, we take advantage of the information to fix the occlusion errors. Fig. 4- 5 present rendering results of the virtual scene with applied *Instancing operator*.

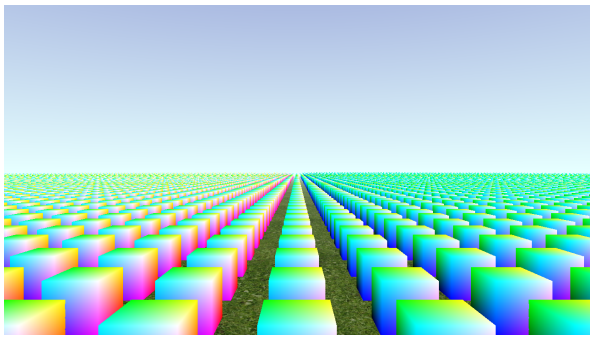


Figure 4: 2D *Instancing operator* applied for the single SDF component virtual scene. 215 FPS with Nvidia GeForce GTX 660.

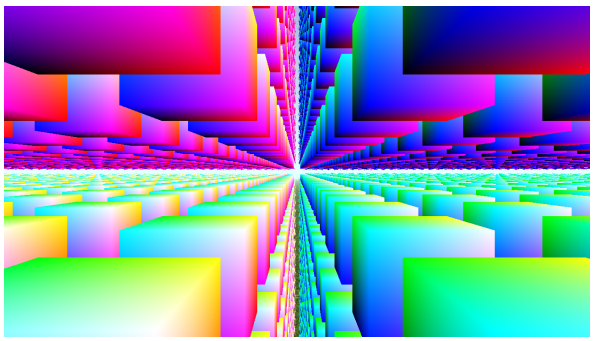


Figure 5: 3D *Instancing operator* applied for the single SDF component virtual scene. 160 FPS with Nvidia GeForce GTX 660.

As one can see in the figures 4 - 5 *Instancing operator* does not cause any noticeable performance loss regardless of the instancing dimension or a number of instances present in the scene.

Following the holistic approach, *Global operators* are applied to the whole scene. It means that we cannot control each object on the virtual scene independently.

Based on the operator's features, the developed *Global operators* have been classified into two groups.

4.4.1 Object operators

Object operators are used to control geometry and material data of the generated objects. Using the cell id acquired from the instancing operator as an input to the procedural noise algorithm (e.g. Perlin/Simplex noise), we can apply unique variations to generated objects. We have developed the following *Object operators*:

1. **Object type operator**—to choose an SVO data buffer which is used in the visualization algorithm.
2. **Material type operator**—to apply unique values for the objects material (e.g. albedo, roughness, metalness values).
3. **Existence operator**—the most advanced operator in the group. Using a noise function with the user input and features of the SDF representation, we can control the existence of generated objects. It is implemented by dynamically creating a new SDF element and performing the boolean subtraction from the virtual scene distance function. Also, if an SDF cube intersects another one it must be added to the final distance function with the boolean union operator. The same algorithm is used to fix the occlusion errors (see Sec. 5).

4.4.2 Transformation operators

The second group of the operators is used to apply an affine transformation to each generated object. We developed the following *Transformation operators*:

1. **Translation operator**—used to apply a translation transformation.
2. **Scale operator**—used to apply scale transformation.
3. **Rotation operator**—used to apply rotation transformation.

All *Transformation operators* are limited to the boundaries of the SDF component grid cell.

5 IMPLEMENTATION DETAILS

In this section, we describe important implementation details of our UOI rendering pipeline. We have implemented our method using OpenGL 4.5 API with C++14 but any other graphics interface or programming language can be used. All included shader source code listings are prepared in GLSL language. Due to the simplicity of SDF ray marching with sphere tracing, the presented approach can be easily implemented and integrated into all popular game engines. The only requirement is the support for programmable compute shaders.

5.1 Virtual scene visualization

The virtual scene visualization is directly based on classic SDF ray marching. The main extension is that in

order to solve a potential occlusion error, ray marching is executed multiple times. Therefore, we need to take the previous hit distance into account and add it to the ray marching start position in the next iteration. Also, for an SVO-based 3D object rendering it is necessary to calculate the volume ray casting start and stop positions so as to perform SVO ray-casting. Listing 1 presents a simplified, SDF with SVO rendering pipeline compute shader code.

```
struct HitResult {
  vec3 cell;
  float distance;
  int lod;
  int material;
};

HitResult RayMarch(vec3 origin,vec3 direction) {
  HitResult hit;
  float distanceDelta = Near + OcclusionDelta;
  float travel = 0.0;
  vec3 position = origin;

  for(int i = 0; i < RayIterations; ++i) {
    position = position + direction * distanceDelta;
    hit = ObjectCube(position, size, lod, material);
    HitResult hit2 = ObjectCube(position, size2,
        lod2, material2);

    if(hit2.distance < hit.distance)
      hit = hit2;

    <Occlusion resolve>
    distanceDelta = hit.distance;
    travel += distanceDelta;

    if(travel > Far) {
      hit.distance = Far;
      return hit;
    }

    if (distanceDelta - Precision < 0.001) {
      hit.distance = travel;
      return hit;
    }
  }

  hit.distance = Far;
  return hit;
}

void main(void) {
  for (int k = 0; k < OcclusionIterations; ++k) {

  HitResult hit = RayMarch(position, direction);

  if (hit.material) // material < 0 means no hit
  {
    vec3 hitPosition = position + direction *
        hit.distance;
    float size = (1.0 / hit.size.x);

    vec3 rayStart = hitPosition * size;
    vec3 rayStop = GetRayStop(hitPosition,
        direction, size);
    bool missed = true;

    <Object LOD calculation>
    <ObjectRayCast>

    if (!missed) // save result
      return;
}
```

Listing 1: Unlimited Object Instancing rendering pipeline simplified visualization source code.

5.2 Occlussion error fixing

The biggest implementation challenge for the SDF and SVO rendering pipeline implementation is potential object occlusion errors. In order to tackle this problem, we need to execute multiple ray marching iterations, one for each occlusion error.

We developed two methods for occlusion fixing. Both of them are using an occlusion stack that is used to save a previous hit information—the grid cell 3D vector and the SDF component id.

The first method is a cell estimation. If we again hit the same object, we set a current distanceDelta to the defined *Escape* value which allows for omitting the current object.

However, this method will not work if two different SDF components intersect. Also, there is need to save the history of the occurred occlusions in the stack. In this case, we need to use the second, more universal method. Using the data from the previous occlusion error, we calculate the distance for the scene cell and, using the subtraction operator, we cut it from the scene SDF function. Listing 2 presents the source code for this method.

```
vec3 occlusionCell;
int occlusionID;

float interval = SceneIntervals[occlusionID];
vec3 offset = vec3(occlusionCell.x * interval,
     occlusionCell.y * interval,
occlusionCell.z * interval); vec3 q = (position +
     SceneOffsets[occlusionID]) - offset;

float occluder = ObjectCell(q,
     SceneSizesOccluder[occlusionID]);
hit.distance = max(-occluder, hit.distance);
```

Listing 2: Occlusion error fixing methods for SDF wih SVO rendering pipeline.

6 RENDERING AND PERFORMANCE TEST RESULTS

All depicted timings were obtained on Intel Core i5 2500K CPU with NVidia GeForce GTX 660 GPU and with NVidia GeForce GTX 980. All algorithms were implemented using OpenGL 4.5 API with C++14 for Windows 10 64-bit. We used Stanford Repository models as a test object [Stanford11].

The presented rendering results show that the developed rendering pipeline is efficient and offers real-time performance even for a medium power hardware. Unfortunately, the results also show that the SDF based virtual scene representation suffers for a visible regularity of the object distribution. Also, SDF ray marching seems to be slower than for example some sort of the uniform grid traversal algorithm like 3D-DDA.

Using instancing operator we create an infinite uniform grid with defined cell size and interval. The obvious

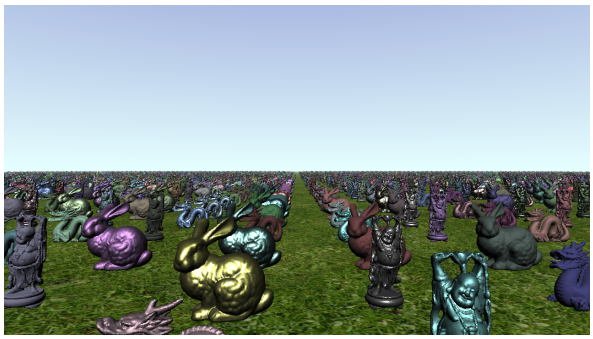


Figure 6: The virtual scene with 2D instancing operator applied. 50 FPS on Nvidia GeForce GTX 660, 183 FPS on Nvidia GeForce GTX 980.

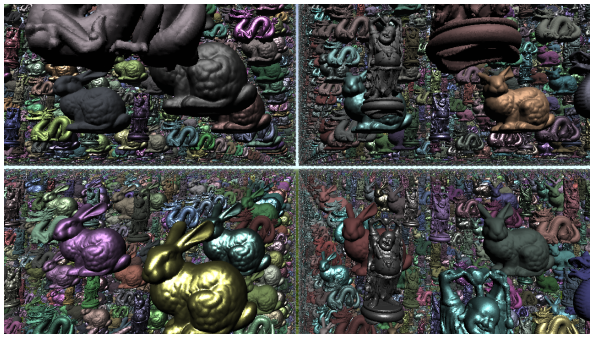


Figure 7: The virtual scene with 3D instancing operator applied. 20 FPS on Nvidia GeForce GTX 660, 67 FPS on Nvidia GeForce GTX 980.

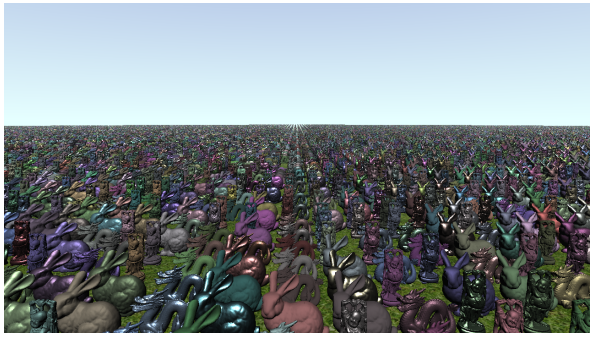


Figure 8: The virtual scene with instancing, type and material operators applied. 35 FPS on Nvidia GeForce GTX 660, 98 FPS on Nvidia GeForce GTX 980.

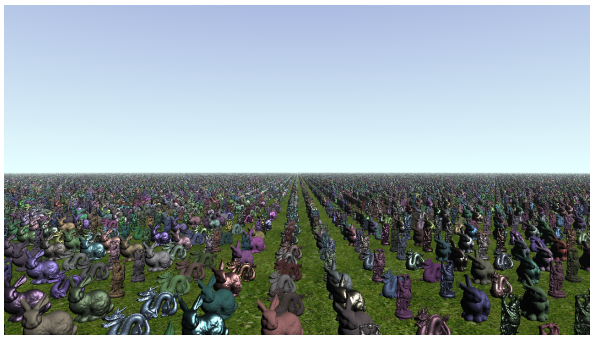


Figure 9: The virtual scene with translation and rotation operators applied. 27 FPS on Nvidia GeForce GTX 660, 81 FPS on Nvidia GeForce GTX 980.

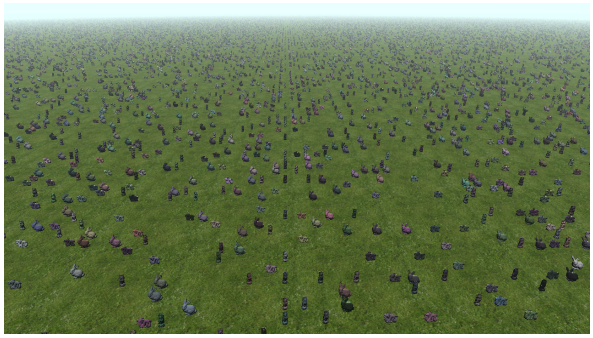


Figure 10: The virtual scene with existence operators applied. 31 FPS on Nvidia GeForce GTX 660, 80 FPS on Nvidia GeForce GTX 980.

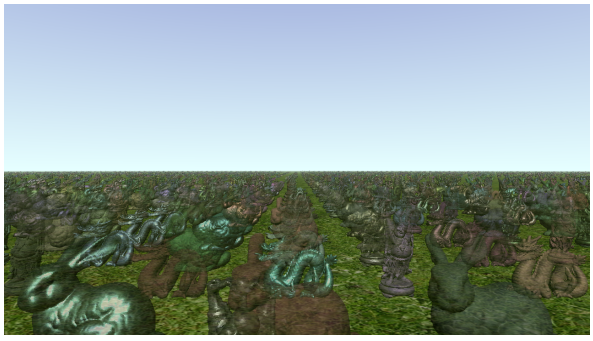


Figure 11: The virtual scene with translucent objects. 34 FPS on Nvidia GeForce GTX 660, 55 FPS on Nvidia GeForce GTX 980.

alternative seems to be a procedural definition of an infinite uniform grid along with a variation of the Bresenham algorithm for the grid traversal. Such a method would be compatible with the remaining components of the holistic approach and could be applied for, e.g., figures 8 - 11. It would presumably offer better performance results thanks to the simplicity of the uniform grid traversal. However, for the rest of the presented figures SDF it could be not applied.

First, with SDF we can efficiently use many independent SDF components to represent the virtual scene. It means that the virtual scene may contain N uniform grids with different attributes and the possible intersection between their cells. Such a virtual scene is presented in the figures 12 - 15. For virtual scenes like those, we could not use the traditional uniform grid traversal algorithm. Moreover, we are not limited only to SVO based 3D objects. Using SDF ray marching we could render highly complex 3D objects and calculate all necessary data for realistic shading like normal vectors, ambient occlusion or even soft shadows. For these reasons, SDF based virtual scene representation seems to be a better solution than the procedural uniform grid for Unlimited Object Instancing rendering pipeline.

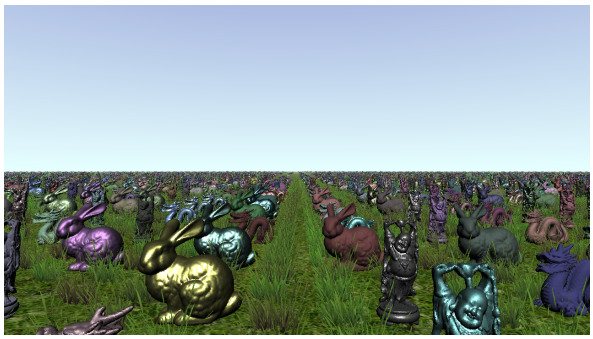


Figure 12: The virtual scene with two SDF components - one for Stanford objects and one for the grass objects. 24 FPS on Nvidia GeForce GTX 660, 100 FPS on Nvidia GeForce GTX 980.

Figure 13: The virtual scene with three SDF components - one for Stanford objects, one for the grass and one for the trees. 32 FPS on Nvidia GeForce GTX 660, 93 FPS on Nvidia GeForce GTX 980.

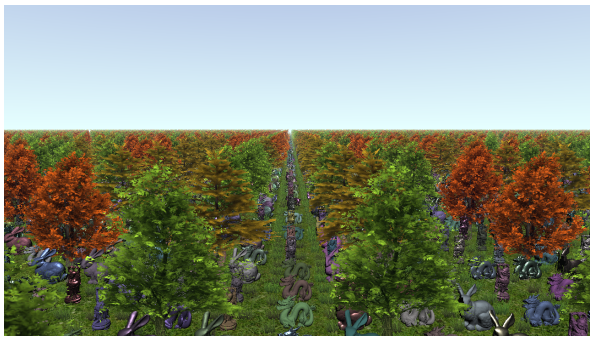


Figure 14: The virtual scene with three SDF components - one for Stanford objects, one for the grass and one for the trees. 23 FPS on Nvidia GeForce GTX 660, 57 FPS on Nvidia GeForce GTX 980.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a novel approach to efficient rendering of an unlimited number of 3D objects in real-time. Thanks to the newly proposed computer graphics paradigm—the *Holistic Graphics Programming*, we created rendering pipeline that can process as many unique instances of 3D objects as we want in real-time. Using a Signed Distance Function, we limited memory requirements for the virtual scene description, making

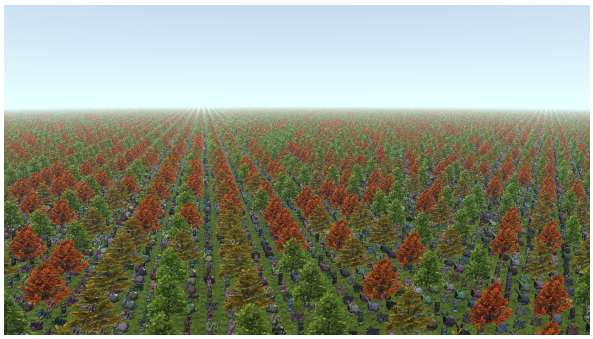


Figure 15: The virtual scene with three SDF components - one for Stanford objects, one for the grass and one for the trees. 29 FPS on Nvidia GeForce GTX 660, 65 FPS on Nvidia GeForce GTX 980.

processing an unlimited number of the 3D object for each SDF component possible. Moreover, taking advantage of a collection of developed *Global operators* we are able to control the content and the complexity of the virtual scene in a procedural way.

Thanks to the Sparse Voxel Octrees integrated with the SDF representation, we are able to render high-resolution 3D objects created by artists. In order to integrate the SDF-based scene with SVO-based objects, we developed an occlusion fixing algorithm. Finally, the developed single pass rendering pipeline can be easily integrated with the e.g. triangle rasterization pipeline for animated, user-controlled objects.

An obvious step forward would be an implementation of *Global operators* that can be used to create dynamic scenes. A good idea seems to be the usage of the dynamic sparse textures to control objects' movement and the existence of the SDF components on the virtual scene. Also, a further optimization and extension for the SDF ray marching rendering pipeline should be considered. For example the distance field based soft shadow should be easy to implement to increase the depth and immersion of the virtual scene.

8 REFERENCES

[Bau11] Bautembach, D., Animated sparse voxel octrees, Bachelor Thesis, University of Hamburg, 2011.

[Cao10] Cao, X., Zhang, Y., Gao, S., Teng, R., Wang, X., The design and implement of Scene Management in 3D engine SR, 2010 International Conference on Mechanic Automation and Control Engineering, pages 183-186.

[Crassin11] Crassin, C., Neyret, F., Sainz, M., Green, S., and Eisemann, E., Interactive indirect illumination using voxel cone tracing, Computer Graphics Forum (Proceedings of Pacific Graphics 2011), vol. 30, no. 7, sep 2011.

[Deussen98] Deussen, O., Hanrahan, P., Lintermann, B., Mesh, R., Pharr, M., Prusinkiewicz, P., Realistic modeling and rendering of plant ecosystems, Proceedings of SIGGRAPH 98, Orlando, Florida, July 19-24, 1998, In Computer Graphics Proceedings, Annual Conference Series, 1998, ACM SIGGRAPH, pages 275-286.

[Domaradzki16] Domaradzki, J., Martyn, T., Fracturing Sparse-Voxel-Octree objects using dynamical Voronoi patterns, Computer Graphics, Visualization and Computer Vision WSCG 2016. Full Papers Proceedings / Pan Zhigeng, Skala Vaclav (red.), Computer Science Research Notes, vol. 2601, 2016, Vaclav Skala - UNION Agency, ISBN 978-80-86943-57-2, pages 37-46.

[Green07] Green, C., Improved alpha-tested magnification for vector textures and special effects, Proceeding SIGGRAPH '07 ACM SIGGRAPH 2007 courses, pages 9-18.

[Hart89] Hart, J., C., Sandin, D., J., Kaufmann, L., H., Ray Tracing Deterministic 3-D Fractals Computer Graphics 23(3), (Proc. SIGGRAPH 89,) July 1989, pages 289-296.

[Hart91] Hart, J., C., DeFanti, T., A., Efficient antialiased rendering of 3D linear fractals. Computer Graphics (SIGGRAPH 91 Proceedings), 25:91-100, 1991.

[Hart92] Hart, J., C., The object instancing paradigm for linear fractal modeling. In Proceedings of Graphics Interface 92, pages 224-231, 1992.

[Hart94] Hart, J., C., Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces, The Visual Computer, Volume 12, pages 527-545.

[Hart97] Hart, J., C., Implicit Representations of Rough Surfaces Computer Graphics forum, Volume 16, Issue 2, June 1997, pages 91-99

[Jab16] Jabłoński, Sz., Martyn, T., Real-Time Rendering of Continuous Levels of Detail for Sparse Voxel Octrees, Computer Graphics, Visualization and Computer Vision WSCG 2016. Short Papers Proceedings / Skala Vaclav (red.), Computer Science Research Notes, vol. 2602, 2016, Vaclav Skala - UNION Agency, ISBN 978-80-86943-58-9, pages 79-88.

[Kampe13] Kämpe, V., Sintorn, E., Assarsson, Ul, High Resolution Sparse Voxel DAGs, ACM Trans. Graph., vol. 32, no. 4, pages 1-13.

[Kay86] Kay, T., L., Kajiya, J., T., Ray tracing complex scenes, Computer Graphics, SIGGRAPH 86 Proceedings, 20(4):269-278, 1986.

[Keinert14] Keinert, B., Schäfer, H., Korndörfer, J., Ganse, U., Stamminger, M., Enhanced Sphere Tracing, STAG: Smart Tools and Apps for Graphics, 2014, pages 1-8.

[Laine10] Laine, S., and Karras, T., Efficient sparse voxel octrees, in Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, ser. I3D 2010. New York, NY, USA: ACM, 2010, pages 55-63.

[Lueb02] Luebke D., Watson B., Cohen, J., D., Reddy, M., and Varshney, A., Level of Detail for 3D Graphics, New York, NY, USA: Elsevier Science Inc., 2002.

[Martyn10] Martyn T., Chaos and graphics: Realistic rendering 3d ifs fractals in real-time with graphics accelerators, Comput. Graph., vol. 34, no. 2, pages 167-175, Apr. 2010.

[Reiner11] Reiner, T., Mückl, G., Dachsbacher, C., Interactive modeling of implicit surfaces using a direct visualization approach with signed distance functions, Computers and Graphics, Volume 35 Issue 3, June, 2011, pages 596-603.

[Stanford11] The Stanford 3D Scanning Repository, Stanford University, 22 Dec 2010, Retrieved 17 July 2011.

[Suther63] Sutherland, I., E., Sketchpad: A man-machine graphical communication system, Proceedings of the Spring Joint Computer Conference, 1963.

[Snyder87] Snyder, J., M., Barr, A., H., Ray tracing complex models containing surface tessellations, Computer Graphics SIGGRAPH 87 Proceedings, 21(4):119-128, 1987.

[Wil13] Willcocks, C. G., Sparse volumetric deformation, Ph.D. dissertation, Durham University, 2013.

[Wright15] Dynamic Occlusion with Signed Distance Fields, Advances in Real-Time Rendering in Games, SIGGRAPH 2015.

Anisotropic Octrees: a Tool for Fast Normals Estimation on Unorganized Point Clouds

Joris Ravaglia
CARTEL
Université de Sherbrooke
Canada
Aix-Marseille Université
LSIS
France
joris.ravaglia@usherbrooke.ca

Alexandra Bac
Aix-Marseille Université
Laboratoire des sciences de l'information et des systèmes (LSIS)
UMR CNRS 7296
France
alexandra.bac@univ-amu.fr

Richard A. Fournier
Centre d'Applications et de Recherche en Télédétection (CARTEL)
Université de Sherbrooke
Sherbrooke (QC) J1K 2R1
Canada
richard.fournier@usherbrooke.ca

ABSTRACT

With the recent advances in remote sensing of objects and environments, point cloud processing has become a major field of study. Three-dimensional point cloud collected with remote sensing instruments may be very large, containing up to several tens of billions of points. This imposes the use for efficient and automatic algorithms to extract geometric or structural elements of the scanned surfaces. In this paper, we focus on the estimation of normal directions in an unorganized point cloud and provide a curvature indicator. We avoid point-wise operations to accelerate the running time for normals estimation. Instead, our method rely on an innovative anisotropic partitioning of the point cloud using an octree structure guided by the geometric complexity of the data and generates patches of points. These patches are then approximated by a quadratic surface in order to estimate the normal directions and curvatures. Our method has been applied to six models of various types presenting different characteristics and performs, in average, 2.65 times faster than multi-threads implementations available in current pieces of software. The results obtained are a compromise between running time efficiency and normals accuracy. Moreover, this work opens up promising perspectives and can be easily inserted in wide range of workflows.

Keywords

Point cloud, normals, curvature, octree, anisotropy, quadratic surface

1 INTRODUCTION

With the current advances in photogrammetry and the availability of instruments recording 3D information (e.g. depth sensors and laser scanning devices from airborne, terrestrial or mobile platforms), point clouds are becoming commonly used in various fields. Data acquired with these tools often contain several tens of millions of points. Processing these large data sets is still a challenge and computing geometric information such as normals or curvature is a real issue. A major difficulty in such processing is the absence of neighbourhood information: unlike meshes, point clouds are unstructured raw data. Consequently, extracting information from a point cloud can be time consuming. Thus, there is a need for new algorithms oriented towards fast point cloud processing.

Normal directions are crucial geometric information for surface analysis. They are a key variable for a large range of algorithms and point clouds analysis approaches [Li08]. Therefore, normal estimation techniques have been widely studied (see review by [Kl09]). However, the estimations of point's normal can be challenging and computationally expensive. Normals estimation can be divided into three major approaches: optimisation-based, local-weighting-based, Hough transform, and Voronoï-based.

The optimisation-based approach locally approximate the point cloud with a predefined surface model from which normals can be computed. The two main models fitted to the data are planes [Ho92], and quadratic surfaces [Kl09]. However other second-order surfaces such as spheres have also been used to approximate the point cloud [Zh16]. Planes can be fitted by using either classical least square fitting, or principal component analysis (PCA) [Gu01]. In local-weighting-based approach, normal directions are estimated by different combinations of the normals deduced from a point and its neighbours [Ji05]. This approach can be ei-

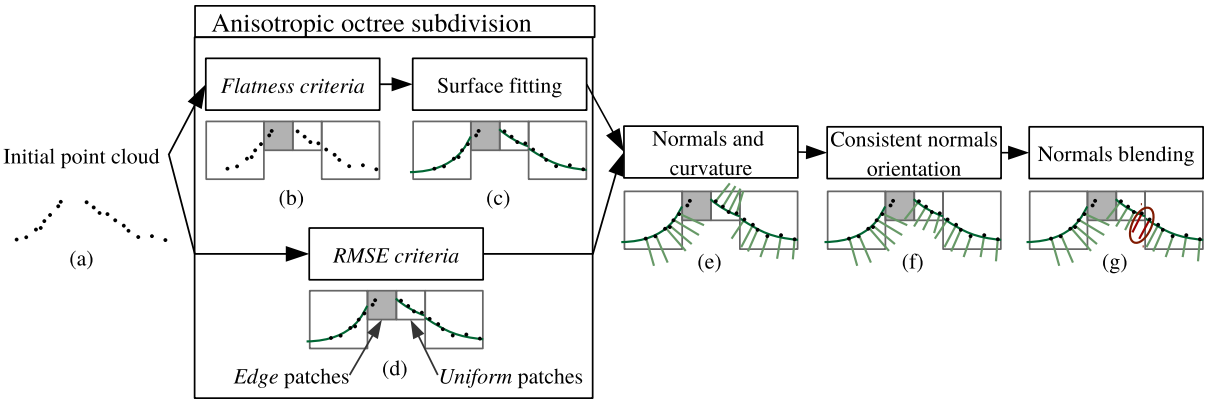


Figure 1: Overview of the methodology.

ther applied on a local Delaunay triangulation of the point cloud, on kNN graphs, or on Reinmann graphs [Kl09]. Hough transform is an alternate approach that considers a point and its neighbourhood. For each point of the cloud a reduced amount of triplets is selected, the Hough transform then provide a reliable accumulator from which robust normal directions can be extracted [Bo12]. This point-wise approach can be supported by a neuronal network to improve its robustness to noise and outliers [Bo16]. The Voronoï-based approach considers the shape of each Voronoï cells [Am99]. The algorithms using Voronoï poles to estimate normal directions may also provide curvature additional information [de05, Me11]. The above approaches are based on point-wise normal estimations and require neighbours finding procedures. While a neighbourhood query can be achieved efficiently with support of accelerating structures such as octrees or KD-trees, the repetition of this procedure for each point is time consuming.

In the present study we propose a new method, following an optimisation-based approach, as a solution to improve the efficiency of normals estimation on large in-core point clouds. Our first specific objective is to avoid point-wise operations. To reach this goal, we developed two distinct procedures that rely on an innovative adaptive subdivision of the point cloud. We propose two novel rules to guide this subdivision in accordance to the local geometric complexity of the point cloud rather than the local density. Our second specific objective is to analyse their efficiency and accuracy in different situations to evaluate their behaviour and the comparative gain in efficiency with currently available algorithms.

2 METHODOLOGY

The normal estimation method we developed is built on a simple idea: surfaces are composed of a combination of *uniform* areas (with respect to the geometrical variation and hence, containing no sharp elements) and sharp features. Consequently, the point clouds resulting from remote sensing of these surfaces also contain uniform and sharp parts. The proposed fast normal estimation procedure takes advantage of this characteristic. We avoided costly point-wise fittings by partitioning the input point cloud into patches in an anisotropic division. Unlike classical subdivisions that are only guided by the point density, our subdivision takes into account the geometry of the data. Indeed, it is designed to split the point cloud anisotropically according to the set of normals. The fast normal estimation procedure is then applied on patches of points instead of point-wise. More precisely, to achieve our objective efficiently, we developed the methodology summarised by the flow diagram of Fig. 1. This methodology is a compromise between running time and normal directions accuracy. The proposed method improves the running time efficiency based on two main aspects. First, it estimates normal directions at a patch level which lowers the computing time . It should also mitigates the impact of noise on normals estimation on large patches in the sense that with larger areas to approximate comes more information about the underlying surface. Selecting a local scale for surface fitting. Second, it provides a curvature indicator and principal curvature directions, which are important geometric features for further processing.

The proposed method has four distinct steps. As demonstrated by Zhao et al. (2016), the use of a point cloud subdivision prior to shape fitting shortens the computational time. The first step of our method addresses patches creation by diverting the octree structure from its initial purpose. Unlike Zhao et al. (2916), who use an octree partitioning guided by the point density, our patch partitioning builds an anisotropic subdivision of the data according to its geometry.

We calibrated the octree subdivision of a point cloud on its local uniformity rather than adopting the usual strategy using the point density. Figure 2 illustrates this anisotropic subdivision: areas containing sharp features such as ground-wall, wall-wall or wall-roof junctions are highly subdivided. whereas bare ground or inner

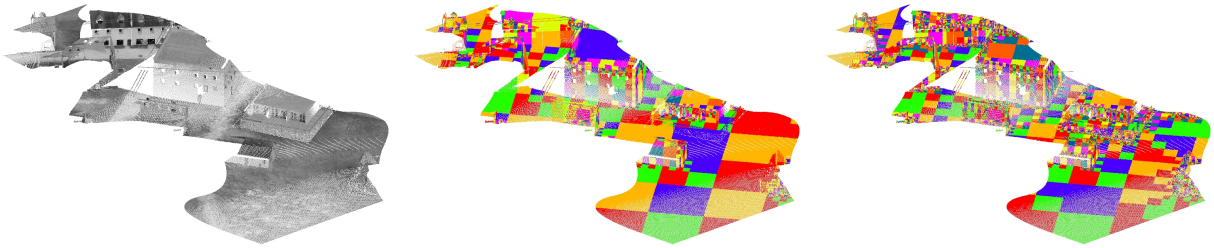


Figure 2: Anisotropic subdivision of an input urban point cloud (left), using σ_3 criterion (centre), or *RMSE* criterion (right). Sharp features creates small patches while uniform surfaces generates large patches. On an almost constant point density, the octree sudvision still creates patches of various size since it is ruled by geometric complexity.

walls are less divided and result in larger patches. This *anisotropic octree* recursively divides a point cloud into two sets of patches (Fig. 1b, 1d): (1) locally smooth patches, and (2) patches containing sharp features (we will refer to them as *edge* patches). The recursive subdivision also includes a surface fit for each generated patch (Fig. 1c, 1d). This way, we generate large patches in even areas and process them at once. Unlike point-wise approaches, our method does not require the computation of a neighbourhood for each point of the input cloud. As a consequence, the complexity of our approach is drastically lower than point-wise approaches. The second step of the method involves estimating normals and curvatures inside each resulting uniform patches using the fitted surfaces (Fig. 1e). Since these surfaces are not oriented, the resulting normal directions are possibly inconsistent across neighbouring patches. Therefore, the third step is designed to improve the coherence of the normal field. To do so, we take advantage of the fast node neighbourhood access given by the octree structure to generate a consistent orientation of normals (Fig. 1f). Eventually, the forth step blends the computed normals and curvatures along nodes boundaries in order to guarantee the smoothness of the resulting normal field (Fig. 1g).

2.1 Anisotropic subdivision

The first step of our method divides the point cloud along an octree subdivision guided by the local uniformity of the point cloud. Starting from the root of the octree $\mathscr{O}$, which contains the entire point cloud, nodes are split into eight octant patches until a uniformity criterion is satisfied.

We developed two procedures to create the anisotropic subdivision. Each of them uses a different criterion to stop the recursive subdivision. The first procedure stops the subdivision when the flatness of the patches is sufficient (Fig. 1b): the corresponding criterion is based on a flatness index (Fig. 3). The second procedure stops the subdivision as soon as the patches can be accurately approximated by a surface model (Fig. 1d): the corresponding criterion thus is based on the modelling error. Both criteria are designed to avoid an over-segmentation of the initial point cloud. Indeed, only patches containing sharp features will be further divided, hence generating large patches for uniform areas, and small patches for sharp features, that is, an anisotropic subdivision. The two procedures have different strengths and balance running time and accuracy, as discussed in the results and the discussion sections. While the first procedure is based on flatness, it requires less computations but captures fewer details on the point clouds. Indeed, figure 2 shows that the σ_3 criterion is more permissive regarding the presence of sharp features and creates patches containing a small amount of sharp features such as ground-wall junctions. The second procedure based on modelling error is more accurate, but involves more calculations. While the verification of the first criterion requires the use of plane fitting operations, quadratic surface fitting is necessary for the evaluation of the second criterion.

2.1.1 *Local plane fitting*

Plane fitting

Let $o \in \mathscr{O}$ be an octree node, $\{p_1, p_2, \cdots, p_m\}$ be the points contained in o, and $\overline{p}$ be their centroid. It has been proven that computing the least square fitted plane P going through $\overline{p}$ and approximating the points p_i can be obtained by a centred PCA of the p_i [Pa02]. Indeed, such a plane is completely determined by its normal $\vec{n}$. Hence, least square fitting is equivalent to minimising the following least square error:

$$\begin{aligned} \sum_{i=1}^{m} \mathrm{d}(p_i, P)^2 &= \sum_{i=0}^{m} \left(\frac{(p_i - \overline{p}) \cdot \vec{n}}{\|\vec{n}\|} \right)^2 \\ &= \frac{1}{\|\vec{n}\|^2} \vec{n}^t \times \mathrm{Cov}(\overline{p_i}) \times \vec{n} \end{aligned} \tag{1}$$

where Cov denotes the covariance matrix, and $\overline{p_i} = p_i - \overline{p}$. Given $\lambda_1 \geqslant \lambda_2 \geqslant \lambda_3$ the eigenvalues of the covariance matrix, and $\vec{v_1}, \vec{v_2}, \vec{v_3}$ the associated eigenvectors, the fitted plane P is actually the plane containing $\overline{p}$ and whose normal direction is $\vec{v_3}$.

Flatness index

The residuals of the PCA plane fitting are given by λ_3. Thus these residuals may be considered as a flatness indicator. However, λ_3 not only depends on the number

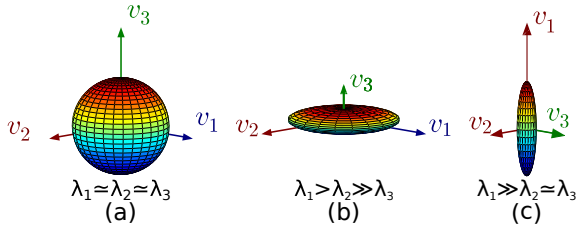


Figure 3: Different point cloud dispersion represented as ellipsoids and corresponding PCA results. Eigenvectors are scaled according the eigenvalue. The colours represents the height map of each ellipsoid for a better appreciation of the shape.

of points considered, but also on the size of the point cloud. Hence we selected an indicator σ_3 called surface variation [Pa02], providing an homogeneous, orientation, and scale independent indicator:

$$\sigma_3 = \frac{\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} \tag{2}$$

From the PCA properties, σ_3 can be seen as the ratio of inertia kept by the plane normal over the total inertia of the point set. This normalisation produces an index $\sigma_3 \in \left[0, \frac{1}{3}\right]$ that is independent of the patch size and of the number of points considered: the lower σ_3, the flatter the corresponding sampled surface (Fig. 3b). This index can be applied as a criterion to regulate the anisotropic recursive subdivision.

2.1.2 *Quadratic surface fitting*

The second procedure models patches as quadratic surfaces for a closer approximation of their geometry. Computing a best-fit surface requires a local frame. An efficient way to obtain this system is to compute an approximate tangent plane using the previously described PCA plane fitting, and to consider the orthogonal resulting basis $B = (\vec{v_1}, \vec{v_2}, \vec{v_3})$, along with $\overline{p}$ its origin.

Given a local frame and a point cloud $\mathscr{P} = \{p_1, p_2, \cdots, p_m\}$ expressed in this basis, we use the classical least square method ([Pr93]) to fit an elevation surface S of equation $S(x,y) = a^2 + bxy + cy^2 + dx + ey + f$ minimising the following distance:

$$d(S, \mathscr{P}) = \sum_{i=1}^{n} (z_i - S(x_i, y_i))^2 \tag{3}$$

The accuracy of this fitting procedure can be evaluated with the corresponding root mean square error (RMSE), which can be used as a criterion to guide the octree subdivision.

2.1.3 *Anisotropic subdivision*

Let us now introduce our anisotropic octree subdivision. Whereas standard octree subdivision iteratively divides voxels according to the relative density of points, we control the subdivision and eventually stop it according to either of the two criteria presented: (1) the σ_3 criterion, and (2) the *RMSE* criterion. The octree creation using the σ_3 criterion performs a PCA plane fitting and stops the subdivision according to a threshold over σ_3 (Fig. 1b). In the *RMSE* criterion, after the initial PCA plane fitting a quadratic surface is fitted and subdivision is stopped according to a threshold over the resulting *RMSE* (Fig. 1d).

The σ_3 criterion therefore requires less computations and as resulting patches have a σ_3 value lesser than a given threshold, they are relatively flat and do not contain folds or sharp features. The *RMSE* criterion requires more calculations since both PCA plane fitting and quadratic surface fitting are computed at each level of the recursive subdivision. However, this criterion assesses more accurately the patches and improves the anisotropy of the resulting subdivision.
Starting from this subdivision, each patch is them modelled by an elevation surface to compute normals. Whereas the *RMSE* criterion directly provides an approximating surface (Fig. 1d), in the case of the σ_3 criterion, we eventually fit a quadratic surface in the least squares sense over each patch. (Fig. 1c).

Regardless of the criterion applied, we set two additional conditions for a patch to be split. At critically small scales, point clouds no longer represent the sampled surfaces. Therefore we use a threshold on the patches dimensions D_{min} to avoid dividing further small nodes. Finally, since we use PCA to locally fit a plane, we set a threshold n_{min} on the number of points contained within the patch to ensure a reliable geometric fitting.

As a result of the recursive binary space division, two main reasons exist for an octree node to include a non uniform patch that cannot be modelled by a surface. The first reason is that nodes may contain less than n_{min} points. This happens: (1) mainly when a node containing an edge is recursively split without reaching the desired level of smoothness, or (2) when a node splitting isolates a few points inside an octant, even if the other nodes reach the smoothness criteria. The second reason for a node not to contain a uniform patch is that the patch is more likely to represent a 3D curve than a surface (Fig. 3c). In order to detect these patches, we use another indicator derived from the PCA plane fitting. These *edge* patches are detected by using the following ratio:

$$\sigma_2 = \frac{\lambda_2}{\lambda_1 + \lambda_2 + \lambda_3} \tag{4}$$

No surface can be fitted on *edge* patches. Similarly, patches not containing enough points are not modelled by a surface since they may contain a sharp feature that cannot be accurately approximated by a quadratic

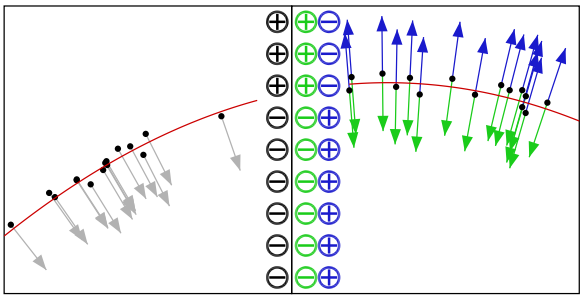


Figure 4: Surfaces are oriented by considering the sign of implicit surfaces at points sampled on the face of contact between neighbour nodes.

model (e.g. several piecewise surface). In both cases, these patches are ignored in the later procedure.

2.2 Normals estimation and curvature indicator

Once uniform patches have been obtained from the anisotropic octree and modelled by a surface S, normals can be estimated in an efficient manner. The normal direction at a point p_i of the patch is approximated by the normal of its projection p'_i onto the surface S. Since the surface equation is known, obtaining the normal $\overrightarrow{n}$ is straight forward. However, at this point, the orientation of the fitted surface cannot be assessed (i.e. discriminating interior and exterior space), and neither do the normal directions.

Similarly to normal directions, principal curvatures can be derived from the fitted surface equation. The first and second fundamental forms of each surface are computed for each point p_i as the curvature of their projection on the fitted surface p'_i. Hence, the curvature tensor can be computed to obtain the corresponding eigenvalues and eigenvectors. However, because the fitted surfaces are quadratic, and since the point cloud division generates uniform patches, the curvature may not be captured with high precision. The resulting eigenvalues of the tensor thus differ from the actual main curvatures of the surface. Nonetheless, this procedure provides a potential indicator of the curvature which can be used to analyse the changes of its value across the point cloud.

2.3 Consistent normal orientation

In order to enhance the consistency of the normal directions, we propose a way to locally orient surfaces. Starting from a patch, neighbouring patches are probed according to their adjacency to propagate a consistent orientation (two patches are considered to be neighbours if their corresponding octree nodes share a common face). Let N_1 and N_2 be two neighbour nodes, S_1 and S_2 be the surface fitted in each node, and f the face shared by N_1 and N_2. In order to orient N_2 relatively to N_1, a set of m points are sampled uniformly on the face f.

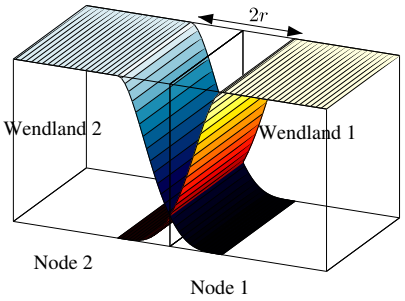


Figure 5: Normals directions close to a node faces are weighted with the neighbours node surface. The weights are computed as a Wendland function around the faces of contact.

	M	CCP	CCQ	CCT	*RMSE*	σ_3
M	1.7	2.4	1.7	0.6	3.2	6.5

Table 1: Average median error for Meshlab (M), CloudCompare's plane fitting (CCP), quadratic fitting (CCQ) and triangulation (CCT), *RMSE* octree, and σ_3 octree.

Now, S_1 and S_2 can be seen as the surfaces corresponding to the zero level set of two implicit functions I_1 and I_2 corresponding to their quadratic equation.. At each point p_i sampled on f, the signs of $I_1(p_i)$ and $I_2(p_i)$ is computed. The orientation of N_2 is set to minimise the difference of signs at the sampled points (Fig. 4).

2.4 Normals blending

Since patches are analysed separately, the junction between the fitted surfaces of two neighbouring patches are discontinuous. Thus, two points inside a small neighbourhood radius may have divergent normals. To minimise this potential bias, we propose a way to blend the normal direction at the nodes junctions taking into account the local information of multiple surfaces. Note N_1, N_2 two neighbouring nodes of size d_1 and d_2 and f the face they share. We define a radius of influence r:

$$r = \frac{min(d_1, d_2)}{3} \tag{5}$$

around f in which normals will be blended (Fig. 5). At each point p, one or several nodes and their surfaces S_i will then influence the estimation of its normal direction. The normal direction $\overrightarrow{\mathcal{N}}_p$ at p will be ultimately estimated as a weighted sum of the normals $\overrightarrow{n_i}$ at p on the set of surfaces S_i:

$$\overrightarrow{\mathcal{N}}_p = \sum_i \omega_i \overrightarrow{n_i} \tag{6}$$

where $\overrightarrow{n_i}$ is the normal at p_i estimated on the surface S_i, and the weights ω_i are computed based a Wendland's function of the distance d_i from p to f_i (Fig. 5):

$$\omega_i(d) = \begin{cases} (1-d)_+^4(4d+1) & \text{if } d < r \\ 0 & \text{else} \end{cases} \tag{7}$$

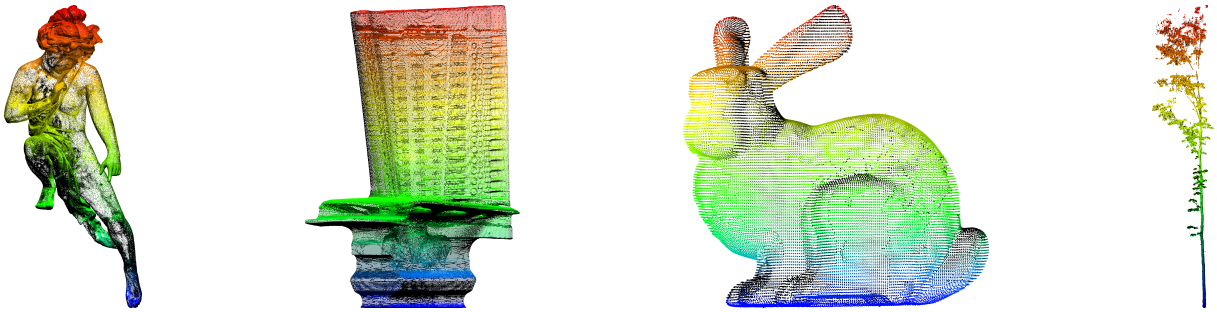


Figure 6: Models used to tests our methods. From left to right: *Angel*, *Blade*, *Bunny*, *Tree* (*Urban* and *Lucy* models are shown in Fig. 2 and 7). Colours are only used to emphasise the point cloud rendering.

In order to blend local estimates smoothly inside a limited radius, we choose to weight normals along boundaries using Wendland's functions. Indeed, such functions are smooth (to any given degree of derivability) while having a compact support. Linear interpolation is clearly not smooth enough whereas Gaussians functions are not compactly supported (whereas we intend to blend estimates only along the boundary).

3 RESULTS

We tested our methodology on six different point clouds with various characteristics and number of points (Fig. 6). These models are indeed representative of a large range of applications: arts, forestry, urban environments and industrial applications. Additionally, while the *Lucy* model contains a higher number of points, the *Angel* model shows variations in point sampling where the point cloud is denser around the details of the statue. Note also that the *Tree* point cloud contains a little noise, mainly around the foliage due to the particular limitations of TLS instruments in forest environments. Four out of six models (*Angel*, *Bunny*, *Blade*, and *Lucy*) are accessible from the Stanford 3D Scanning Repository, or the Large Geometric Models Archive of Georgia Institute of Technology. They were available as a mesh with known normal directions which were used as reference directions. The other two point clouds represent a tree (acquired with terrestrial laser scanning), and a simulated urban environment. These six data sets were used to analyse both the running time efficiency and the accuracy of our method. Results from our method were compared with those obtained by two geometric modelling pieces of software providing normals estimation procedures. In Meshlab, the normals estimation is based on a point-wise k-neighbours operation. CloudCompare includes three different methods: point-wise local plane fitting, point-wise local quadratic surface fitting, and triangulation. Our anisotropic octree subdivision with both criteria (σ_3 and $RMSE$), were also evaluated for a total of six different methods.

The running time for all six normal estimation methods on the six point clouds are presented in Fig. 8 (note the logarithmic scale). We ran the tests on an Intel© Core™ i7-4790 CPU (3.60 GHz). Our methods outperformed the methods in Meshlab and CloudCompare on all tested point clouds in terms of running time. More specifically, the σ_3 octree was faster than the *RMSE* octree. These results can be explained by (1) the higher tolerance of the σ_3 criterion to sharp features which induces less subdivisions of the point cloud, and (2) the need to compute the quadratic surface fitting at each level of recursion when using the *RMSE* criterion, whereas this surface fitting is only computed on the octree leaves when the σ_3 criterion is selected.

The complexity of the anisotropic subdivision is related to the size and the local uniformity of the point cloud. In contrast, since Meshlab and CloudCompare perform point-wise operations, the resulting complexity is then dependent on the number of points in the cloud. In point wise optimisation-based approach, for each point, a neighbour query is required before fitting a surface to the extracted neighbourhood (either with a plane fitting, a quadric surface fitting, or a plane fitting followed by a quadratic surface fitting). Such neighbour finding procedure can be achieved in $\mathcal{O}(log(n))$. Denoting by F the complexity of the chosen fitting procedure and S the complexity of the accelerating structure creation, the overall complexity of point-wise approaches is thus $\mathcal{O}(S+(n*(log(n)+F)))$.
The complexity of our methodology, in the worst case, is that of the octree creation (that is $S=\mathcal{O}(n(d+1)$ where d is the depth of the octree) plus a PCA plane fitting and a surface fitting in each node of the octree. The overall complexity of the normal estimation in an anisotropic octree containing M nodes is thus $\mathcal{O}(S+(M*(PCA+F)))$. In the vast majority of point clouds, $M \ll n$. Hence the low complexity of our approach compared to point-wise approaches witnessed by empirical results.

When comparing computing time, we found that the CloudCompare triangulation method is, on average, the most efficient concurrent of our methods. The *RMSE* anisotropic octree subdivision speeds up, on average, the normals estimation by a factor 2.65 in comparison with the CloudCompare triangulation. This factor rises

Figure 7: Estimated normal directions and curvature on the *lucy* model using the *RMSE* criterion. Point cloud is rendered using estimated normal directions (left), and curvature indicator (right).

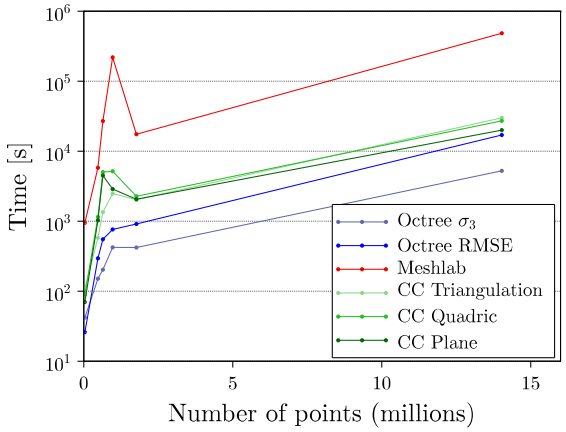


Figure 8: Comparison of computational time for normal directions estimation using different methods.

to 4.92 when considering the σ_3 octree. Overall, the σ_3 octree performs in average 2.09 times faster than the *RMSE* octree.

We also analysed the normal estimation accuracy and computed the curvature indicators for all six methods on four of the point clouds: the *Angel*, *Bunny*, *Blade*, and *Lucy* models (Fig. 7). To do so, we considered the normals of the mesh models as the reference. Hence, we evaluated the estimation error as the angle between the reference normals and the estimated ones (Fig. 9). Let us also point out that no normal vectors were estimated on the points identified as belonging to *edge* patches (corresponding to sharp features). These points are actually discarded during the comparison procedure. The *RMSE* octree produces an error distribution similar to the plane fitting and triangulation methods from CloudCompare but with higher errors on the *Bunny* model. The σ_3 octree however is less accurate with error distributed on a larger range. Average median errors are summarised in table 1. As illustrated in figure 9, even though the values of the curvature indicator differs from the actual Gaussian curvature, the variations of this indicator are actually relevant and capture the reliefs of the statue.

Finally we illustrate the differences between each method, including the tree and urban data sets (Fig. 10). A difference pattern emerges as the anisotropic octree σ_3 induces higher differences for each compared method, and for the majority of data set. Overall, the difference between Meshlab and CloudCompare is regular, and the octree *RMSE* presents low difference variations with these methods. However, the data sets characteristics influence the resulting differences as shown in the *Tree* and *Angel* data sets. The differences observed on the *Tree* data set are due to the high geometric complexity of the analysed object, as well as the occlusion effect inherent to terrestrial laser scanning. The *Angel* point cloud in turn has a non homogeneity of the point sampling on the surface which may impact the neighbourhood search operations. On the contrary, the *Urban* point cloud present lowest inter-methods differences including the σ_3 octree. This is explained by the simplicity of the sampled scene: the point cloud is mainly composed of flat surfaces such as soil, walls, and roofs.

4 DISCUSSION

We achieved our objective and developed a method for fast normal estimation. Indeed, tests have demonstrated the running time efficiency of our methods. The calculations by surface patches instead of point-wise operations reduces the complexity of the normals estimation. Whereas the implementation of our anisotropic subdivisions is mono-thread, the CloudCompare normals estimation methods are multi-thread processes taking advantage of the eight cores available on the computer used during the tests. Thus, using thread partitioning libraries could even increase the benefit in terms of running time. Besides normal directions, our methods also provide a curvature indicator unavailable with the other tested pieces of software.

Comparing our methods on six data sets with different characteristics (number of points, sampling density, and type of model) provided sufficient content for evaluation. Our two methods differ in terms of running time efficiency and accuracy. Whereas the *RMSE* octree performs slower than the σ_3 octree, it generates errors in the range of the Meshlab and CloudCompare methods. The σ_3 octree subdivision, however, produces larger errors, mainly due to its inability to capture details and sharp features. Nonetheless, the σ_3 octree reached the same accuracy as other methods when it was applied on

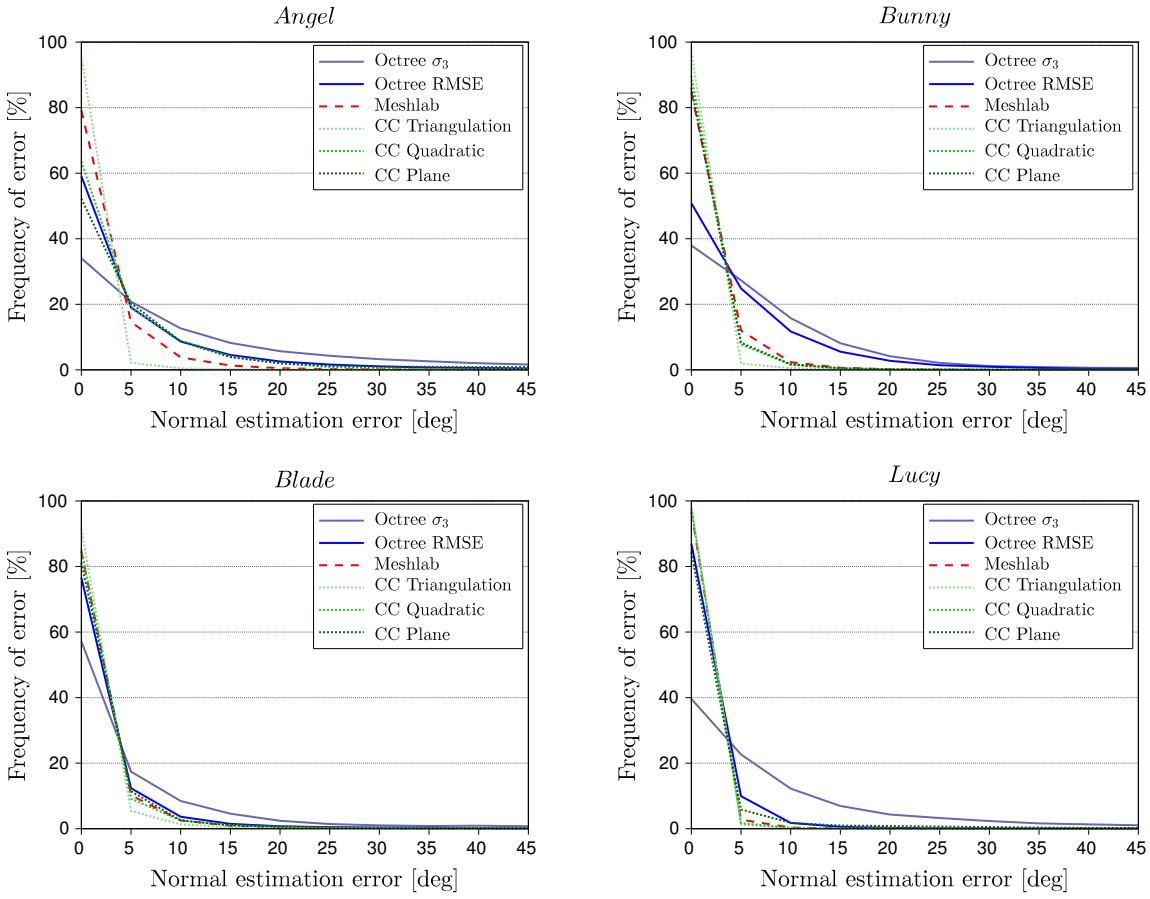


Figure 9: Distribution of error angle between original model normals and estimated normals for different methods. Above 45 degrees, the frequency continues to decrease for each method.

a point cloud with a majority of flat surfaces. Furthermore, patches identified as *edge* cannot be modelled by a surface. Consequently, normals are not estimated on such points, which, in turn, impacts the surface orientation along neighbouring patches.

The impact of noise on the resulting normals has not been studied through an extensive validation yet and is the next step towards a deeper understanding of the capacities of the proposed normal estimation. Robustness to noise was not the main motivation of the presented developments, rather we focused on the balance between computing time and accuracy, as stated earlier (other existing methods were developed to overcome the problem of noisy data). However, we provide some elements to analyse the potential response of our methods. First, given a point cloud affected by a certain level of noise points, processing larger patches (larger than point-wise neighbourhoods) improves surface fitting. This in turn should improve the robustness of normal estimation (let us point out that thanks to these larger patches, we could also detect and remove outliers during surface fitting to mitigate their impact).

When considering a spherical neighbourhood, the chosen radius has to be set carefully: it should be higher than the distance from noisy points to the underlying surface. Otherwise, the local dispersion of the cloud is too important to obtain accurate information. The same rule applies to the consideration of the k-closest neighbours, which generally implicitly engenders a small neighbourhood radius. By contrast, the subdivision into patches allows to analyse a noisy surface over larger areas. In such case, the noise dispersion in less likely to amalgamate noise and pertinent information. Overall, the chosen scale of analysis during the surface fitting is a critical aspect that has to be carefully considered in optimisation-based approaches.

Our work opens up several perspectives in terms of performance improvements and new applications. A way to improve the efficiency of our method is to take advantage of both the σ_3 and *RMSE* criteria to obtain a compromise between efficiency and estimation accuracy. We believe this could be achieved by using a criterion based on the value of λ_3 rather than the σ_3 indicator. We also intend to explore the relevance of trilinear interpolation during normals blending and apply it

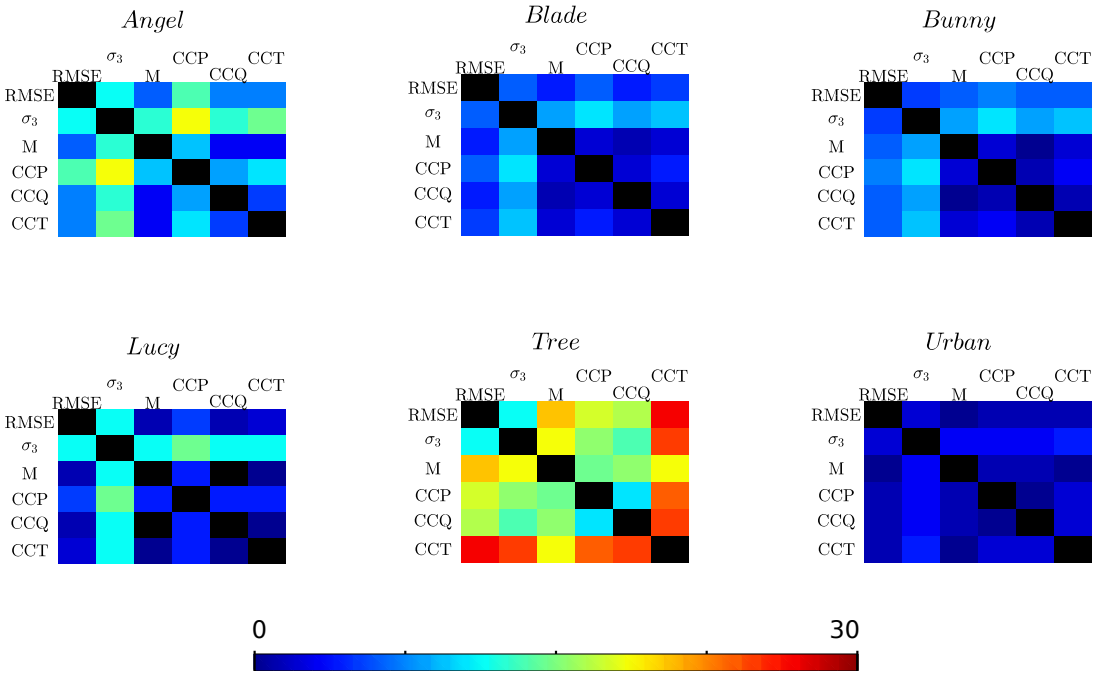


Figure 10: Difference of normal directions estimation in degres between the *RMSE* octree (*RMSE*), the σ_3 octree (σ_3), Meshlab (M) and CloudCompare's plane fitting (CCP), quadratic fitting (CCQ) and triangulation (CCT).

to estimate normals on currently ignored patches. Octree structures are a major element in computer graphics and they take part in many algorithms in order to accelerate the processing. KD-trees are also a popular space partitioning structure. We chose to use octrees because of the regularity of the subdivision (the space is divided into cubes) and because of the efficiency of octrees for neighbourhood searches; both of these properties are further required in other workflows. However, our method could be adapted to KD-trees with few changes. This might even enhance the anisotropy of the resulting procedure since patches could be split at regions of maximum estimated curvature. Doing so would be highly relevant to obtain the larger uniform patches. Another improvement would be to develop these changes in a multi-thread implementation. Additionally, the method could be adapted to handle out-of-core point clouds containing up to several tens of billions of points. Indeed, once patches have been identified and approximated with a quadratic surface, the data points are no longer needed. Thus, the only memory footprint remaining is that of the octree structure itself, which is manageable. Therefore, loading separately appropriate chunks of the point cloud is a viable solution that do not interfere with the normal orientation procedure. Moreover, out-of-core clouds usually represent wide areas (e.g. an entire and detailed building, or parts of a city). With these data, it would be recommended to impose a maximum patch size prior to the octree creation. For example, patches of 15 m^2 would have a great chance of containing sharp features with a junctions of different surfaces such as wall-ground. This patch size limit would fasten the procedure by avoiding several subdivisions and make it easier to handle the amount of data points. Finally we are currently working on a point cloud compression method based on the presented octrees. This lossy compression represents a highly efficient way to store data and could be used for fast computing of levels of details.

5 CONCLUSION

We have presented two new methods to estimate normal directions and curvatures on a point cloud which are major elements in geometric analysis. Both rely on an efficient anisotropic octree subdivision of the data and perform faster than the benchmark pieces of software tested. Results show that the *RMSE* criterion is preferable for capturing great details on the point cloud, whereas the usage of the faster σ_3 criterion has to be restricted to point clouds containing mostly flat surfaces (e.g. urban or indoor environments).

The generated anisotropic octrees represent a multipurpose and efficient tool for point cloud handling. Indeed, besides the normal and curvature estimation presented here, octrees are included in a wide range of applications in computer graphics. As an example they are almost inevitable for visualisation and neighbours finding, and may be involved in segmentation, classification, and spatial analysis. Overall, the proposed methodology can be inserted in various workflows with

few modifications and gives access to elaborated methods based on normals directions.

6 REFERENCES

[Am99] Amenta, N. and Bern, M. Surface reconstruction by Voronoi filtering, Discrete & Computational Geometry, pp.481-504, 1999.

[Bo12] Boulch, A. and Marlet, R. Fast and robust normal estimation for point clouds with sharp features, Computer Graphics Forum, vol. 31, no. 5, pp.1765-1774, 2012.

[Bo16] Boulch, A. and Marlet, R. Deep learning for robust normal estimation in unstructured point clouds, Computer Graphics Forum, vol. 35, no. 5, pp.281-290, 2016.

[de05] Dey, T.K., Li, G. and Sun J. Normal Estimation for Point Clouds: A Comparison Study for a Voronoi Based Method, in Conf.proc. PBG'05, New York, IEEE, pp.39-46

[Gu01] Gumhold, S., Wang, X. and MacLeod, R. Feature extraction from point clouds, in Conf.proc. IMR'01, Newport Beach

[Ho92] Hoppe, H., DeRose, T. and Duchamp, T. Surface reconstruction from unorganized points, in Conf.proc. SIGGRAPH'92, New York, ACM

[Ji05] Jin, S., Lewis, R.R. and West, D. A comparison of algorithms for vertex normal computation, The visual computer, vol. 21, no. 1, pp. 71-82, 2005

[Kl09] Klassing, K., Althoff, D., Wollherr, D., and Buss, M. Comparison of surface normal estimation methods for range sensing applications, in Conf.proc. ICRA'09, Kobe, IEEE

[Li08] Liu, Y., and Xiong, Y. Automatic segmentation of unorganized noisy point clouds based on the Gaussian map, Computer-Aided Design, vol. 40, no. 5, pp. 576-594, 2008

[Me11] Merigot, Q., Ovsjanikov, M., and Guibas, L.J. Voronoi-based curvature and feature estimation from point clouds, IEEE Transactions on Visualization and Computer Graphics, vol. 17, no. 6, pp. 743-756, 2011

[Pa02] Pauly, M., Gross, M., and Kobbelt, L.P. Efficient simplification of point-sampled surfaces, IEEE Transactions on Visualization and Computer Graphics, in Proc.conf. Visualization'02, Boston, IEEE, 2002

[Zh16] Zhao, H., Yuan, D., Zhu, H. and Yin, J. 3-D point cloud normal estimation based on fitting algebraic spheres, in Conf.proc. ICIP'16, Phoenix, IEEE 2016

[Pr93] Press, H., W., Teukolsky, S., A., Vetterling, W., T., and Flannery, B., P. Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press, 1993

Parameterization of unorganized cylindrical point clouds for least squares B-spline surface fitting

Seonghyeon Moon
Shool of Mechanical Engineering
Gwangju Insititute of Science and Technology
61005, Gwangju, South Korea
moonsh@gist.ac.kr

Jin-Eon Park
Shool of Mechanical Engineering
Gwangju Insititute of Science and Technology
61005, Gwangju, South Korea
sigma@gist.ac.kr

Kwanghee Ko
Shool of Mechanical Engineering
Gwangju Insititute of Science and Technology
61005, Gwangju, South Korea
khko@gist.ac.kr

ABSTRACT

In this study, a method for parameterizing unorganized cylindrical point clouds is suggested. The proposed method creates an initial base surface onto which points are projected to estimate parameter values for each point. To produce an initial base surface, we suggest the concept of a virtual turntable and apply multilevel B-splines. Grid points are projected to the point cloud to increase the accuracy of the initial base surface. During the projection process, a modified weight factor is introduced. Lastly, global B-spline surface interpolation is executed to generate an initial base surface, which undergoes a process of refinement and relaxation. Parameter values are then obtained by projecting the point cloud onto the surface orthogonally. Experiments show that the proposed method successfully estimates parameters and solves the problems of self-loop and crossover. Furthermore, the results of experiment show that the proposed weight factor is more effective than the existing weight factor for point directed projection.

Keywords

Parameterization, Base surface, Multilevel B-splines, B-spline surfaces

1 INTRODUCTION

Recently, three-dimensional (3D) scanners have been used to obtain 3D unorganized points called a point cloud. However, such points cannot be utilized in many applications directly because the point cloud possesses only depth or position information. Thus, much research has focused on means by which to obtain geometric information from the point data for use in applications. To obtain geometric properties from point data, a surface representing the point data can be constructed. Various properties can be then obtained from the surface directly. One means of obtaining such a surface is to fit data points using a parametric surface such as a B-spline, where parametrization of data points is required for fitting. A parametric surface can then be obtained from the parameters and the data points in the least squares sense. In particular, surface fitting is a crucial part [VMC97] in reverse engineering, and parametrization is essential in this process. However, a-point-based parameterization method for a cylindrical shape has not been proposed and no robust method has been suggested for parameterization.

Many authors have proposed methods for surface resconstruction. However, the main purpose of our method is improving the parameterization of point clouds for B-spline surface fitting using a surface reconstruction method as an intermediate step. Therefore, a complete review of the papers on surface reconstruction was not included in this paper [BTS17] [GGG08] [KH13].

For parameterization of unorganized points, Ma and Kruth [MK95] proposed a base-surface. Boundary or section curves are used to create a base surface. This surface roughly reflects the shape of the points. Then, parameterization values are estimated by projecting the points onto the surface orthogonally. However, this approach is hard to apply to irregularly scattered data and more complexly shaped point clouds. Azariadis [Aza04] introduced the notion of dynamic base surface (DBS) in which the concept of points projection is applied to improve an initial base surface iteratively by projecting the surface grids onto an input point cloud.

The error function and weight factor are suggested for calculating a point direct projection (DP). DBS achieves parameterization of more complex point clouds by overcoming the limitations of the base surface method. However, DBS encounters problems of self-loops and cross-overs during surface generation and four boundary curves information should be provided to produce an initial base surface. Azariadis subsequently introduced a subdivision technique for a point cloud in order to improve the accuracy of the DBS [AS07]. The author also suggested a new weight factor which not only considers the distance between the projection points and point cloud but also the distance of the axis direction from the point cloud [AS05]. However, this method still assumes that four boundary curves are given and does not consider the problems of self-loops and crossovers. Following the DBS method, many authors have suggested improvements to projection methods. Liu *et al.* [LPY06] proposed a method to determine unknown normal vectors. This approach enables projection without information about the project direction. Du and Liu [DL08] suggested projecting the points onto noisy point clouds. Zhang and Ge [ZG10] introduced a method that generate a surface from sampled points. The generated surface is used for point projection. Yingjie and Liling [YL11] employed a moving least squares (MLS) algorithm for a directed projection. Similar to [ZG10], the MLS surface is used for a local area and the projection is then implemented.

Although all the aforementioned methods presented improved point projection, they did not consider the problems of the DBS approach and still Azariadis's point projection method is easy to apply. Furthermore, no attempts have been made to parameterize a point cloud that represents a cylinder.

This study suggests an approach that can parameterize unorganized cylindrical point clouds while overcoming the problems of the previous methods. This method can parameterize a point cloud by generating an initial base surface close to the point cloud without requiring boundary curve information used by some of the previous approaches. Furthermore, this approach drastically minimizes the problems of self-loops and crossovers and suggests the modified weight factor for projecting points to the point cloud more effectively for the squared distance error computation.

The main contributions of this study are twofold as follows. First, we introduce the concept of a virtual turntable and propose a method of creating an accurate initial base surface by applying the multilevel B-spline method [LWS97] using the hidden point removal technique [KTB07]. Second, we suggest a modified weight factor for point projection to cylindrical point clouds that is more efficient than existing weight factors.

2 BACKGROUND THEORY

In this section, multilevel B-spline approximation [LWS97], points direct projection [Aza04] and dynamic base surface [Aza04] are explained briefly.

2.1 Multilevel B-spline approximation

B-spline approximation

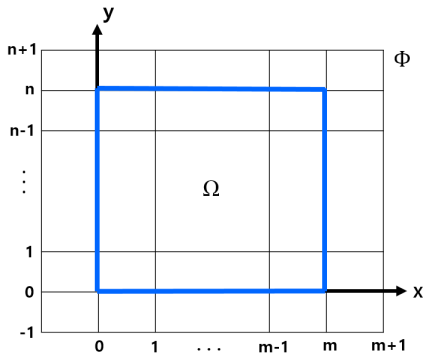


Figure 1: Rectangular Ω domain and lattice Φ

Let Ω = { (x, y) | $0 \le x \le m$, $0 \le y \le n$ } be a rectangular domain in the xy-plane and there is a lattice Φ which has $(m+3) \times (n+3)$ control points as in Fig. 1. Φ_{ij} is the value of the ij-th control point on Φ, located at (i, j) for i = -1, 0, 1, …, m+1 and j = -1, 0, …, n+1. Then, using these control points the following equation is derived that defines a B-spline surface near the position (i, j).

$$F(x,y) = \sum_{k=0}^{3}\sum_{l=0}^{3} B_k(s)B_l(t)\phi_{(i+k)(j+l)} \ , \qquad (1)$$

where $i = \lfloor x \rfloor - 1$, $j = \lfloor y \rfloor - 1$, $s = x - \lfloor x \rfloor$ and $t = y - \lfloor y \rfloor$. B_k and B_l are the uniform cubic B-spline basis functions. Consider there is a point **p** (x_c, y_c, z_c) and 1 $\le x_c, y_c < 2$. Then, Eq. (1) is given as

$$F(x,y) = z_c = \sum_{k=0}^{3}\sum_{l=0}^{3} B_k(s)B_l(t)\phi_{kl} \ . \qquad (2)$$

ϕ_{kl} are control points to be determined. Using the least-squared sense, they are computed by

$$\phi_{kl} = \frac{B_k(s)B_l(t)z_c}{\sum_{a=0}^{3}\sum_{b=0}^{3}(B_a(s)B_b(t))^2} \ . \qquad (3)$$

Suppose there is a point cloud. Each point has its own 4×4 control points by computing Eq. (3). Some of the points close to each other may have overlapped control points. Therefore, to deal with this situation the following equation for ϕ_c is suggested:

$$\phi_c = \frac{B_k(s)B_l(t)z_c}{\sum_{a=0}^{3}\sum_{b=0}^{3}(B_a(s)B_b(t))^2} \qquad (4)$$

and ϕ_{ij} is chosen to minimize the error $e(\phi_{ij}) = \sum_{k=0}^{3}\sum_{l=0}^{3}(B_k(s)B_l(t)\phi_{ij} - B_k(s)B_l(t)\phi_c)^2$. Then, differentiating $e(\phi_{ij})$ with respect to ϕ_{ij} gives

$$\phi_{ij} = \frac{\sum_{k=0}^{3}\sum_{l=0}^{3}(B_k(s)B_l(t))^2\phi_c}{\sum_{k=0}^{3}\sum_{l=0}^{3}(B_k(s)B_l(t))^2} \ . \tag{5}$$

When there are shared control points ϕ_{ij}, Eq. (5) provides the least squared solution for minimizing a local approximation error. If there is no overlapped control points, a zero value is allocated to ϕ_{ij}. Alternatively, the following equation also suggested [Hje01], which is less sensitive to outliers.

$$\phi_{ij} = \frac{\sum_{k=0}^{3}\sum_{l=0}^{3}(B_k(s)B_l(t))\phi_c}{\sum_{k=0}^{3}\sum_{l=0}^{3}(B_k(s)B_l(t))^2} \tag{6}$$

Multilevel B-spline approximation

Consider a control lattice ϕ_0 of $(m+3)\times(n+3)$. Then the difference between the real values and the approximation is calculated and used for generating the next control lattice ϕ_1 of $(2m+3)\times(2n+3)$. The ij-th control point in ϕ_k corresponds to $(2i,2j)$-th control point in ϕ_{k+1}. Again, approximation of ϕ_1 is used for calculating the difference and used to obtain next the control lattice ϕ_2. Repeating this process finds a surface more accurate to the given points.

2.2 Point directed projection (DP)

Assume that there is a point cloud consisting of n points: $\mathbf{p}_d = (x_d, y_d, z_d),\ d = 0,1,\cdots n-1$ with a_d. Let us assume that there is a point $\mathbf{p} = (x,y,z)$ to be projected onto the point cloud. Then, the sum of the weighted squared distances can be computed:

$$\begin{aligned} E(\mathbf{p}) &= \sum_{d=0}^{n-1} a_d \|\mathbf{p}-\mathbf{p}_d\|^2 \\ &= \sum_{d=0}^{n-1} a_d[(x-x_d)^2+(y-y_d)^2+(z-z_d)^2] \ . \end{aligned} \tag{7}$$

Eq. (7) can be computed quickly using a five dimensional vector $\mathbf{c} = (c_0, c_1, c_2, c_3, c_4)$ by [EM99].

$$\begin{aligned} &c_0 = \sum_{d=0}^{n-1} a_d, \quad c_1 = \sum_{d=0}^{n-1} a_d x_d, \quad c_2 = \sum_{d=0}^{n-1} a_d y_d \ , \\ &c_3 = \sum_{d=0}^{n-1} a_d z_d, \quad c_4 = \sum_{d=0}^{n-1} a_d({x_d}^2+{y_d}^2+{z_d}^2) \ , \end{aligned} \tag{8}$$

$$E(\mathbf{p}) = c_0(x^2+y^2+z^2) - 2(c_1x+c_2y+c_3z)+c_4 \ . \tag{9}$$

Here, let us assume that there is $\mathbf{p}^* = (x^*, y^*, z^*)$, the result of projection onto the point cloud along the direction of projection is $\mathbf{n} = (n^x, n^y, n^z)$.

$$\mathbf{p}^* = \mathbf{p}^*(t) = \mathbf{p} + t\mathbf{n} \tag{10}$$

Using Eq. (10), Eq. (9) can be written as

$$\begin{aligned} E(\mathbf{p}^*(t)) =& c_0((x^*(t))^2+(y^*(t))^2+(z^*(t))^2) \\ &- 2(c_1x^*(t)+c_2y^*(t)+c_3z^*(t))+c_4 \ . \end{aligned} \tag{11}$$

Eq. (11) is differentiated with respect to t to find a minimum.

$$\frac{dE(\mathbf{p}^*(t))}{dt} = E'(\mathbf{p}^*(t)) = 0 \ \Rightarrow \ t = \frac{\lambda - \mathbf{pn}}{\|\mathbf{n}\|^2} \ , \tag{12}$$

$$\lambda = \frac{(c_1n^x + c_2n^y + c_3n^z)}{c_0} \ , \tag{13}$$

$$\frac{d^2E(\mathbf{p}^*(t))}{dt^2} = E''(\mathbf{p}^*(t)) = 2c_0\|\mathbf{n}\|^2 \ > \ 0 \ . \tag{14}$$

Eq. (14) demonstrates that the result of Eq. (12) is a minimum solution for Eq. (11). Then, this t is utilized for projecting the point $\mathbf{p}$ to the point cloud using Eq. (10). During the projection process, the point's weight affects the result of projection. Therefore, two weight factors for effective point projection were suggested [Aza04] and [AS05].

$$a_d = \frac{1}{\|\mathbf{p}-\mathbf{p}_d\|^4}, \quad a_d = [0,\infty] \ , \tag{15}$$

$$a_d = \frac{1}{1+\|\mathbf{p}-\mathbf{p}_d\|^2\|(\mathbf{p}_d-\mathbf{p})\times\mathbf{n}\|^2}, \quad a_d = [0,1] \ . \tag{16}$$

Eq. (15) imposes strong weight factors to the nearest points to $\mathbf{p}$ and Eq. (16) adds the axis distance from points to the axis direction to give more weights near the direction of axis $\mathbf{n}$.

2.3 Dynamic base surface (DBS)

An initial base surface can be created by using the given four boundary curves of a point cloud. Then, a grid on the surface is computed. Using the concept of the point directed projection, the grid is projected to the point cloud and these grid points are interpolated to form a new surface. If the surface does not satisfy user-defined thresholds, then computing the grid and DP is applied again. Dynamically the initial base surface is improved to approximate the point cloud accurately. Therefore, this method is called the dynamic base surface (DBS).

3 PROPOSED METHOD

3.1 Overall process of the proposed method

The main idea of the proposed method is to produce an initial base surface close to a point cloud. Multilevel B-splines are used for this purpose. However, the multilevel B-splines method can only be applied to 2.5D data. Therefore, a virtual turntable is introduced and the hidden point removal technique [KTB07] is employed

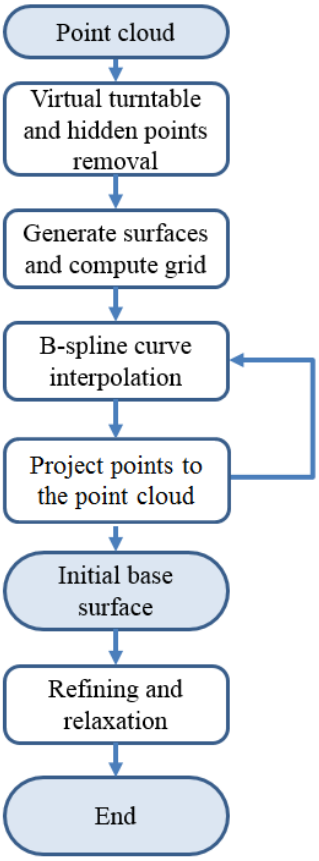


Figure 2: The flowchart of the proposed method

to transform the point cloud to 2.5D data. The transformed data are used as input for the multilevel B-spline method. Surfaces are generated and grid points on the surfaces are computed. These points in the same layer are used for B-spline curve interpolation and projected to the point cloud. This process is repeated until the user-defined terminate condition is satisfied. An initial base surface is then generated by interpolating the grid points and it undergoes a process of refining and relaxation.

The simple human head point cloud (Fig. 3) is used to describe the proposed method. Fig. 2 shows the outline of the proposed algorithm and the following subsections explain each step in detail.

3.2 Virtual turntable and hidden points removal

Motivated by the real scanning process, we introduce a virtual turntable concept for the next step of generating surfaces using multilevel B-spline [LWS97]. However, the multilevel B-spline method is not applicable to data points sampled from a nonplanar surface. Therefore, we convert the point cloud to 2.5D data by combining the virtual turntable with the hidden point removal method [KTB07].

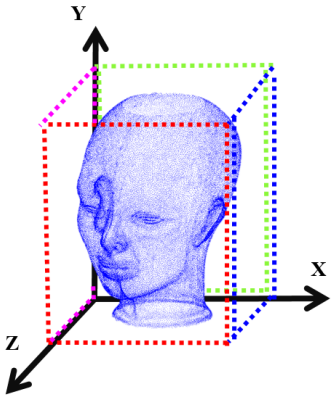


Figure 3: Simple human head point cloud on the xz-plane.

Assume that there is a point cloud $\mathbf{p_i} = (p_{ix}, p_{iy}, p_{iz})$, $(i = 1, \cdots, n)$. The point cloud is projected to the xz-plane and the two points the distance of which is the largest are selected for making a bounding sphere by using an efficient bounding sphere [Rit90]. Let c_0 be the middle of these two points. Then we have

$$P_i' = M \begin{pmatrix} p_{ix} - c_{0x} \\ p_{iy} - c_{0y} \\ p_{iz} - c_{0z} \end{pmatrix} . \tag{17}$$

Using this c_0, the point cloud's center can be moved to the y-axis by computing Eq. (17) and the point cloud can be rotated maintaining its center position by the y-axis rotation matrix $\mathbf{M}$. The result is named as P_i'.

The view position is set to be oriented in the z-axis direction from $+z$ to $-z$ for applying the hidden point removal method.

$$P_i'' = M^{-1} \begin{pmatrix} p_{ix}' \\ p_{iy}' \\ p_{iz}' \end{pmatrix} + \begin{pmatrix} c_{0x} \\ c_{0y} \\ c_{0z} \end{pmatrix} \tag{18}$$

The point cloud P_i' can be moved to the original position P_i'' by computing Eq. (18). By rotating 90 degrees, four different results are acquired and moved to the original position after the hidden point removal process (Fig. 4).

3.3 Generate surfaces and compute grid points

After the point cloud is rotated and the hidden points are removed, the multilevel B-spline method is applied using the xy plane control lattice. The generated surfaces are moved to the origin location (Fig. 5). Then, the equal number of parameters u and v are used to produce the surface grid points. Using the u direction gird points positioned at the same height (y coordinate), the closest

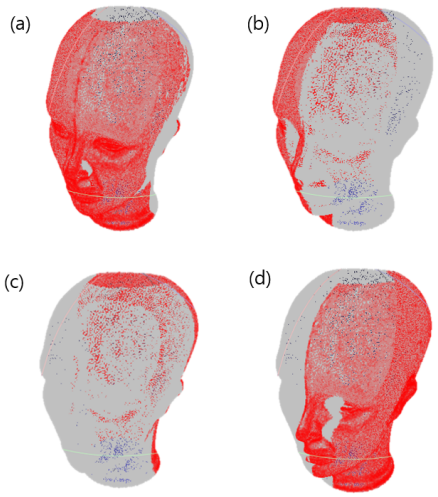


Figure 4: Results of hidden points removal. Red points are visible. (a) 0° rotation, (b) 90° rotation, (c) 180° rotation, (d) 270° rotation.

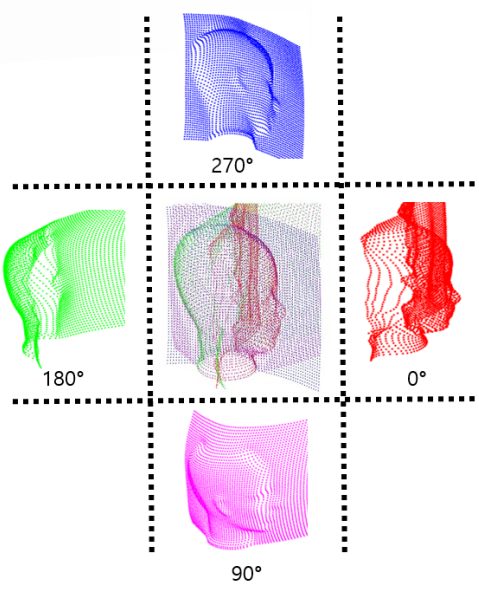


Figure 5: Generated four surfaces using the multilevel B-spline.

pairs between the two surfaces are treated as intersection pairs. By analysis of these pairs, the grid points irrelevant to the point cloud are discarded. Note that the surface should produce the same number of grid points for finding the intersection pairs and the u direction is parallel to the xz-plane. During the implementation of the multilevel B-spline method, the control lattice resolution affects the accuracy of the generated surfaces.

3.4 B-spline curve interpolation

The remaining grid points are used for global B-spline curve interpolation [PT97] along the row direction (u direction) layer by layer (Fig. 6). The column grid points close to the point cloud are selected as the starting and ending interpolation points. Interpolations are implemented through the remaining points after outliers are removed. Uniform sampling is then implemented based on the numbers of points remaining in each layer. During the interpolation process, parameterization is conducted by using the chord length method and knot vectors are computed by using the averaging method [MK95]. All grid points are then interpolated layer by layer. The results of all curves are then identified as the initial curves. Note that each layer has a different number of sample points as a result of using uniform sampling. These sample points are employed following the step of point projection.

3.5 Project points to the point cloud

The grid points acquired by uniform sampling are employed to project to the point cloud. Projection directions are attained by the derivative of B-spine curves. Directed projection (DP) approach is applied for the projection process. In the experiment, the projection using the weight factors Eq. (15) or Eq. (16) does not produce different results. Therefore, the following modified weight factor is used.

$$a_u = \frac{1}{\|\mathbf{p} - p_u\|^4 \|(p_u - \mathbf{p}) \times \mathbf{n}\|^2}, \quad a_u = [0, \infty] \quad (19)$$

After projection to the point cloud, resampling and projection are repeated until the user defined threshold is satisfied to increase the accuracy of each curve.

3.6 Initial base surface

After projection is terminated, the same number of samples is implemented by each curve for the global B-spline surface interpolation [PT97]. This is because each curve has a different number of points. If the number of samples is inadequate, the accuracy of the surface is degraded. Therefore, the sampling number is set to the number of points in the curve that has the highest number of samples. During surface interpolation, parameterization is performed through uniform parameterization. In addition, non-uniform knot vectors are selected. An initial base surface is then generated by interpolating points from the final curves.

3.7 Refining and relaxation

After the initial B-spline surface is generated, refining and relaxation are conducted. In the former, the grid is projected over the point cloud along the surface normal direction. The latter is the method used in the DBS approach. This grid is then used to create a B-spline surface again. Finally, we obtain a B-spline surface that approximates the point cloud, and the parameters are obtained by projecting the point cloud onto the surface orthogonally.

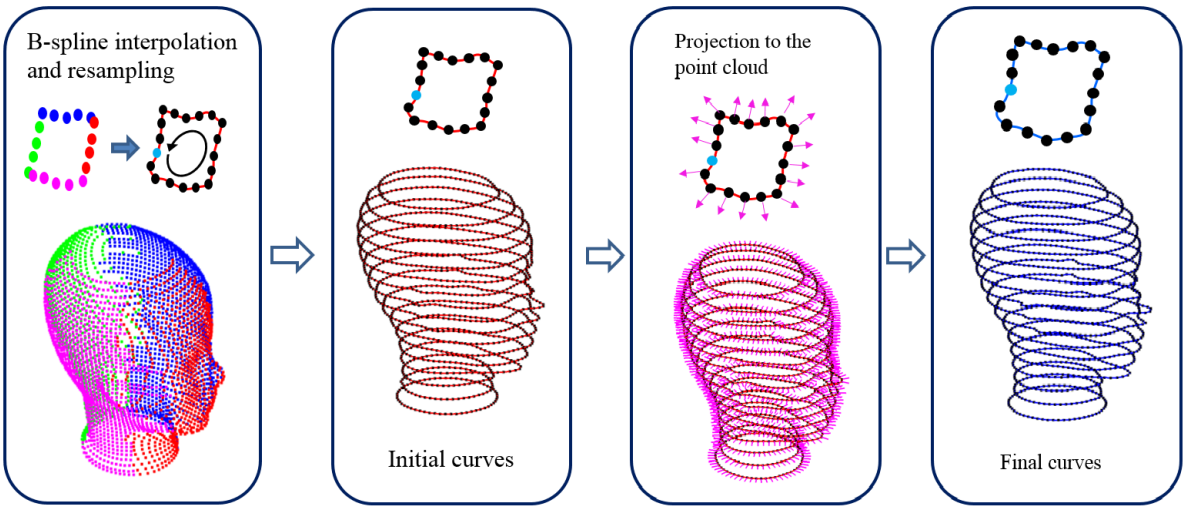


Figure 6: Process of B-spline curve interpolation and projection to the point cloud.

4 RESULTS AND DISCUSSION

The simple human head point cloud, the Lisa skull head point cloud, and the Stanford bunny point cloud are all employed to demonstrate the proposed method. Surfaces are generated to approximate each point cloud and parameterization is achieved by projecting the point cloud onto the surface orthogonally. Parameterized points are then exploited for the least squares B-spline surface fitting in order to show that the parameterization has been successfully accomplished. Figs. 7a-9a show the point cloud used to test the proposed method of parameterization and the point cloud is approximated by the final surface (Figs. 7b-9b). After parameterization using the final surface, least squares B-spline surface fitting is implemented as shown in Figs. 7c-9c.

In our experiment, final surfaces accurately approximated point clouds and parameterization was implemented successfully as shown in Figs. 7-9. Compared to the DBS method [Aza04], the proposed method is much simpler and minimizes user intervention for problems of self-loops and crossovers, as initial base surfaces are generated close to the point cloud without boundary curve information. To avoid these problems, each curve and the projection directions are restricted to a plane. This approach drastically minimizes problems.

In addition, the modified weight factor can project a point to point clouds in few iterations (Figs. 10-11). Detailed results are listed in Tables. 1-2. In the experiment, the modified weight factor in Eq. (19) was more efficient than those of Eq. (15) or Eq. (16) in terms of RMSE accuracy and speed. Eq. (15) requires more iterations to project a point onto a point cloud because the value of t in Eq. (10) is small when the points that are far from the projection direction have strong weights. Eq. (16) overcomes this problem by introducing the axis distance. However, it is inefficient in the surrounding point clouds. Points near the axis of direction that is behind projection direction also get strong weights. By Comparison, the modified weight factor using Eq. (19) is efficient for the surrounding point clouds, because it not only considers the projection direction but also gives stronger weights to near points.

Note that the values of root mean square error(RMSE) are calculated using the minimum distances between the point cloud and projection points.

5 CONCLUSION

This study presented a method for parameterization of unorganized cylindrical point clouds. A virtual turntable concept was introduced for applying the multilevel B-splines method to a point cloud. The multilevel B-splines method can generate a surface close to a point cloud in less time. This more accurate surface reduces the problems of self-loops and crossovers for the projection process. Furthermore, four boundary curves are not required to generate an initial base surface. During projection process, a modified weight factor can efficiently project points onto a point cloud. In our study, selected point clouds were tested to demonstrate the proposed method. Our experiments confirmed that parameterization was successfully accomplished, as shown in Figs. 7-9.

However, many possibilites remain for improving the proposed method. Using the T-spline would generate a surface that is efficient and accurate without having to resample for the global B-spline surface interpolation. Decreasing or increasing rotation angles of a virtual turntable based on the complexity of point clouds would make the proposed method more robust. These are recommended suggestions for future work.

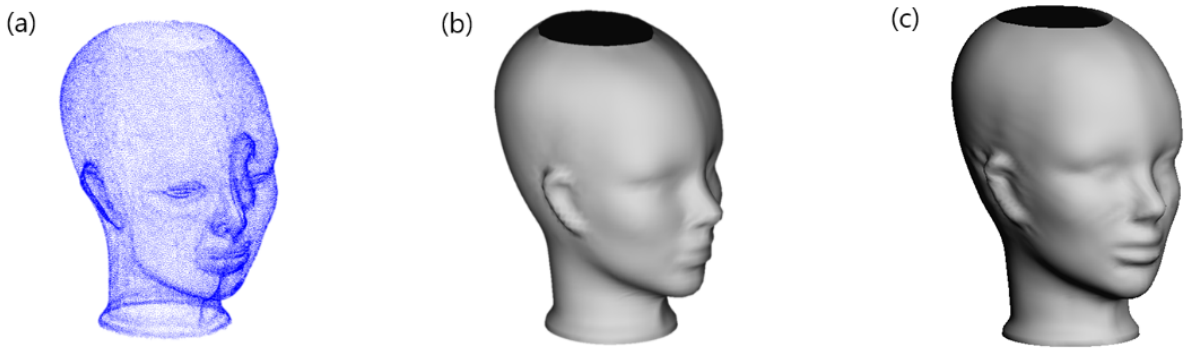


Figure 7: Result of the proposed method and application in least squares B-spline surface fitting. (a) The simple human head point cloud (N=64009, Bounding box(158x246x190)). (b) The final surface (51x101 control points, RMSE=0.221). (c) Result of the B-spline surface fitting (55x55 control points).

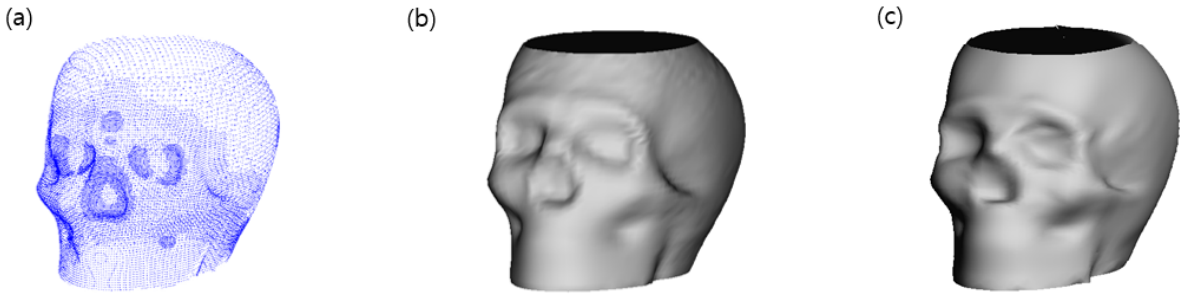


Figure 8: Result of the proposed method and application in least squares B-spline surface fitting. (a) The Lisa skull head point cloud (N=28384, Bounding box(22x182x234)). (b) The final surface (37x120 control points, RMSE=0.144). (c) Result of the B-spline surface fitting (39x39 control points).

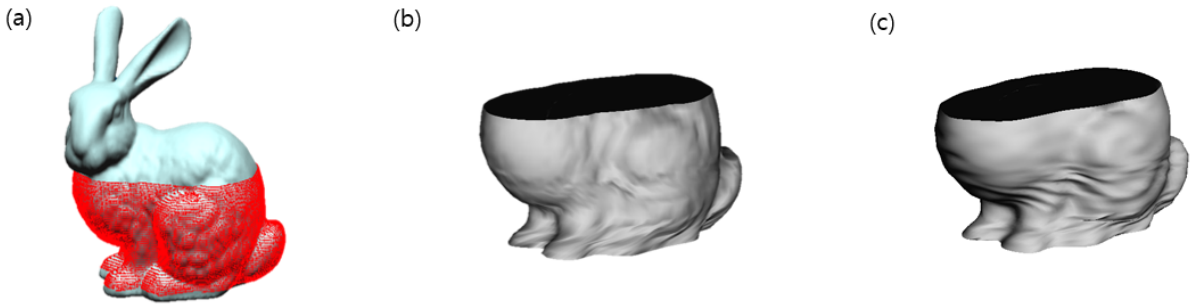
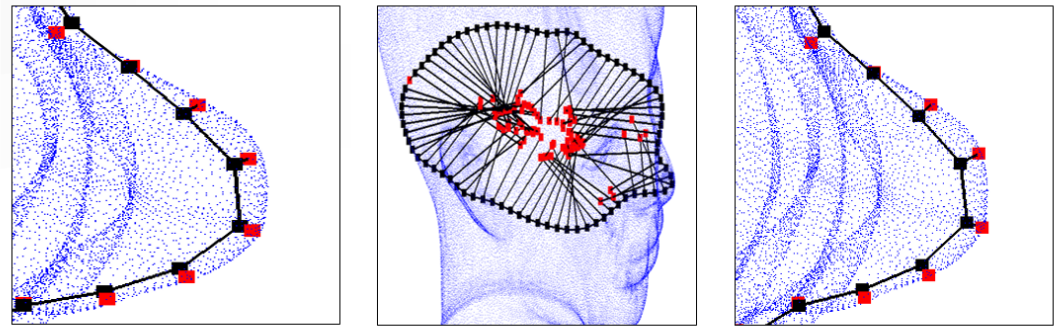


Figure 9: Result of the proposed method and application in least squares B-spline surface fitting. (a) Part of the Stanford bunny point cloud (N=16234, Bounding box(270x117x138)). (b) The final surface (24x110 control points, RMSE=0.184). (c) Result of the B-spline surface fitting (30x30 control points).

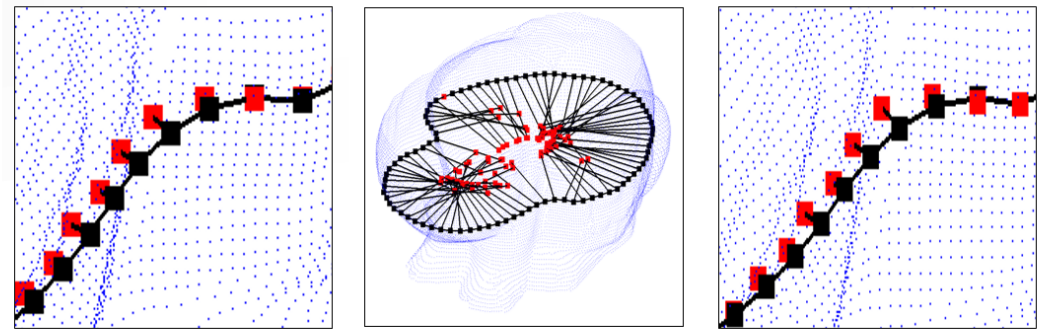


(a) (b) (c)

Figure 10: Results of a single iteration projection to the simple human head point cloud based on the weight factor. Black points are sampled from a B-spline curve. Red points are results of projection. (a) DP method using Eq. (15) (b) DP method using Eq. (16) (c) DP method using the proposed Eq. (19).

Table 1: Comparison of RMSE errors between the proposed weight factor Eq. (19) and DP method using Eq. (15) during projecting points to the simple human head point cloud.

Iteration	Proposed weight factor (RMSE)	Azariadis's 2004 factor (RMSE)
0	0.289	0.289
1	0.240	0.252
2	0.236	0.252
3	0.236	0.248
4	0.236	0.247
5	0.236	0.245



(a) (b) (c)

Figure 11: Results of a single iteration projection to part of the Stanford bunny point cloud based on the weight factor. Black points are sampled from a B-spline curve. Red points are results of projection. (a) DP method using Eq. (15) (b) DP method using Eq. (16) (c) DP method using the proposed Eq. (19).

Table 2: Comparison of RMSE errors between the proposed weight factor Eq. (19) and DP method using Eq. (15) during projecting points to part of the Stanford bunny point cloud.

Iteration	Proposed weight factor (RMSE)	Azariadis's 2004 factor (RMSE)
0	0.305	0.305
1	0.168	0.181
2	0.149	0.151
3	0.134	0.148
4	0.126	0.141

6 ACKNOWLEDGMENTS

This work was supported by the GIST Research Institute(GRI) in 2017.

7 REFERENCES

[AS05] Azariadis, P.N., and Sapidis, N.S. Drawing curves onto a cloud of points for point-based modelling, Computer-Aided Design, Volume 37, Issue 1, January 2005, Pages 109-122.

[AS07] Azariadis, P.N., and Sapidis, N. Product design using point-cloud surfaces: A recursive subdivision technique for point parameterization, Computers in Industry, Volume 58, Issues 8-9 December 2007, Pages 832-843.

[Aza04] Azariadis, P.N. Parameterization of clouds of unorganized points using dynamic base surfaces, Computer-Aided Design, Volume 36, Issue 7, June 2004, Pages 607-623.

[BTS17] Berger, M., Tagliasacchi, A., Seversky, L. M., Alliez, P., Guennebaud, G., Levine, J. A., Sharf, A. and Silva, C. T. (2017), A Survey of Surface Reconstruction from Point Clouds. Computer Graphics Forum, 36:301-329.

[DL08] Du, M.C., and Liu Y.S. An extension on robust directed projection of points onto point clouds, Computer-Aided Design, Volume 40, Issue 5, May 2008, Pages 537-553.

[EM99] Erikson, C., and Manocha, D. GAPS: general and automatic polygonal simplification. In Proceedings of the 1999 symposium on Interactive 3D graphics (I3D '99). ACM, New York, NY, USA, 79-88, 1999.

[GGG08] Guennebaud, G., Germann, M. and Gross, M. (2008), Dynamic Sampling and Rendering of Algebraic Point Set Surfaces. Computer Graphics Forum, 27: 653-662.

[Hje01] Hjelle, Ø. Approximation of scattered data with multilevel b-splines, Tech. Rep. STF42 A01011, Oslo (2001).

[KH13] Kazhdan, M., and Hoppe, H. Screened poisson surface reconstruction. ACM Trans. Graph. 32, 3, Article 29 (July 2013), 13 pages.

[KTB07] Katz, S., Tal, A., and Basri, R. Direct visibility of point sets. In ACM SIGGRAPH 2007 papers (SIGGRAPH '07). ACM, New York, NY, USA, Article 24.

[LPY06] Liu,Y.S., Paul, J.C., Yong, J.H., Yu, P.Q., Zhang, H., Sun, J.G., Ramani, K. Automatic least-squares projection of points onto point clouds with applications in reverse engineering, Computer-Aided Design, Volume 38, Issue 12, December 2006, Pages 1251-1263.

[LWS97] Lee, S., Wolberg, G., and Shin, S.Y. Scattered Data Interpolation with Multilevel B-Splines. IEEE Transactions on Visualization and Computer Graphics 3, 3 (July 1997), 228-244.

[MK95] Ma, W., and Kruth, J.P. Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces, Computer-Aided Design, Volume 27, Issue 9, 1995, Pages 663-675.

[PT97] Piegl, L., and Tiller, W. The NURBS Book (2nd Ed.). Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[Rit90] Ritter, J. An efficient bounding sphere. In Graphics gems, Andrew S. Glassner (Ed.). Academic Press Professional, Inc., San Diego, CA, USA 301-303, 1990.

[VMC97] Várady, T., Martin, R.R., and Cox, J. Reverse engineering of geometric models - An introduction, Computer-Aided Design, Volume 29, Issue 4, 1997, Pages 255-268,

[YL11] Yingjie, Z., and Liling, G. Improved moving least squares algorithm for directed projecting onto point clouds, Measurement, Volume 44, Issue 10, December 2011, Pages 2008-2019.

[ZG10] Zhang, Y.J., and Ge, L.L. A robust and efficient method for direct projection on point-sampled surfaces, International Journal of Precision Engineering and Manufacturing 11 (1) (2010) 145-155.

Volume estimation of biomedical objects described by multiple sets of non-trimmed Bézier triangles

Aleksandrs Sisojevs
Ventspils University Colleges
101a Inzhenieru Str.
LV-3601, Ventspils, Latvia
alexiv@inbox.lv

Mihails Kovalovs
Riga Technical University
1 Setas Str.
LV-1048, Riga, Latvia
mihails.kovalovs@rtu.lv

Olga Krutikova
Riga Technical University
1 Setas Str.
LV-1048, Riga, Latvia
olga.krutikova@rtu.lv

ABSTRACT

Estimating the volume of a 3D model of an object is an actual task in many scientific and engineering fields (for example, CAD systems, biomedical engineering tasks etc.). Spline surfaces is one of the most powerful and flexible methods used to describe a 3D model. At the same time, it is rather difficult to estimate the volume of an object described by spline surfaces. A model of a Bézier triangle is a simple type of a spline surface, but it is practically advantageous.

This paper describes a method of estimating the volume for 3D objects that are described by a set of Bézier triangles. The proposed method was tested on 3D models of objects of biomedical origin. A theorem is presented in this paper for volume estimation, based on different properties of researched models, acquired by a projection of a set vertices of a Bézier triangle onto a coordinate system axis. The proposed approach is based on using methods of differential geometry: surface integrals of scalar fields, Euler's integral of the first kind and Beta functions. Experimental results prove the accuracy of presented theorems. The proposed method can be successfully used to calculate the volume of different 3D models, including objects of biomedical origin.

Keywords

Beta function, Bézier triangles, volume estimation.

1. INTRODUCTION

Determining the volume of a three-dimensional surface can be a very complex task, depending on the object's description method. There are different ways a 3D model of an object can be described. Most commonly used approaches are: approximation by primitives, polygonal mesh approximation or use of parametrical surfaces [Sun13a, Wro06a]. While different methods exist for model description, it should be noted that spline surfaces have a significant role in 3D modeling of complex objects, because they can describe curved models with high precision. At the same time, the problem of precise volume estimation can be very critical in many areas. For example, exact volume estimation of biomedical objects can significantly increase the accuracy and reliability of medical diagnosis. Therefore, the use of a precise model and its volume estimation can have a significant impact on biomedical engineering. In work [Sis09a] the author describes a volume estimation method for models of biomedical objects bounded by Bézier surfaces. In [Duer03a] several quantification methods for calculating the volume of soft tissues are discussed that include Geometric Best-Fitting, based on shape assumptions and stereological procedures. In [Matt87a], [Mich88a], [How93a], [McNul00a], the Cavalieri method is discussed which is a quantification method that is commonly used for unbiased estimation of the volume of a variety of biological objects from serial histological sections. In [Mor98a], [Duer00b] a different method is proposed that is used to calculate the volume of a reconstructed 3D model. In these works, two algorithms (Isocontouring and Surface Tiling) for 3D surface reconstruction were tested. These algorithms were used to construct an accurate wireframe surface of a biological object and calculate its volume [Baj96a]. The provided results showed minor quantitative differences between the Isocontouring, Surface Tiling and other standard qualitative approaches [Duer00b].

Calculating the volume of solids bounded by Bézier Surfaces in analytical form, based on integral calculations, was discussed in [Juh00a].

A similar idea was described in [Zha01a], but it was used to calculate the area of a closed 2D mesh and the volume of a bounded 3D mesh.

Approximation of a solid object (that is represented as a triangle mesh) by a bounding set of spheres to minimize the sum of the volume of the sphere and, subsequently, the approximation of the object' volume was discussed in [Wan06a]. A variational approach to computing an optimal segmentation of a 3D shape for estimating a union of tight bounding volumes was described in [Lu07a].

In this paper, an approach for calculating the volume of three-dimensional objects described by a set of Bézier triangles is proposed. 3D models of biomedical origin are considered as the object of study, but the proposed method can be used on any model that is described by Bézier triangles.

2. BÉZIER TRIANGLES AS ELEMENTS OF 3D MODELING

One of the topical problems, when analysing an object, is its volume estimation. In this paper, a method is proposed that can be used to calculate the volume of an object by means of integral calculations. Input data for this method is a 3D model that is described by a set of Bézier triangles. An example of constructing such 3D models, based on biomedical objects can be seen in [Sis12a].

A Bézier triangle is described as follows:

$$S(u,v,w) = \\ = \sum_{\substack{i,j,k=0 \\ i+j+k=n}}^{n} \binom{n}{i,j,k} \cdot u^i \cdot v^j \cdot w^k \cdot p_{i,j,k}\,, \quad (1)$$

where: i,j,k – indexes of sum;

u,v,w – parameters;

$p_{i,j,k}$ – control points

$\binom{n}{i,j,k} = \frac{n!}{i! \cdot j! \cdot k!}$ - trinomial coefficients.

To calculate the volume, the Bézier triangle is transformed as follows:

$$S(u,v) = \sum_{\substack{i,j,k=0 \\ i+j+k=n}}^{n} T_{i,j,k}(u,v) \cdot p_{i,j,k}\,, \quad (2)$$

where:

$$T_{i,j,k}(u,v) = \frac{n!}{i! \cdot j! \cdot k!} \cdot u^i \cdot v^j \cdot (1-u-v)^k\,. \quad (3)$$

In this case partial derivatives of base functions could be described as:

$$\frac{\partial T_{i,j,k}(u,v)}{\partial u} = \frac{n!}{i! \cdot j! \cdot k!} \cdot u^{i-1} \cdot v^j \cdot \\ \cdot (1-u-v)^k \cdot (i \cdot (1-u-v) - u \cdot k)\,, \quad (4)$$

and

$$\frac{\partial T_{i,j,k}(u,v)}{\partial v} = \frac{n!}{i! \cdot j! \cdot k!} \cdot u^i \cdot v^{j-1} \cdot \\ \cdot (1-u-v)^k \cdot (j \cdot (1-u-v) - v \cdot k) \quad (5)$$

3. THE PROPOSED METHOD OF VOLUME ESTIMATION

Let's consider that the volume of an object is a sum of volumes of curvilinear prisms:

$$V_{obj} = \sum V_{prism}\,. \quad (6)$$

where: V_{obj} – volume of an object;

V_{prism} – volume of a curvilinear prism.

The method of estimating the volume of curvilinear prisms is described in Theorem 1.

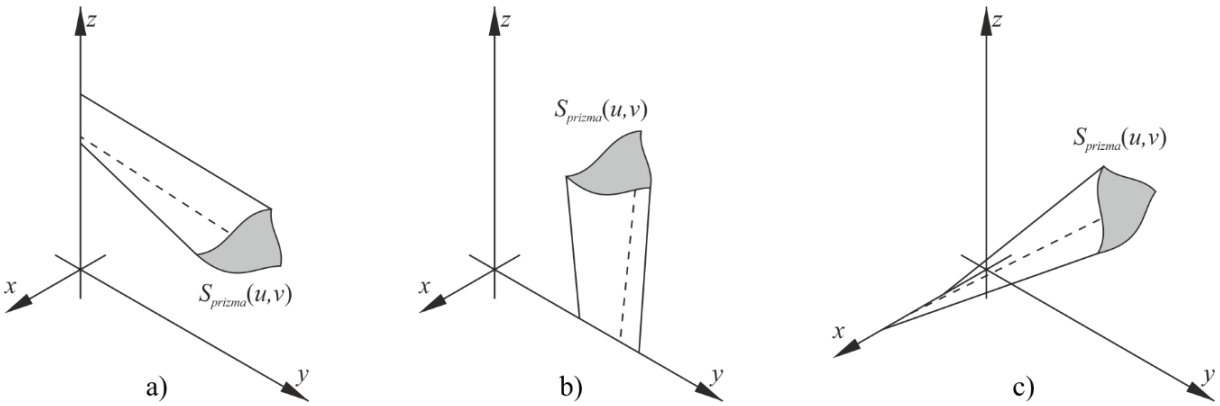


Figure 1. Examples of a curvilinear prism.

Theorem 1

If a curvilinear prism was constructed by projecting the vertices of a Bézier triangle onto coordinate axis, its volume could be calculated as follows:

$$V_{prism} = \frac{(n!)^3}{2} \cdot \cdot \sum_{\substack{i1,j1,k1=0 \\ i1+j1+k1=n}}^{n} \sum_{\substack{i2,j2,k2=0 \\ i2+j2+k2=n}}^{n} \sum_{\substack{i3,j3,k3=0 \\ i2+j2+k2=n}}^{n} Q \,, \quad (7)$$

where:

$$Q = \frac{\det(M_Z) \cdot I}{i1! \cdot j1! \cdot k1! \cdot i2! \cdot j2! \cdot k2! \cdot i3! \cdot j3! \cdot k3!}, \quad (8)$$

When projecting onto the *Oz* axis (Fig.1a.), the matrix takes the following form:

$$M_Z = \begin{bmatrix} x_{i1,j1,k1} & y_{i1,j1,k1} & 0 \\ x_{i2,j2,k2} & y_{i2,j2,k2} & z_{i2,j2,k2} \\ x_{i3,j3,k3} & y_{i3,j3,k3} & z_{i3,j3,k3} \end{bmatrix}. \quad (9)$$

Accordingly, when projecting onto the *Oy* axis (Fig.2b.):

$$M_Y = \begin{bmatrix} x_{i1,j1,k1} & 0 & z_{i1,j1,k1} \\ x_{i2,j2,k2} & y_{i2,j2,k2} & z_{i2,j2,k2} \\ x_{i3,j3,k3} & y_{i3,j3,k3} & z_{i3,j3,k3} \end{bmatrix}, \quad (10)$$

And, accordingly. on the *Ox* axis (see Fig.2c.):

$$M_X = \begin{bmatrix} 0 & y_{i1,j1,k1} & z_{i1,j1,k1} \\ x_{i2,j2,k2} & y_{i2,j2,k2} & z_{i2,j2,k2} \\ x_{i3,j3,k3} & y_{i3,j3,k3} & z_{i3,j3,k3} \end{bmatrix}. \quad (11)$$

A similar method was described in [Sis13a], except it is used for rational Bezier surfaces. The approach that is proposed in Theorem 1 is a logical continuation of ideas described in [Sis09a] and [Sis13a] and is used in this work for the case of Bezier triangle.

The method of calculating the integral *I* in symbol form, is described in Theorem 2.

Theorem 2

Beta function is used to calculate the values of integral I. Historical overview of Beta function with detailed description of its mathematical properties can be found in [Mas09a]. If the argument values of the Beta function are natural numbers ($a, b \in N$), the Beta function could be described as follows:

$$B(a;b) = \frac{(a-1)! \cdot (b-1)!}{(a+b-1)!}. \quad (12)$$

The method of calculating the value of integral *I* depends on the values of coefficients that are calculated as follows:

$$\begin{aligned} i4 &= i1 + i2 + i3 \\ j4 &= j1 + j2 + j3, \\ k4 &= k1 + k2 + k3 \end{aligned} \quad (13)$$

The integral *I* is calculated using the formulas described below.

1) for case ($i4 > 0;\ j4 > 0;\ k4 > 1$):

$$\begin{aligned} I = {} & C_1 \cdot D_1 \cdot B(i4;\ j4 + k4 + 1) + \\ & + C_2 \cdot D_2 \cdot B(i4 + 1;\ j4 + k4) + \\ & + C_3 \cdot D_2 \cdot B(i4;\ j4 + k4) + \\ & + C_4 \cdot D_3 \cdot B(i4 + 2;\ j4 + k4 - 1) + \\ & + C_5 \cdot D_3 \cdot B(i4 + 1;\ j4 + k4 - 1) + \\ & + C_6 \cdot D_3 \cdot B(i4;\ j4 + k4 - 1) \end{aligned}, \quad (14)$$

where:

$$\begin{aligned} C_1 &= i2 \cdot k3 + i2 \cdot j3 \\ C_2 &= k2 \cdot k3 + i2 \cdot k3 + j3 \cdot k2 + \\ &+ 2 \cdot i2 \cdot j3 \\ C_3 &= -i2 \cdot k3 - 2 \cdot i2 \cdot j3 \\ C_4 &= j3 \cdot k2 + i2 \cdot j3 \\ C_5 &= -j3 \cdot k2 - 2 \cdot i2 \cdot j3 \\ C_6 &= i2 \cdot j3 \end{aligned}, \quad (15)$$

and

$$\begin{aligned} D_1 &= \sum_{q=0}^{k4-2} \binom{k4-2}{q} \cdot \frac{(-1)^q}{j4+q+2}, \\ D_2 &= \sum_{q=0}^{k4-2} \binom{k4-2}{q} \cdot \frac{(-1)^q}{j4+q+1}, \\ D_3 &= \sum_{q=0}^{k4-2} \binom{k4-2}{q} \cdot \frac{(-1)^q}{j4+q}. \end{aligned} \quad (16)$$

2) for case ($i4 = 0;\ j4 > 0;\ k4 > 1$):

$$I = \frac{k2 \cdot k3 \cdot D_2 - j3 \cdot k2 \cdot D_4}{j4 + k4}, \quad (17)$$

where:

$$D_4 = \sum_{q=0}^{k4-1} \binom{k4-1}{q} \cdot \frac{(-1)^q}{j4+q}. \quad (18)$$

3) for case ($i4 > 0$; $j4 = 0$; $k4 > 1$):

$$I = \frac{k2 \cdot k3}{k4-1} \cdot B(i4+1; k4) - \frac{i2 \cdot k3}{k4} \cdot B(i4; k4+1). \quad (19)$$

4) for case ($i4 > 0$; $j4 > 0$; $k4 = 0$):

$$I = \frac{i2 \cdot j3}{j4} \cdot B(i4; j4+1). \quad (20)$$

5) for case ($i4 > 0$; $j4 > 0$; $k4 = 1$):

$$I = C_7 \cdot B(i4, j4+2) - \frac{j3 \cdot k2}{j4} \cdot B(i4+1, j4+1)', \quad (21)$$

where

$$C_7 = \frac{i2 \cdot (j3 - j4 \cdot k3)}{j4 \cdot (j4+1)}. \quad (22)$$

6) for case ($i4 = 3n-1$; $j4 = 0$; $k4 = 1$):

$$I = -\frac{i2 \cdot k3}{3n \cdot (3n-1)}. \quad (23)$$

7) for case ($i4 = 0$; $j4 = 3n-1$; $k4 = 1$):

$$I = -\frac{j3 \cdot k2}{3n \cdot (3n-1)}. \quad (24)$$

8) for cases ($i4 = 3n$; $j4 = 0$; $k4 = 0$), ($i4 = 0$; $j4 = 3n$; $k4 = 0$) and ($i4 = 0$; $j4 = 0$; $k4 = 3n$):

$$I = 0. \quad (25)$$

Proof of Theorem 1

To find the differential volume of a prism dV_{prism}, as seen on Fig.2, in the case where vector $\vec{r}$ is perpendicular to $dS*$, the differential volume could be calculated using the following formula:

$$dV_{prism} = \frac{1}{2} \cdot |\vec{r}| \cdot dS*, \quad (26)$$

where: $\vec{r}$ – perpendicular vector from *Oz* axis to a point on a surface.

Considering that infinitely small values are used in the calculations, it can be assumed that the following expression is valid:

$$dS* = \cos\alpha \cdot dS_{prism}, \quad (27)$$

where: α – the angle between the vector $\vec{r}$ and the normal vector $\vec{n}$ on dS_{prism}.

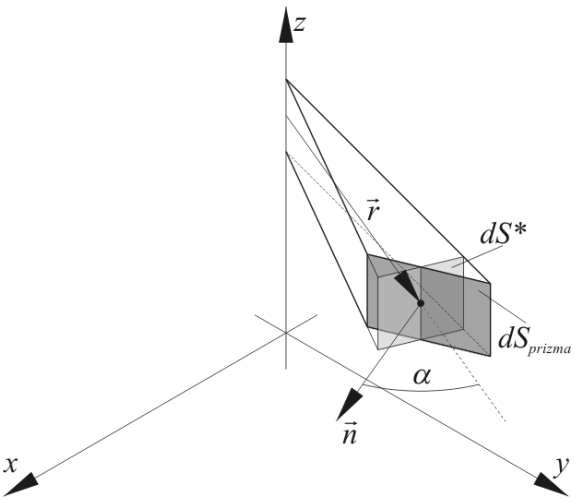


Figure 2. Differential volume calculation.

We calculate $\cos\alpha$ value using the following formula:

$$\cos\alpha = \frac{\vec{r} \cdot \vec{n}}{|\vec{r}| \cdot |\vec{n}|} = \frac{r_X \cdot n_X + r_Y \cdot n_Y + r_Z \cdot n_Z}{|\vec{r}| \cdot |\vec{n}|}, \quad (28)$$

differential equation can be obtained from (26), (27) and (28):

$$dV_{prism} = \frac{1}{2} \cdot \frac{\vec{r} \cdot \vec{n}}{|\vec{n}|} \cdot dS_{prism}, \quad (29)$$

To solve this differential equation, both sides of (29) are integrated with a surface integral:

$$V_{prism} = \frac{1}{2} \cdot \iint_{(S_{prism})} \frac{\vec{r} \cdot \vec{n}}{|\vec{n}|} dS_{prism}, \quad (30)$$

In order to solve the surface integral of scalar fields (30), it is necessary to convert into a double integral. The following values also need to be considered:

$$A = \begin{vmatrix} \frac{\partial S_Y(u,v)}{\partial u} & \frac{\partial S_Z(u,v)}{\partial u} \\ \frac{\partial S_Y(u,v)}{\partial v} & \frac{\partial S_Z(u,v)}{\partial v} \end{vmatrix}, \quad (31)$$

and

$$B = \begin{vmatrix} \frac{\partial S_Z(u,v)}{\partial u} & \frac{\partial S_X(u,v)}{\partial u} \\ \frac{\partial S_Z(u,v)}{\partial v} & \frac{\partial S_X(u,v)}{\partial v} \end{vmatrix}, \quad (32)$$

and

$$C = \begin{vmatrix} \frac{\partial S_X(u,v)}{\partial u} & \frac{\partial S_Y(u,v)}{\partial u} \\ \frac{\partial S_X(u,v)}{\partial v} & \frac{\partial S_Y(u,v)}{\partial v} \end{vmatrix}, \quad (33)$$

where: $S_X(u,v), S_Y(u,v), S_Z(u,v)$ are coordinates of function (1).

Considering the *A*, *B* and *C* values it can be concluded that the normal vector could be described as follows:

$$\vec{n} = (A \quad B \quad C), \quad (34)$$

In this case the vector $\vec{n}$ module can be calculated as follows:

$$|\vec{n}| = \sqrt{A^2 + B^2 + C^2}, \quad (35)$$

By placing (35) in (30) and transforming (30) to a double integral (using the transition properties from surface integral of the first kind to double integral), we get:

$$V_{prism} = \frac{1}{2} \cdot \int_{v\_\min}^{v\_\max} \int_{u\_\min}^{u\_\max} (r_X \cdot n_X + r_Y \cdot n_Y + r_Z \cdot n_Z) du dv, \quad (36)$$

Considering the nature of vector $\vec{r}$ it could be described as follows:

$$\vec{r} = (S_X(u,v) \quad S_Y(u,v) \quad 0), \quad (37)$$

Considering the range of parameters *u* and *v*, the final integral of volume estimation of a curvilinear prism could be described as follows:

$$V_{prism} = \frac{1}{2} \cdot \int_0^1 \left\{ \int_0^{1-u} f_1(u,v) dv \right\} du, \quad (38)$$

where:

$$f_1(u,v) = A \cdot S_X(u,v) + B \cdot S_Y(u,v), \quad (392)$$

By substituting the values (2) into (39) and after transforming we get:

$$f_1(u,v) = \sum_{\substack{i1,j1,k1=0 \\ i1+j1+k1=n}}^{n} \sum_{\substack{i2,j2,k2=0 \\ i2+j2+k2=n}}^{n} \sum_{\substack{i3,j3,k3=0 \\ i2+j2+k2=n}}^{n} \{ \det(M_Z) \cdot T_{i1,j1,k1}(u,v) \cdot \frac{\partial T_{i2,j2,k2}(u,v)}{\partial u} \cdot \frac{\partial T_{i3,j3,k3}(u,v)}{\partial v} \}, \quad (40)$$

where M_Z is described in (9)

It is easy to see that if the curvilinear prism is constructed by projecting the points of Bézier triangle onto the coordinate axis *Oy* or *Ox* (Fig.6b and Fig.6c.), the matrices in formula (9) take the form (10) and (11).

By placing (3)-(5) into (39) and after transforming the volume estimation integral could be described as follows:

$$V_{prism} = \frac{(n!)^3}{2} \cdot \sum_{\substack{i1,j1,k1=0 \\ i1+j1+k1=n}}^{n} \sum_{\substack{i2,j2,k2=0 \\ i2+j2+k2=n}}^{n} \sum_{\substack{i3,j3,k3=0 \\ i2+j2+k2=n}}^{n} G_{prism}, \quad (41)$$

where:

$$G_{prism} = \frac{\det(M_Z) \cdot I}{i1!\, j1!\, k1!\, i2!\, j2!\, k2!\, i3!\, j3!\, k3!}, \quad (42)$$

where:

$$I = \int_0^1 u^{i4-1} \cdot I_u du, \quad (43)$$

where:

$$I_u = \int_0^{1-u} v^{j4-1} \cdot (1-u-v)^{k4-2} \cdot (i2 \cdot (1-u-v) - u \cdot k2) \cdot (j3 \cdot (1-u-v) - v \cdot k3) dv. \quad (44)$$

Solving the integral (44) is described in the Proof of Theorem 2.

Proof of Theorem 2.

1) for case ($i4 > 0$; $j4 > 0$; $k4 > 1$) the integral (44) after simplifying and transforming takes the following form:

$$I_u = C_1 \cdot \int_0^{1-u} v^{j4+1} \cdot (1-u-v)^{k4-2}\,dv +$$
$$+ C_2 \cdot u \cdot \int_0^{1-u} v^{j4} \cdot (1-u-v)^{k4-2}\,dv +$$
$$+ C_3 \cdot \int_0^{1-u} v^{j4} \cdot (1-u-v)^{k4-2}\,dv + \quad , \qquad (45)$$
$$\{C_4 \cdot u^2 + C_5 \cdot u + C_6\} \cdot$$
$$\int_0^{1-u} v^{j4-1} \cdot (1-u-v)^{k4-2}\,dv$$

where the $C_1 - C_6$ values are described in (15). The separate integral (45) is solved using binomial theorem:

$$\int_0^{1-u} v^{j4+1} \cdot (1-u-v)^{k4-2}\,dv =$$
$$= \sum_{q=0}^{k4-2} (-1)^q \cdot \binom{k4-2}{q} \cdot (1-u)^{k4-2-q} \cdot \qquad (46)$$
$$\cdot \int_0^{1-u} v^{j4+q+1} dv =$$
$$= \sum_{q=0}^{k4-2} (-1)^q \cdot \binom{k4-2}{q} \cdot \frac{(1-u)^{k4+j4}}{j4+q+2}$$

Similarly, the other integrals in (45) can be solved. Substituting the results in (43), after transformations, we get:

$$I_b = C_1 \cdot D_1 \cdot \int_0^1 u^{i4-1} \cdot (1-u)^{k4+j4}\,du +$$
$$+ C_2 \cdot D_2 \cdot \int_0^1 u^{i4} \cdot (1-u)^{j4+k4-1}\,du +$$
$$+ C_3 \cdot D_2 \cdot \int_0^1 u^{i4-1} \cdot (1-u)^{j4+k4-1}\,du +$$
$$+ C_4 \cdot D_3 \cdot \int_0^1 u^{i4+1} \cdot (1-u)^{j4+k4-2}\,du + \qquad (47)$$
$$+ C_5 \cdot D_3 \cdot \int_0^1 u^{i4} \cdot (1-u)^{j4+k4-2}\,du +$$
$$C_6 \cdot D_3 \cdot \int_0^1 u^{i4-1} \cdot (1-u)^{j4+k4-2}\,du$$

where D_1 - D_3 are described in (16). Some of the integrals in (47) are Euler's integrals of the first kind. They can be solved with a Beta function, described in (12). The solution of (47) is (14).

2) for case ($i4 = 0$; $j4 > 0$; $k4 > 1$) integral (44) takes the following form:

$$I_u = (-u \cdot k2) \cdot$$
$$\cdot \left[j3 \int_0^{1-u} v^{j4-1} \cdot (1-u-v)^{k4-1}\,dv - \right. \qquad (48)$$
$$\left. - k3 \int_0^{1-u} v^{j4} \cdot (1-u-v)^{k4-2}\,dv \right]$$

Some of the integrals in (48) can be solved similarly as in (46). The result of integrating (48) is as follows:

$$I_u = u \cdot (1-u)^{j4+k4-1} \cdot$$
$$\cdot [k2 \cdot k3 \cdot D_2 - j3 \cdot k2 \cdot D_4], \qquad (49)$$

Where D_4 is described in (18). By placing (49) into (43) after integration we get (17).

3) for case ($i4 > 0$; $j4 = 0$; $k4 > 1$) integral (44) takes the following form:

$$I_u = u \cdot k2 \cdot k3 \cdot \int_0^{1-u} (1-u-v)^{k4-2}\,dv -$$
$$- i2 \cdot k3 \cdot \int_0^{1-u} (1-u-v)^{k4-1}\,dv = \qquad (50)$$
$$= \frac{k2 \cdot k3}{k4-1} \cdot \{u \cdot (1-u)^{k4-1}\} -$$
$$- \frac{i2 \cdot k3}{k4} \cdot \{(1-u)^{k4}\}$$

By placing (50) into (43) after integration we get (19).

4) for case ($i4 > 0$; $j4 > 0$; $k4 = 0$) integral (44) takes the following form:

$$I_u = i2 \cdot j3 \int_0^{1-u} v^{j4-1} dv = \frac{i2 \cdot j3}{j4} (1-u)^{j4}. \qquad (51)$$

By placing (51) into (43) after integration we get (20).

5) for case ($i4 > 0$; $j4 > 0$; $k4 = 1$) the value of (44) is calculated for three sub cases:

a) for case $k1 = 1$ and $k2 = k3 = 0$:

$$I_u = i2 \cdot j3 \cdot \int_0^{1-u} v^{j4-1} \cdot (1-u-v)\,dv \qquad (52)$$

b) for case $k1 = 0$, $k2 = 1$ and $k3 = 0$:

$$I_u = i2 \cdot j3 \cdot \int_0^{1-u} v^{j4-1} \cdot (1-u-v)dv - \\ - j3 \cdot u \cdot \int_0^{1-u} v^{j4-1}dv \quad (53)$$

c) for case $k1=0$, $k2=0$ and $k3=1$:

$$I_u = i2 \cdot j3 \cdot \int_0^{1-u} v^{j4-1} \cdot (1-u-v)dv - \\ - i2 \cdot \int_0^{1-u} v^{j4}dv \quad (54)$$

Combining (52)-(54) after transformations we get:

$$I_u = i2 \cdot j3 \cdot (1-u) \cdot \int_0^{1-u} v^{j4-1}dv - \\ - j3 \cdot k2 \cdot u \cdot \int_0^{1-u} v^{j4-1}dv - \\ (i2 \cdot j3 + i2 \cdot k3) \cdot \int_0^{1-u} v^{j4}dv, \quad (55)$$

After substituting the integration results (55) into (43) we get (21).

6) for case ($i4=3n-1$; $j4=0$; $k4=1$) the value of (44) is calculated for three sub cases:

a) for cases ($k1=1$ and $k2=k3=0$) and ($k1=0$, $k2=1$ and $k3=0$):

$$I_u = 0; \quad (56)$$

b) for case $k1=0$, $k2=0$ and $k3=1$:

$$I_c = -i2 \cdot \int_0^{1-u} dv = -i2 \cdot (1-u). \quad (57)$$

Combining (56) and (57) after transformations we get:

$$I_u = -i2 \cdot k3 \cdot (1-u). \quad (58)$$

After substituting the integration results (58) into (43) we get (23).

7) for case ($i4=0$; $j4=3n-1$; $k4=1$) the value of (44) is calculated for three sub cases:

a) for cases ($k1=1$ and $k2=k3=0$) and ($k1=0$, $k2=0$ and $k3=1$):

$$I_u = 0; \quad (59)$$

b) for case $k1=0$, $k2=1$ and $k3=0$

$$I_u = -j3 \cdot u \cdot \int_0^{1-u} v^{3n-2}dv; \quad (60)$$

Combining (59) and (60) after transformations we get:

$$I_c = -j3 \cdot k2 \cdot u \cdot \int_0^{1-u} v^{3n-2}dv = \\ = u \cdot (1-u)^{3n-1} \cdot \left[\frac{-j3 \cdot k2}{3n-1}\right]. \quad (61)$$

After substituting the integration results (61) into (43) we get (24).

8) for cases ($i4=3n$; $j4=0$; $k4=0$), ($i4=0$; $j4=3n$; $k4=0$) and ($i4=0$; $j4=0$; $k4=3n$):

It is necessary to consider that in the described cases the value of the determinant from the formula (8) takes the values:

$$\det(M_Z) = \det(M_Y) = \det(M_X) = 0. \quad (62)$$

In this case the value of integral *I* does not affect the value of *Q* (from (8)):

$$\det(M_Z) \cdot I = 0 \Rightarrow \forall I \in R. \quad (63)$$

For the convenience of software implementation of the method, the integral *I* can take the zero value as in (25).

An illustration of cases with different values of *i*4, *j*4 and *k*4 can be seen in Fig.4 in Appendix.

Numerical integration

To check the accuracy of symbolic integration, the numerical integration was also used to calculate the approximate value of integral *I*.

The range of parameters *u* and *v* is divided into *m* parts. In this case, the volume of the curvilinear prism could be approximated as follows:

$$I_b \approx \frac{1}{m^2} \cdot \left\{ \left(\sum_{q1=0}^{(m-1)} \sum_{q2=0}^{(m-2-q1)} f_2(u_{q1}, v_{q2}) \right) + \right. \\ \left. + \frac{1}{2} \left(\sum_{q1=0}^{(m-1)} f_b(u_{q1}, v_{m-2-q1}) \right) \right\}, \quad (64)$$

where:

$$u_{q1} = \frac{2 \cdot q1 + 1}{2 \cdot m}, \quad (65)$$

and

$$v_{q2} = \frac{2 \cdot q2 + 1}{2 \cdot m}, \quad (66)$$

and

$$f_2(u,v) = u^{i4-1} \cdot v^{j4-1} \cdot (1-u-v)^{k4-2} \cdot \\ \cdot (i2 \cdot (1-u-v) - u \cdot k2) \cdot \\ \cdot (j3 \cdot (1-u-v) - v \cdot k3) \quad (67)$$

4. EXPERIMENTS

To test the accuracy of the proposed method, the volume was estimated for a 3D model of an antiprism (Fig.3a.), with an approximately known volume in a range from 9351487,18 to 9732392,24.

The proposed volume estimation method was also used to calculate the volume of 3D models of real biological objects. Three models were acquired through 3D scanning (foot, hand, human torso), and two models of the head, that were automatically generated through processing Computed Tomography images (Fig 3e.) and regular photographs (Fig. 3f.), these objects can be seen on Figure 3. A method, described in [Sis12a], was used to create spline models of biomedical objects.

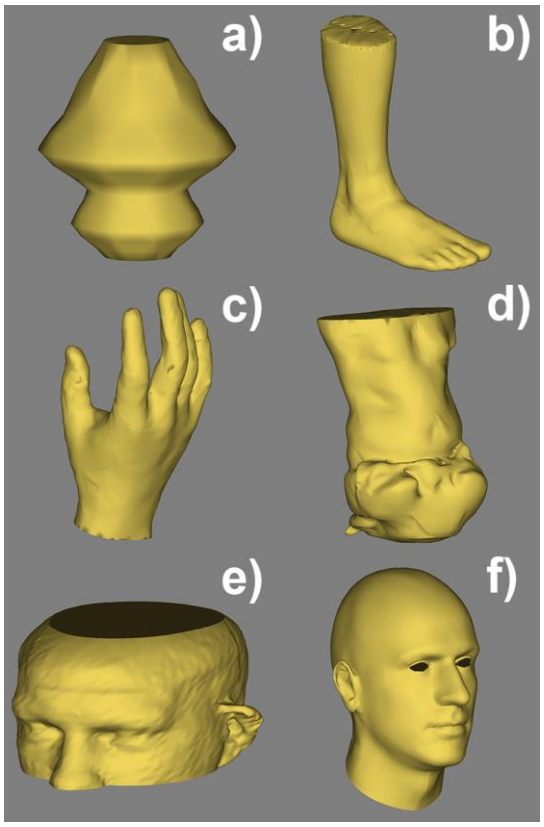


Figure 3. 3D models of scanned biological objects, a) Antiprism, b) scanned foot, c) scanned hand, d) scanned torso, e) head (from CT), f) head (from photographs)

The volume was estimated using both the symbolic and numerical integration (m=10). The error of numerical integration was also calculated. The results can be seen in Table 1.

Object	Numerical	Symbolic	Error, %
Antiprism	9632649,96	9614998,57	0,1832
foot	3391616,27	3391342,52	0,0081
hand	374145,34	374030,57	0,0307
body	54876035,86	54870496,04	0,0101
head (CT)	21059419,31	21055516,85	0,0185
Head (photo)	7951163,16	7949147,47	0,0254

Table 1. Volume estimation of biological objects

The results of the experiments show that the estimated volume of the antiprism falls within the known volume range (9351487,18 to 9732392,24). The proposed algorithm has also managed to estimate volume of all the other 3D models.

The error of the numerical integration was very low (below 0,2%). The highest error appeared when estimating the volume of the antiprism, due to the size of the triangles relative to full 3D model size. The size of the antiprism triangles was considerably greater compared to the other models.

To evaluate the proposed method, it is also necessary to consider the time of calculation of volumes, as well as the models. The calculation times were measured on a computer with the following specifications: Processor - Intel Core i7-3770K 3.50GHz, RAM - 8GB, OS - Windows 7 64bit. The acquired calculation times can be seen in Table2.

Object	Numerical, s.	Symbolic, s.	Number of Bezier triangles
Antiprism	0,2646	0,0204	92
Foot	5,7684	0,4435	1996
Hand	5,7181	0,4395	1979
body	5,7910	0,4455	1996
Head (CT)	43,3005	3,3389	14695
Head (photo)	8,0820	0,6228	2794

Table 2. Calculation times of volume estimation

As can be seen in Table 2 the calculation time of volume estimation using symbolic integration is 13 times faster than using the numerical integration. In addition, when using symbolic integration, an

average of 4481 Bezier triangles are processed in 1 second. However, when numerical integration is used, an average of 345 Bezier triangles are processed in 1 second.

5. CONCLUSIONS

In this paper, a volume estimation method was proposed, for objects described by a set of Bézier triangles. This proposed approach is based on integral calculations methods: Surface integrals of scalar fields, Euler's integral of the first kind and Beta function for natural arguments.

The proposed method was tested on several models: an etalon object and five biomedical objects that were generated in different ways. Two volume estimation approaches were used in the experiments: symbolic integration and numerical integration where the value of the parameter *m* was 10. The results of the experiments can be seen in Table 1. The experiments have proved the accuracy of the proposed volume estimation method. Based on the results of the experiments it is possible to conclude that using the numerical integration the error was bellow 0,2%.

The proposed method could be used to estimate volume of objects that are described by spline surfaces (in our case - Bezier triangles). If the model of the object is described using different kinds of patches (for example, triangular + rectangular), then the proposed method will only estimate a part of the object's volume. If the object is more precisely described with a polygonal mesh, then the volume can be estimated using the proposed method (in particular cases) or using the approach from [Zha01a].

Further research on this subject would be related to optimizing the volume estimation functions to decrease the number of necessary calculations and decrease the calculation time.

6. ACKNOWLEDGEMENT

The research was supported by the National Research Programme "The next generation of information and communication technologies" (NexIT, Project 2).

7. REFERENCES

[Baj96a] Bajaj, C., Cole, E. & Lin, K. Boundary and 3D triangular meshes from planar cross sections. Proceedings of the 5th International Imaging Roundtable, Sandia National Laboratories, Sandia Report SAND 96–2301, VC-405, Pittsburgh, PA, USA. pp. 169–178, 1996.

[Bres91a] Bresnahan, J.C., Beattie, M.S., Stokes, B.T. & Conway, K.M. (1991) Three dimensional computer-assisted analysis of graded contusion lesions in the spinal cord of the rat. J. Neurotrauma, 8, 91–97, 1991.

[Col11a] Colley, S.J. Vector Calculus (4th Edition), 2011, 624 p.

[Duer00b] Duerstock, B.S., Bajaj, C.L., Pascucci, V., Shikore, D., Lin, K. & Borgens, R.B. (2000) Advances in three-dimensional reconstruction of the experimental spinal cord injury. Comput. Med. Imag. Grap., 24, 389–406, 2000.

[Duer03a] Duerstock, B.S., Bajaj C. L., A comparative study of the quantitative accuracy of three-dimensional reconstructions of spinal cord from serial histological sections, Journal of Microscopy, Vol. 2010, Pt 2, pp. 138-148, 2003.

[How93a] Howard, C.V., et al. Measurement of total neuronal volume, surface area and dendrite length following intracellular physiological recording. Neuroprotocols. 2, 113–120, 1993.

[Juh00a] Juhász, I. Computing Volume of Solids Bounded by Bézier Surfaces. Mathematical Notes, Miskolc, Vol. 1., No. 2., 2000, pp. 127—133.

[Lu07a] Lu, L., Choi, Y.-K., Wang, W., Kim, M.-S. Variational 3D Shape Segmentation for Bounding Volume Computation. Journal Computer Graphics Forum, Volume 26, Issue 3, 2007, pp. 329–338.

[Mas09a] Massa Esteve, M. R., and Delshams, A. Euler's beta integral in Pietro mengoli's works. Archive for History of Exact Sciences, 2009, Volume 63, Issue 3, pp 325–356.

[Matt87a] Mattfeldt, T. Volume estimation of biological objects by systemic sampling. J. Math. Biol. 25, 685–695, 1987.

[McNul00a] McNulty, V., Cruz-Orive, L.M., Roberts, N., Holmes, C.J. & Gual-Arnau, X. Estimation of brain compartment Volume from MR Cavalieri slices. J. Comput. Assist. Tomogr. 24, 466–477, 2000.

[Mich88a] Michel, R.P. & Cruz-Orive, L.M. Application of the Cavalieri principle and vertical sections method to lung: estimation of Volume and pleural surface area. J. Microsc. 150, 117–136, 1988.

[Mor98a] Moriarty, L.J., Duerstock, B.S., Bajaj, C.L., Lin. K. & Borgens, R.B. Two and three dimensional computer graphic evaluation of the subacute spinal cord injury. J. Neurol. Sci. 155, 121–137.

[Ras05a] Rasmussen, A.F. and Floater M.S A point-based method for estimating surface area. Electronic Proceedings of the SIAM conference on Geometric Design, Phoenix, 2005.

[Sis09a] Sisojevs, A., Boločko, K. and Glazs, A. 3D Modeling of Free-Form Object (Interpolation, Visualization and Volume Estimation). The 17th International Conference WSCG'2009: Communication Papers Proceedings, 2009, pp. 125.-128.

[Sis12a] Sisojevs A., Kovalovs M. and Glazs A. Medical Object 3D Visualization Method Based on the Bézier Triangles. IADIS International Conference CGVCVIP'2012 proceedings. Lisbon, 2012, pp. 185 – 187.

[Sis13a] Sisojevs, A. An Approach of Semi-numerical Computing Volume of Solids Bounded by Rational Bézier Surfaces. Scientific Journal of RTU: Technologies of Computer Control, Vol.14, 2013, pp. 25.-31.

[Sun13a] Sun, F., Choi, Y.-K., Yu, Y., Wang, W. Medial meshes for volume approximation. http://dblp.uni-trier.de/db/journals/corr/corr1308.html#SunCYW13

[Wan06a] Wang, R., Zhou, K., Snyder, J., Liu, X., Bao, H., Peng, Q., Guo, B. Variational Sphere Set Approximation for Solid Objects. Journal Visual Computer, 2006, Vol. 22, Issue 9, pp. 612-621.

[Wro06a] Wronecki, J., Concept Modeling with NURBS, Polygon and Subdivision Surfaces. Proceedings of the 2006 American Society for Engineering Education Annual Conference & Exposition

[Zha01a] Zhang, C., and Chen, T. Efficient Feature Extraction for 2D/3D Objects in Mesh Representation. Proceedings of IEEE International Conference on Image Processing ICIP, 2001, pp.935-938.

8. APPENDIX

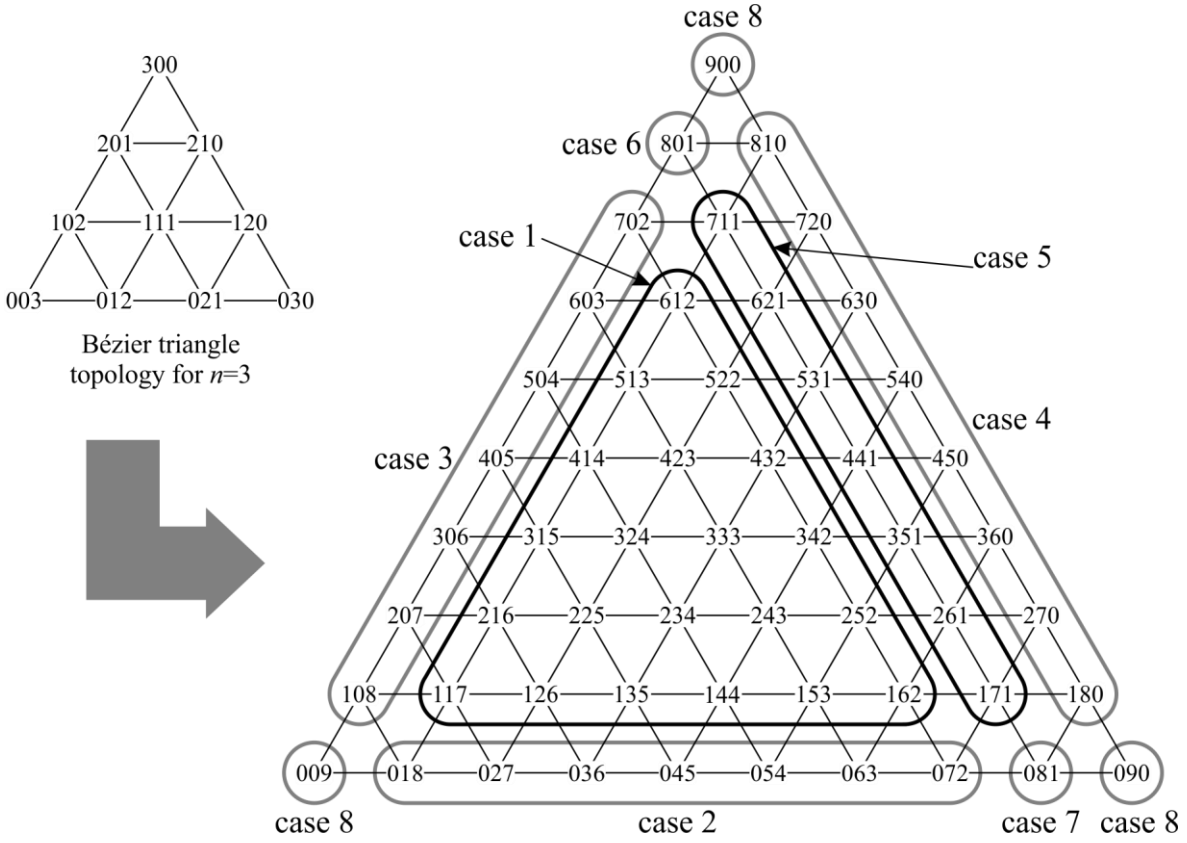


Figure 4. Possible values of *i*4, *j*4, *k*4 when *n*=3

CUDA-based SeqSLAM for Real-Time Place Recognition

Safa Ouerghi
Carthage Univ,
Sup'Com, GRESCOM,
2083 El Ghazela,
Tunisia
safa.ouerghi@supcom.tn

Remi Boutteau
Normandie Univ,
UNIROUEN
ESIGELEC, IRSEEM
76000 Rouen, France
Remi.Boutteau@esigelec.fr

Fethi Tlili
Carthage Univ,
Sup'Com, GRESCOM,
2083 El Ghazela,
Tunisia
fethi.tlili@supcom.tn

Xavier Savatier
Normandie Univ,
UNIROUEN
ESIGELEC, IRSEEM
76000 Rouen, France
Xavier.Savatier@esigelec.fr

ABSTRACT

Vehicle localization is a fundamental issue in autonomous navigation that has been extensively studied by the Robotics community. An important paradigm for vehicle localization is based on visual place recognition which relies on learning a database, then consecutively trying to find matchings between this database and the actual visual input. An increasing interest has been directed to visual place recognition in varying conditions like day and night cycles and seasonal changes. A major approach dealing with such challenges is Sequence SLAM (SeqSLAM) based on matching a sequence of images to the database instead of a single image. This algorithm allows global pose recovery at the expense of a higher computational time. To solve this problem with a certain amount of speedup, we propose in this work, a CUDA-based solution for real-time place recognition with SeqSLAM. We design a mapping of SeqSLAM to CUDA architecture and we describe, in detail, our hardware-specific implementation considerations as well as the parallelization methods. Performance analysis against existing CPU implementation is also given, showing a speedup to six times faster than the CPU for common sized databases. More speedup could be obtained when dealing with bigger databases.

Keywords

Place recognition, global localization, SeqSLAM, robotics, CUDA, GPU.

1 INTRODUCTION

Visual place recognition systems have become increasingly important for a variety of mobile robotic platforms and applications. A typical setup of a visual recognition system contains a given database of images (the map) and the robot consecutively tries to find matchings between this database and the actual visual input (query images). The query is either a single image, which is most common, or an organized set of images *i.e.* a sequence. Recently, there has been an increasing focus on visual place recognition in varying conditions which promoted the design of vision-based systems that can work across common real-world perceptual changes such as varying weather conditions and day and night cycles [4, 16, 10, 11].

A well-known successful approach is SeqSLAM (Sequence SLAM) for visual place recognition introduced by Milford and Weyth in [10] and also described in [9]. This work puts forward the concept of camera-based GPS, which means performing global localization with ensuring to never lose again and aims at matching image sequences under strong seasonal changes. SeqSLAM is based on the computation of image-by-image dissimilarity scores between all query and database images, stored in a so-called difference matrix, then, computing a straight-line path through the full matching matrix to finally select the path with the smallest sum of dissimilarity scores. Despite the quite remarkable results achieved by SeqSLAM, building a full matching matrix introduces substantial computational complexity and enhancing it then computing scores for all straight lines presents an additional computational bottleneck.

However, many problems are being solved using programmable graphics hardware to achieve a certain amount of speedup and the usage of GPU computation has become a popular topic in the community. In fact, the Compute Unified Device Architecture (CUDA) has enabled graphics processors to be explicitly programmed as general purpose shared-memory multi-core processors with a high level of parallelism [6].

In this paper, we focus on an efficient parallelization of the state-of-the-art visual place recognition SeqSLAM to achieve a certain amout of speedup. We consider a single GPU implementation and we describe the strategies to efficiently map the problem components to CUDA architecture.

The paper proceeds as follows: Section II presents some related work. Section III summarises the SeqSLAM algorithm, and brievly reviews GPU architecture and performance considerations. Implementation details are, subsequently, presented in Section IV. Finally, Section IV, presents the results, discusses the outcomes of this paper and presents suggestions for future work.

2 RELATED WORK

Visual place recognition is a well-defined but very broad problem. State of the art techniques for visual place recognition based on features, fall generally into two categories: those that selectively extract parts of the image and those that use the whole image without a selection phase. Examples of the first category are local feature descriptors such as scale-invariant feature transforms (SIFT) [7] and speeded-up robust features (SURF) [1]. In the second category, we find global descriptors such as HOG [3] and GIST [13] that use to process the whole image regardless of its content. In fact, a considerable emphasis was placed on accelerating image descriptors for visual place recognition using CUDA GPUs that many problems are recently being solved with. In [20], Scale-invariant feature transform (SIFT) has been accelerated using CUDA GPU. Speeded-up robust features (SURF) has also been ported to CUDA to achieve a certain amount of speedup [19]. In addition, GIST and HOG global descriptors have been GPU accelerated in [18] and [15] respectively. We find, as well, another broad category of methods for visual place recognition based on a pure image retrieval. This process can be accelerated using inverted indices where image ID numbers are stored against words that appear in the image. FabMAP, a well known technique for visual place recognition along a 1000-km path [2] has relied on an inverted index with a bag-of-words model. Inverted index technique has been recently implemented on GPU demonstrating its efficiency [21]. The bag-of-words technique has also been GPU-implemented [17]. Furthermore, image retrieval systems can be enhanced by adding topological information as both FabMap and SeqSLAM do. Topological maps have in turn been improved by incorporating metric odometry information such as SMART [14] and CAT-SLAM [8]. The metric information within the topological place node can be stored as a sparse landmark map or as a dense occupancy grid map. However, dense spatial modeling has only become feasible in the past few years with the advent of GPU technology [12].

3 BACKGOUND

3.1 SeqSLAM

The SeqSLAM algorithm demonstrated impressive place recognition performance across significant condition variance such as seasonal changes and in case of low quality imagery (low resolution, low depth, and image blur). In order to increase the discriminative nature of the observation and to avoid the problem of false-positives, location is represented in SeqSLAM as a sequence of images, rather than a single image from one pose. Images are, beforehand, resolution-reduced and patch-normalised to enhance contrast. Then, SeqSLAM builds a difference matrix holding SAD (Sum of Absolute Differences) scores between all query and database image sequences. A key processing step is to normalize the image difference values within their (spatially) local image neighborhoods. Subsequently, a search for diagonals of low difference values is performed over the defined sequence length. However, SeqSLAM assumes similar speeds in repeated route traversals and negligible accelerations, limiting its performance in certain application contexts where these criteria are not met.

3.2 GPU architecture summary and performance considerations

NVIDIA GPUs comprise a set of SMs (Streaming Multiprocessors). Each SM consists of many SPs (Streaming Processors) and SFUs (Special Function Units), which are simple but energy-efficient processing units that execute instructions in a SIMD fashion (Single Instruction Multiple Data). The main memory of GPUs, called global memory, can be accessed from all the SPs in every SM. Furthermore, each SM has a set of registers and a small memory (shared memory). This shared memory is much faster than global memory. Each kernel is invoked by a set of threads that are grouped into thread blocks. The blocks are distributed by the hardware among the available SMs. Depending on the amount of required resources, each SM may be able to simultaneously execute several blocks. Each block has assigned an amount of shared memory that allows the exchanging of data among threads of the same block. A more detailed description of NVIDIA's GPU architecture can be found in [5]. The main important features influencing GPU performance are synchronization barriers, the trade-off between available threading and shared resources, coalescence issues in global memory access and shared memory bank conflicts.

4 CUDA-BASED SEQSLAM IMPLEMENTATION

In this section, we present the mapping strategies of SeqSLAM to the CUDA architecture. In total, we deal with three different kernels, each having a specific launch configuration that optimizes the use of the hardware. The first kernel is dedicated to the difference matrix computation, the second to contrast enhancement of the obtained difference matrix and the last one performs route searching in order to find the best match.

4.1 Difference Matrix Computation

The SeqSLAM algorithm uses Sum of Absolute Differences (SAD) to compare each query image I_q from the query sequence $\mathscr{Q} = (I_1, ..., I_Q)$ with $Q = |\mathscr{Q}|$ to each database image I_D from the database Sequence

$\mathscr{D} = (I_1, ..., I_D)$ with $D = |\mathscr{D}|$ and calculates the difference score:

$$d = \frac{1}{R_x R_y} |I_Q - I_D| \quad (1)$$

where d is the difference score and R_x and R_y are the horizontal and vertical image dimensions, respectively. The difference scores are assembled into the so-called difference matrix M of size $D \times Q$.

We assume that the database and query sequences reside in the GPU's global memory as unsigned char types. In fact, database images are cropped, converted to grayscale and resolution-reduced during the preprocessing step done offline and could therefore easily fit into GPU's global memory. On another note, it has been reported in [16, 10] that, the longer the query sequences are, the more difficult is to localize the precise match of a specific frame. This stems from the fact that although longer sequences are more distinguishable, the weight of each frame decreases with the sequence length. These considerations show that both the database and the query sequences could be stored in global memory, the first at the initialization of the GPU after an offline preprocessing, and the second, online acquired, is transferred from host to GPU after preprocessing as well.

Our parallelization strategy is based on a common strategy called tiling. This strategy consists in the partition of the data into subsets called tiles, such that each tile fits into the shared memory of a block. We apply this technique to the difference matrix computation where each tile of the difference matrix is handled by a 2-dimensional block. The tiling process requires for each block to load $Tile_Q$ query images into shared memory and $Tile_D$ database images as described in Algorithm 1. Due to shared memory size limitations, we use the unsigned char type for the allocated buffers in shared memory. Although the use of 1-byte unsigned char type is not recommended in shared memory as bank conflicts occur when threads in the same half warp access the same 4 or 8 bytes wide memory banks, the limited size of shared memory imposes its use. In our implementation, we use the buit-in type *uchar1* denoting an unsigned char. We empirically adjust the pair *(Tile$_D$, Tile$_Q$)* through a *time* vs *(Tile$_D$, Tile$_Q$)* evaluation. The best performance, overall, was achieved by *(Tile$_D$, Tile$_Q$)*$= (4,4)$.

input : $\mathscr{D}, \mathscr{Q}$
output: Difference Matrix M

```
__shared__ uchar1 Buf_D[Tile_D × R_x × R_y];
__shared__ uchar1 Buf_Q[Tile_Q × R_x × R_y];
__shared__ int temp[R_x × R_y/2], M_t[Tile_D × Tile_Q];
blockIdx.x ← B_x, blockIdx.y ← B_y;
s_d ← min(Tile_D, D − B_x × Tile_D);
s_q ← min(Tile_Q, D − B_y × Tile_Q);
threadIdx.x ← Idx, Dim ← R_x × R_y;
for i ← Idx to s_d × Dim do
    Buf_D[i] ← 𝒟[i + B_x × TileD × Dim];
    i ← i + 8 × warpSize;
end
synchronise the threads ;
for i ← Idx to s_q × Dim do
    Buf_Q[i] ← 𝒬[i + B_y × TileD × Dim];
    i ← i + 8 × warpSize;
end
synchronise the threads ;
for i ← 0 to s_d do
    for j ← 0 to s_q do
        Initialize temp ;
        for k ← Idx to Dim do
            id ← mod(k, 8 × WarpiSize);
            temp[id] ← temp[id] + |Buf_D[i × Dim +
            k] − Buf_Q[j × Dim + k]|;
            i ← i + 8 × warpSize;
        end
        synchronise the threads ;
        M_t[i × s_q + j] ← Reduction(temp);
    end
end
synchronise the threads ;
Copy M_t to global memory;
```

Algorithm 1: Parallel difference matrix computation

4.2 Difference Matrix Contrast Enhancement

After creating a difference matrix comparing all previously seen locations to each other using SAD scores, SeqSLAM employs local neighbourhood normalisation to remove biases from lighting variations between route traversals. Each element M_i in the difference matrix M is normalised within a fixed range l by subtracting the local mean and dividing by the local standard deviation to enhance the local matching contrast. This process gives the new difference matrix $\hat{M}$:

$$\hat{M}_i = \frac{M_i - \bar{M}_l}{\sigma_l} \quad (2)$$

where $\bar{M}_l$ is the local mean and σ_l is the local standard deviation, in a range l templates around template i.

We apply, as well, the tiling technique to enhance the previously calculated difference matrix, stored in global memory in a row major order. The tile size is fixed to $Tile \times Q$. The threads within each block handle the computation of a tile and are designed to be two-dimensional. In our design, each block of index *0<blockIdx.x<gridDim.x-1* will load $Tile+l$ of M. The blocks handling the border tiles, *i.e.* $block_0$ and $block_{gridDim.x-1}$ only load $Tile+\frac{1}{2}l$.

The cuda kernel begins with loading a tile of M into a preallocated shared memory of size $(Tile+l) \times Q$.

Next, the computation of Q means for *Tile* rows of the enhanced $\hat{M}$ is handled by $Q \times Tile$ threads where *threadIdx.x*<*Tile* and *threadIdx.y*<Q. Each thread sums $(1+l)$ values of M in a column where the y-coordinate is *threadIdx.y* and the x-coordinate starts from *threadIdx.x* to finally divide the sum by the number of values summed up *i.e.*, $(1+l)$. After the kernel has finished computing the means, threads are synchronised to subsequently perform the computation of standard deviations the same way they did for the means computation thus described using the formula of standard deviation and the means already calculated. However, border tiles handled by *blockIdx.x=0* and *blockIdx.x=gridDim.x-1* are special cases. In this way, for the first block, if *threadIdx.x*$\leq \frac{1}{2}l$ the local neighbourhood would be below $(1+l)$ and have thus to adapt its range with each row of $\hat{M}$, *i.e.* each *threadIdx.x*. The same principle applies to the last block. If *threaIdx.x*$\geq \frac{1}{2}l$ the range has to be adapted as well. In our implementation, we compute the range for each thread using thread registers. Finally, each block writes in column major the tile of $\hat{M}$ to global memory. Implementation details thus described are presented in Algorithm 2. However, we only describe the *means* computation given that the same principle applies to standard deviations computation. The final straightforward step is also not presented and it consists in substracting the *mean* and dividing by the *standard deviation* each element M_i to obtain $\hat{M}_i$.

4.3 Route Searching

The inputs to the localization algorithm are the query sequence locally acquired and the database Sequence that serves as the visual memory of the robot. The algorithm attempts to find a subsequence of equal length in the database that is the most similar to all regarded subsequences. This is performed by building the image difference matrix as stated above, contrast enhancing it and then searching for the minimizing sub-route referring to connected regions of low difference in the difference matrix resembling to lines shapes. This process is presented in Figure 1 where, for each starting search position in left column (marked by a green dot), a range of slopes is traversed (highlighted by the semi-transparent green area) modelizing possible other sub-routes that will be traversed for other slopes. For each matrix entry that is traversed during a sub-route traversal, its difference value is added to an accumulative value that is initialized with zero in the beginning. This value is called sub-route score. Since there is a range of possible slopes for each starting point there is an equal number of scores for each sub-route starting point. When all scores are calculated for one starting point the minimal score of them is selected. Finally, the sub-route with the smallest score and the second smallest score are used to compute the best database matching to the input query sequence.

input : Difference Matrix M, l
output: Enhanced Difference Matrix $\hat{M}$

$\_\_shared\_\_\ \hat{M}_{sh}[Tile][Q], M_{sh}[l+Tile][Q];$
$\_\_shared\_\_\ Mean[Tile][Q];$
Initialize M_{sh} *to* 0;
$Idx \leftarrow threadIdx.x, Idy \leftarrow threadIdx.y,$
$B_{id} \leftarrow blockIdx.x, grid \leftarrow gridDim.x;$
$a \leftarrow \max(0, Tile \times Bl_{id} - \frac{1}{2}l);$
$b \leftarrow \min(D, Tile \times Bl_{id} + (Tile-1) + \frac{1}{2}l);$
if $Idx \leq b-a$ **and** $Idy < Q$ **then**
  $M_{sh}[Idx][Idy] \leftarrow M[a+Idx][Idy];$
end
synchronise the threads ;
if $(Idx \leq Q)$ **and** $(Idy < Tile)$ **and** $(B_{id} = 0)$ **then**
  $a \leftarrow \max(0, Idx - \frac{1}{2}l);$
  $b \leftarrow \min(1+l, 1+Idx+\frac{1}{2}l);$
  $Mean[Idx][Idy] \leftarrow \frac{1}{b} \cdot \sum_{i=0}^{b} M_{sh}[a+i][Idy];$
end
synchronise the threads ;
if $(Idx \leq Q)$ **and** $(Idy < Tile)$ **and** $(0 < B_{id} < grid)$ **then**
  $Mean[Idx][Idy] \leftarrow \frac{1}{1+l} \cdot \sum_{i=0}^{1+l} M_{sh}[Idx+i][Idy];$
end
synchronise the threads ;
if $(Idx \leq Q)$ **and** $(Idy < Tile)$ **and** $(B_{id} = grid-1)$ **then**
  $b \leftarrow (1+\frac{1}{2}l) + \min(D - B_{id} \times Tile - 1 - Idx, \frac{1}{2}l);$
  $Mean[Idx][Idy] \leftarrow \frac{1}{b} \cdot \sum_{i=0}^{1+l} M_{sh}[Idx+i][Idy];$
end
synchronise the threads ;

Algorithm 2: Parallel mean computation for Difference Matrix contrast enhancement

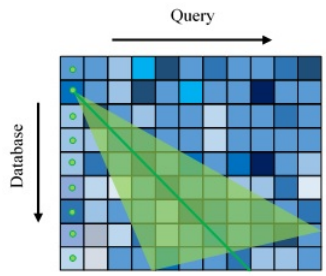


Figure 1: Minimal slope search in the difference matrix.

One of the main important features influencing GPU performance is coalescence issues in global memory access. However, subroute scores computation require multiple non coalesced global memory accesses (to sum up values in diagonals). We hence aim to design a novel subroute score computation that enables coalesced accesses.

One common solution to coalescence issues is the use of shared memory to load data from global memory in a coalesced fashion, then, have contiguous threads stride through it. Unlike global memory, there is no penalty for strided access in shared memory especially when there is no bank conflict. We assume that the difference

matrix sizes are bigger than the shared memory of a single block, which would be the case for the majority of real applications. It is also worth noting that depending on the database size is not a limiting factor in our implementation as the whole process of SeqSLAM is based on a learning phase where several learnable parameters have to be tuned.

We present our design using the mapping vector technique that assigns CUDA ressources including blocks, threads and registers to SeqSLAM data. We will first begin with presenting the global memory mapping vector followed by the SM mapping vector. The main structure of SeqSLAM; the difference Matrix M of size $D \times Q$, is stored in global memory in a column major order according to the following mapping vector:

$$[\underbrace{M_0 M_1 \dots M_{D-1}}_{\text{col 0}} \underbrace{M_D M_{D+1} \dots M_{2D-1}}_{\text{col 1}} \dots \underbrace{M_{D(Q-1)-1} \dots M_{DQ-1}}_{\text{col Q-1}}] \tag{3}$$

This means that columns of size D are contiguously stored in global memory. Hence $M(0,0)$ starts at location 0, $M(0,1)$ at location D and $M(i,j)$ at location $j \times D + i$.

The proposed mapping vector of data on the *SM* ressources is the following:

$$[\underbrace{M_0, \dots, M_{2D-1}}_{\text{block 0}}, \underbrace{M_{2D}, \dots, M_{4D-1}}_{\text{block 1}}, \dots, \underbrace{M_{2\times i\times D}, \dots, M_{2\times(i+1)\times D-1}}_{\text{block i}},$$
$$\dots, \underbrace{M_{D(Q-2+MOD(Q,2))-1}, \dots, M_{DQ-1}}_{\text{block ceil(Q/2)}}] \tag{4}$$

This means that each block i performs a coalesced memory access to global memory in order to load data from location $2 \times i \times D$ to location $2 \times (i+1) \times D - 1$ into its shared memory which is equivalent to 2 columns of the difference Matrix M. The last block of index $blockIdx.x = ceil(Q/2)$ loads the remaining columns which would be either two if $MOD(Q,2) = 0$ or only one column if not. Thus, a total number of $ceil(Q/2) + 1$ one-dimensional blocks is used in the launch configuration of the kernel and 4 warps of threads per block.

As shown in Algorithm 3, the function of each block is, therefore, to compute a subscore for each sub-route where the used values are only strided by D+*offset* with *offset*$\in \{0,1,2\}$. Within a block, the subscore of a given sub-route for all slope possibilities is handled by a thread. We note that simultaneous accesses of threads within a block to shared memory are done to consecutive values, *i.e.* a column of the matrix M to minimize bank conflicts. As only 4 *warps* are used per block, $128 \times k$ subscores are computed in parallel by the active threads, where k designs the number of slope possibilities. The process is, hence, repeated $(1 + floor(D/128))$ times to account for the D possible subroutes. The $128 \times k$ subscores are then written to global memory via an *atomicAdd* between the active blocks. We note that the *atomicAdd* is done inside the loop due to the limited size of shared memory and only $128 \times k$ floats are allocated for *Scores* vector. The final Score for each sub-route would be equal to the sum of the calculated subscores for the slope possibility k.

5 RESULTS

In this section, we describe the conducted experiments to evaluate the performance of porting SeqSLAM to CUDA GPU. We will first begin with exposing the experimental setup, then, discuss the performance expressed in terms of timing of our implementation versus exsisting CPU implementation.

5.1 Experimental Setup

For the experiments described in the following, we extracted still frames from the original video of the Nordland dataset[1], downsampled them to 64×32 and converted them into grayscale. It is worth noting that in related litterature, the downscaling was 64×32 for not only the Nordland dataset, but also the Alderley[2]. The downscaling was even greater for other datasets such as the Nurburgring dataset, equal to 32×24 and the approach of SeqSLAM is fundamentally based on important downscaling ratios.

Furthermore, all the data reside in the GPU device memory at the beginning of each test, so there are no data transfers to CPU during the benchmarks to prevent interactions with other factors in the study. The performance of the experiments for Parallel SeqSLAM is measured in time t in $milliseconds(ms)$. The system on which our implementation was evaluated is equipped with an i7 CPU running at up to 3.5 GHz, the intel i7 CORE. The CUDA device is an NVIDIA GeForce GTX 850M running at 876 MHz with 4096 MB of GDDR5 device memory. The evaluation has been performed with CUDA version 7.5 integrated with VisualStudio 2012. At the first execution of SeqSLAM, memory allocations have to be performed. This is required only once and takes about $10ms$. All the experiments were run for 5 database sequence lengths as presented in Table 1. Morever, 3 query sequences were used with 3 different lengths as presented in Table 2. As stated before, database sequences were obtained using the open-Source Code OpenseqSLAM[3] by varying the parameter *imageSkip*. However, the CPU-based code used

[1] http://nrkbeta.no/2013/01/15/nordlandsbanen-minute-by-minute-season-by-season/

[2] https://wiki.qut.edu.au/display/cyphy/Michael+Milford+Datasets+and+Downloads

[3] https://openslam.org/openseqslam.html

input : $\hat{M}$, mov_{min}, mov_{max}
output: $Match_{index}$, $Match_{score}$

$slopes \leftarrow mov_{max} - mov_{min} + 1$;
$Idx \leftarrow threadIdx.x$;
$B_{id} \leftarrow blockIdx.x$;
$\_\_shared\_\_$ $Icrem_{indices}[slopes][Q], Score[slopes][Q]$;
$\_\_shared\_\_$ $Buffer[2 \times D], Score[D][128]$;
if $Idx = 0$ **then**
  $Slopes \leftarrow FindSlopePossibilities(mov_{min}, mov_{max})$;
  $Icrem_{indices} \leftarrow SlopeIndices(Slopes)$;
end
synchronise the threads ;
for $i \leftarrow Idx + 2 \times D \times B_{id}$ **to** $2 \times D \times (1 + B_{id})$ **do**
  $Buffer[i - 2 \times D \times B_{id}] \leftarrow \hat{M}[i]$;
  $i \leftarrow i + 4 \times warpSize$;
end
synchronise the threads ;
for $s \leftarrow Idx$ **to** D **do**
  $intindices[slopes][Q]$;
  for $i \leftarrow 0$ **to** $slopes$ **do**
    for $j \leftarrow 0$ **to** Q **do**
      $indices[i][j] \leftarrow j \times D + s +$
      $\min(Icrem_{indices}[i \times Q + j], D - s)$;
    end
  end
  synchronise the threads ;
  for $i \leftarrow 0$ **to** $slopes$ **do**
    $sum \leftarrow 0$; **for** $j \leftarrow 0$ **to** 2 **do**
      $index \leftarrow indices[i][j + 2 \times B_{id}]$;
      $sum \leftarrow sum + Buffer[index - 2 \times D \times B_{id}]$;
    end
    $Subscore[i][mod(s, 128)] \leftarrow sum$;
  end
  synchronise the threads ;
  atomicAdd of Subscore in Temp of size $D \times slopes$
  in global memory;
  $s \leftarrow s + 4 \times warpSize$;
end
$Idx \leftarrow threadIdx.x \times blockDim.x + blockIdx.x$;
for $k \leftarrow Idx$ **to** D **do**
  $Score[k] \leftarrow$ `MinOverSlopes`$(Temp)$;
  $k \leftarrow k + blockDim.x \times gridDim.x$;
end
synchronise the threads ;
$min1 \leftarrow$ `FindMinWithShuffle`$(Score)$;
$index_1 \leftarrow$ `FindIndexOfMin`(min_1);
Set values in Score around min_1 *of radius* R_{window} *to max machine value*;
$min2 \leftarrow$ `FindMinWithShuffle`$(Score)$;
$Match_{index} \leftarrow index_1$;
$Match_{score} \leftarrow min_1 / min_2$;

Algorithm 3: Parallel Route Searching

in benchmarking is a C++/OpenCV port of OpenSeqSLAM [4].

Database	Database Seq Length	imageSkip
D1	714	50
D2	1428	25
D3	2747	13
D4	3570	10
D5	5100	7

Table 1: Database Sequences

Query	Query Seq Length
query 1	11
query 2	20
query 3	32

Table 2: Query Sequences

5.2 CUDA based implementation vs CPU based implementation

5.2.1 *Difference Matrix Computation timing*

In the first experiment, we measured the execution time of difference matrix computation kernel using the query and the database sequences presented in Table 1 and Table 2 respectively. We selected the pair $(Tile_D, Tile_Q) = (4,4)$. The evaluation is depicted in Figure 2 for $query = 32$ showing an increasingly important speedup of CUDA-based parallel SeqSLAM with the database sequence length. The speedup is averaged at $4\times$.

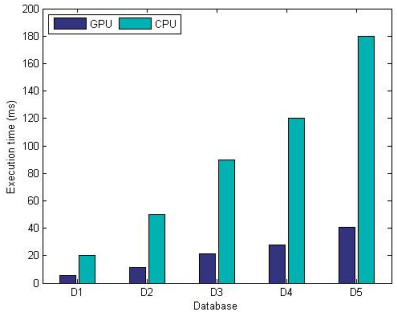


Figure 2: Difference matrix computaion.

5.2.2 *Difference Matrix Enhancement timing*

We subsequently evaluated the performance of parallel SeqSLAM contrast enhancement over Sequential contrast enhancement. The optimal tile selected is $Tile = 20$ and the performance for both sequential and parallel implementations is presented in Figure 3 for $query = 32$. The obtained speedup exceeds $16\times$ for D1 and almost $13\times$ for D5. Overall, a good speedup was shown with parallel seqSLAM averaged at $14\times$.

[4] https://github.com/subokita/OpenSeqSLAM

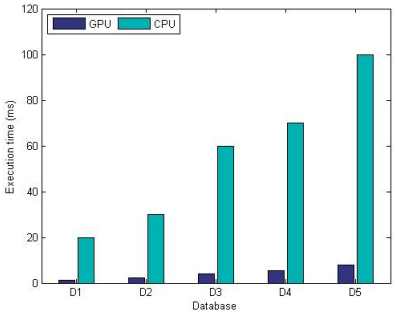


Figure 3: Difference matrix enhancement.

5.2.3 *Route searching timing*

The third experiment was dedicated to the comparison of the Route searching execution time for both CPU-based and CUDA GPU- based implementations. The speedup of GPU over CPU is clearly visible averaged at almost 6× for the datasets used in the evaluation and is depicted in Figure 4 for $query = 32$. An interesting point is that the route searching is not a botteleneck for CPU when attempting to localize a single image using a short query sequence as recommended. However, we aim to design a GPU only solution and a good speedup was though shown.

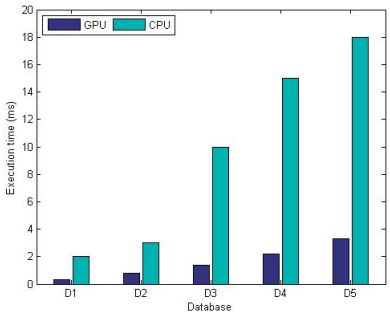


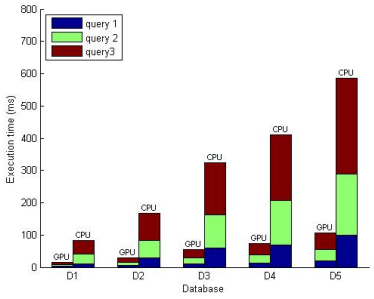
Figure 4: Route searching.

5.2.4 *Performance anlalysis of CUDA based SeqSLAM*

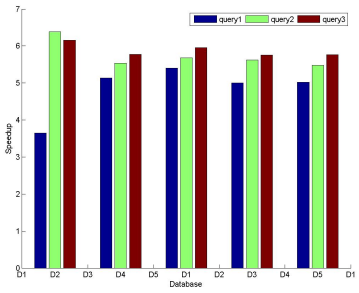
In Figure 5, we show the performance results of parallel SeqSLAM using a CUDA GPU. Firstly, in Figure 5a, we compare the mean computation time of CPU and GPU implementations, for the different database and query sequences used in this evaluation. We show a computation time even more important for CPU reaching 298ms for $D5$ and $query3$ against 51ms for GPU. In figure 5b we demonstrate that the speedup is about 6× compared to the CPU implementation.

5.3 Discussion

A special feature of the Nordland dataset is that the viewpoint for the database and query sequences is exactly the same. The camera has only one degree of freedom along its track and corresponding images from two traverses in different conditions overlap almost perfectly. This condition is only met for certain applications where the camera has only one degree of freedom and SeqSLAM remains a major approach from which many recent approaches are being built upon. Hence, since our aim is to enhance timing and not localization precision, we have relied on the Nordland dataset images as input for different database and query sequences sizes. Furthermore, we aimed to design a GPU only solution as in a real application for a robotic platform, several modules are required and we can put some tasks that do not depend on the results of the kernel on the CPU while the GPU is executing a different task in order to achieve a better performance and reduce the hardware complexity. In fact, this is possible thanks to the asynchronous execution feature between the CPU and the GPU of CUDA systems meaning that the control returns to the CPU immediately after the kernel is launched.



(a) GPU vs CPU time



(b) Speedup

Figure 5: Performance of CUDA based SeqSLAM.

6 CONCLUSION

In this paper, we have proposed a parallel CUDA GPU based implementation of Sequence SLAM (SeqSLAM) for visual place recognition. SeqSLAM is a well-known successful approach for mobile localization and place recognition in varying conditions like seasonal changes and day and night cycles when certain conditions are met. Our parallelization method was based on the allocation of the three major steps of the approach to three GPU kernels, each of which with a specific launch

configuration promoting the best possible performance. Experimentations showed promising results thanks to the parallel exploitation of CUDA ressources that offered a good overall speedup averaged at 6 times better than its CPU counterpart. As future work, we plan to apply other algorithmic techniques instead of simple image difference to deal better with a reasonable viewpoint change. We plan also to use a multi-GPU system to scale with bigger databases and achieve a greater speedup.

7 REFERENCES

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *In. Elsevier Computer Vision Image Understanding*, 110:346–359, 2008.

[2] M. Cummins and P. Newman. Appearance-only slam at large scale with fab-map 2.0. *In. The International Journal of Robotics Research (IJRR)*, 30:1100–1123, 2011.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Washington, DC, USA*, 110:886–893, 2005.

[4] E. Johns and G. Z. Yang. Feature co-occurrence maps: Appearance-based localisation throughout the day. pages 3212–3218, 2013.

[5] D. B. Kirk and W. W. Hwu. *Programming Massively Parallel Processors: a Hands-on Approach*. 2010.

[6] E. Lindholm, J. Nickolls, S. Oberman, and J. Montrym. Nvidia tesla: A unified graphics and computing architecture. *In. IEEE Microwave Magazine (IEEE Micro)*, 28:39–55, 2008.

[7] D. G. Lowe. Object recognition from local scale-invariant features. *Proc. of the International Conference on Computer Vision (ICCV), Washington, DC, USA*, 2:1150–, 1999.

[8] W. Maddern, M. Milford, and G. Wyeth. Cat-slam: probabilistic localisation and mapping using a continuous appearance-based trajectory. *In. The International Journal of Robotics Research (IJRR)*, 31:429–451, 2012.

[9] M. Milford. Vision-based place recognition: how low can you go? *In. The International Journal of Robotics Research (IJRR)*, 32:766–789, 2013.

[10] M. Milford and G. Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. pages 1643–1649, 2012.

[11] T. Naseer, L. Spinello, W. Burgard, and C. Stachnis. Robust visual robot localization across seasons using network flows. pages 2564–2570, 2014.

[12] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. *Proc. of International Conference on Computer Vision (ICCV), Barcelona, Spain*, pages 2320–2327, 2011.

[13] A. Oliva and A. Torralba. Building the gist of a scene: the role of global image features in recognition. *In. Visual Perception, Progress in Brain Research*, 155, Part B:23 – 36, 2006.

[14] E. Pepperell, P. Corke, and M. Milford. All-environment visual place recognition with SMART. *Proc. of IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China*, pages 1612–1618, 2014.

[15] V. Prisacariu and I. Reid. fasthog - a real-time gpu implementation of hog. Technical Report 2310/09, Department of Engineering Science, Oxford University, 2009.

[16] N. Sunderhauf, P. Neubert, and P. Protzel. Predicting the change, a step towards life-long operation in everyday environments. 2013.

[17] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Empowering visual categorization with the gpu. *In. IEEE Transactions on Multimedia (IEEE Trans. Multimedia)*, 13:60–70, 2011.

[18] Y. Wang, Z. Feng, H. Guo, C. He, and Y. Yang. Scene recognition acceleration using cuda and openmp. *Proc. of First International Conference on Information Science and Engineering (ICISE), Nanjing, China*, pages 1422–1425, 2009.

[19] W. Yan, X. Shi, X. Yan, and L. Wang. Computing opensurf on opencl and general purpose gpu. *In. International Journal of Advanced Robotic Systems (IJARS)*, 10:375, 2013.

[20] Z. Yonglong, M. Kuizhi, J. Xiang, and D. Peixiang. Parallelization and optimization of sift on gpu using cuda. *Proc. of IEEE International Conference on High Performance Computing and Communications (HPCC), Zhangjiajie, China*, pages 1351–1358, 2013.

[21] J. Zhou, Q. Guo, H. V. Jagadish, W. Luan, A. K. H. Tung, Y. Yang, and Y. Zheng. Generic inverted index on the GPU. *In. Computing Research Repository (CoRR)*, abs/1603.08390, 2016.

Screen-Space Ambient Occlusion for Light Field Displays

Oleksii Doronin
Holografika
Budapest, Hungary
o.doronin@holografika.com

Peter A. Kara
Kingston University
London, UK
p.kara@kingston.ac.uk

Attila Barsi
Holografika
Budapest, Hungary
a.barsi@holografika.com

Maria G. Martini
Kingston University
London, UK
m.martini@kingston.ac.uk

ABSTRACT

In this paper, we discuss the challenge of generating the screen-space ambient occlusion (SSAO) visual effect on the state-of-the-art HoloVizio light field display. We detail the main features of modern SSAO techniques that are currently being applied during visualization on conventional 2D displays, and describe difficulties that potentially can appear when implementing this visual effect on 3D light field or similar systems. The main contribution of this paper is our own modification of the SSAO algorithm for light field displays. The paper also includes suggestions on the possible ways to improve its visual quality and performance.

Keywords

Screen-space ambient occlusion (ssao), surface shading, light field, light field display, algorithm performance.

1 INTRODUCTION

Ambient occlusion (AO) is a technique of modeling the behavior of ambient light. It takes into account all irregularities of the surface of a virtual scene. After this technique is applied, rough surfaces look more contrast and realistic. This effect creates an impression that the scene elements with relatively bigger depth appear darker, while elements on the top remain light (e.g., like at the pencil drawing of a human face on Figure 1). Physical explanation of the described phenomenon is that the incoming light is easily reflected by the top parts of the surfaces, but vanishes at their bottoms because of multiple reflections with absorption.

Currently the main approaches to implement AO are the global illumination techniques [Chris10, Dim08, Tab04], the methods using simplified scene geometry [Bun05, Laine10, Shan07], and the *screen-space ambient occlusion* (SSAO) techniques [Hoang10, Mitt07].

The global illumination approach is based on the simulation of the physical properties of light itself. It is usually implemented in graphical frameworks that target the generation of complex photo-realistic effects. This includes caustics, multiple reflections, refractions, atmospheric conditions, indoor or outdoor lighting, etc. The ambient occlusion effect is present here as a natural consequence of the physical behavior of light. The main drawback of this approach is that it is usually far away from being able to run in real-time, which results in its applications for offline-only systems, like the ones that are used in the production of photo-realistic posters, movies, or design solutions (e.g., architectural design).

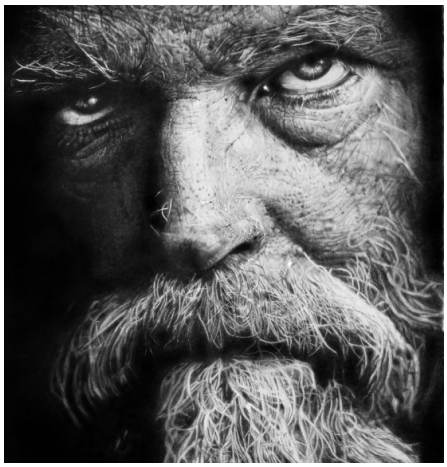


Figure 1: Drawing of a human face by Franco Clun. Details with relatively bigger depth (with their neighborhoods) are painted darker.

Other approaches can be considered as a simplification of the global illumination approach. They are usually designed to simulate some particular visual effects. This paper focuses on the screen-space ambient occlusion (SSAO) method that simulates ambient occlusion taking into account only visible parts of the virtual scene.

All techniques described above work well with conventional 2D displays. However, it is far from trivial to make them work on the state-of-the-art 3D visualization systems, like augmented or virtual reality (AR and VR), stereoscopic displays that work with glasses, multi-

view displays, or light field displays (LFDs). Light field displays – such as the HoloVizio system [Balazs14, Balogh07] – require much more computational power for rendering compared to the visualization on 2D displays. Because of this, graphics algorithms need to be as light-weight and as efficient as possible. Based on its attributes, the SSAO can be considered to be a fair solution for this type of systems.

This paper introduces the adaptation of SSAO algorithm of CryEngine 2 (see [Mitt07]) to the HoloVizio light field display, as well as some improvements for it.

The paper is structured as follows. Section 2 provides a brief overview of the state-of-the-art work performed in the topic of surface shading. This is followed by the overview of the HoloVizio light field visualization system in Section 3. The operation of the conventional SSAO technique is detailed in Section 4, followed by our contribution of a 3D SSAO extension in Section 5 and possible quality improvements in Section 6. Section 7 describes the performance of our algorithm. The paper is concluded in Section 8, also pointing out potential continuations of the investigated topic.

2 RELATED WORK

It is intuitively understandable that human observers can perceive the variations of depth by the level of darkness of the appropriate parts of the image. This fact was empirically proven [Lan00], together with some clarifications on the accuracy of subjective depth estimation under certain conditions. This evidence gives us the scientific justification of using the ambient occlusion family of algorithms to depict the variation of depth by the variation of light intensity.

Currently, the most precise method to depict the variation of light intensity in a given scene is to make a physically-based simulation. This approach is directly implemented in the global illumination (GI) frameworks, and it is widely applied in the creation of modern movies and cartoons [Chris10, Tab04]. However, using GI techniques is not always an optimal solution for real-time rendering – such as virtual and augmented reality, video games, or the real-time preview during the editing of 3D graphics – as these techniques come with higher requirements for computational power.

Another approach to render the photo-realistic images is to use the *ray tracing* family of methods. This includes the ray tracing method in its classic form [Wald09, Whit79], *path tracing* (also known as the Monte Carlo ray tracing [Veach97]), bidirectional path tracing [Laf93], etc. However, these mentioned approaches usually require additional computational power and necessitate the construction of additional acceleration structures (e.g., BVH, BIH, kd-trees, octrees, etc.). Also, despite of numerous algorithmic and hardware improvements, and the fact that they are already being widely applied offline, they are still only entering the market of existing real-time solutions at the time of this paper.

One way to avoid the unnecessary complexity of ray tracing algorithms in real-time frameworks is to trace the whole beam of light instead of a single ray. This method is called *cone tracing*, and can be applied in real-time to compute the illumination term only [Cras11].

Instead of doing a complete simulation of light behavior, one can try to imitate only a few necessary visual effects, including the AO effect. The mathematical background for these effects can be derived from the *rendering equation* [Kaj86], altogether with the other effects produced by the global illumination approach. Rather straightforward method to produce the AO effect in a given point of the virtual scene is to trace a limited number of rays in different directions from this point, and to check whether or not they intersect another surface of the scene (see Figure 2). In this case, the AO term is usually defined as the ratio of the number of rays that do not result in intersection to the total number of the emitted rays [Bun05, Dim08, Laine10, Shan07].

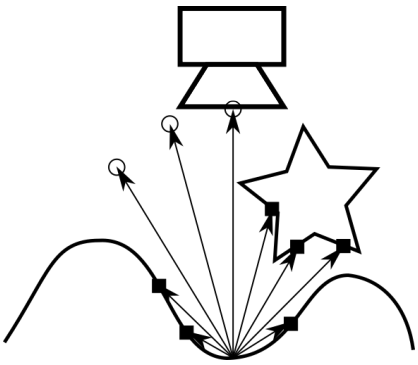


Figure 2: Conventional AO approach. Arrows represent rays emitted from the point of interest. Black squares designate the points that increase the AO value, white circles – that do not.

Several improvements and modifications have been made within the family of AO algorithms. Dimitrov et al. [Dim08] introduced the *horizon-split ambient occlusion*. Laine and Karras [Laine10] developed a ray-tracing-based method that takes into account the neighboring set of geometrical primitives (i.e., triangles). The evaluation of light intensity with the help of an additionally constructed simplified representation of the scene geometry in form of disks is introduced by Bunnell [Bun05], and by Shanmugam *et al.* [Shan07] – in the form of balls.

The main difference between the AO and SSAO algorithms is that the SSAO algorithm computes the AO value for visible surface only (see Figure 3), while conventional AO algorithm uses the whole scene for

this purpose. In conventional graphics pipeline, visible scene geometry can easily be constructed using additional depth texture that contains the information about the depth of each pixel.

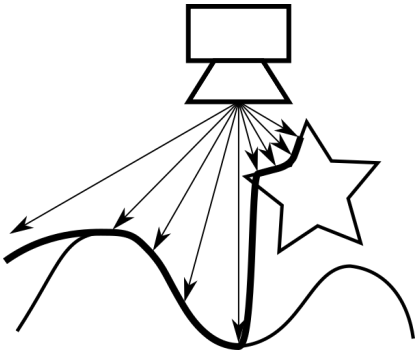


Figure 3: Approximation of the scene geometry by its visible parts (thick black line) used in the SSAO-family of algorithms.

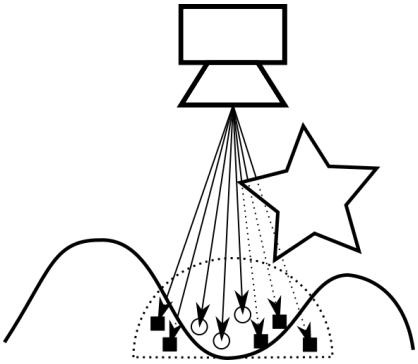


Figure 4: Screen-space ambient occlusion. Black squares designate the invisible samples, white circles are for the visible samples.

The collection of SSAO methods is implemented in CryEngine 2 [Mitt07], which is well-known for the Far Cry video game. In this approach, the random points (*samples*) are generated within a certain volume around the point of interest (*hit point*) on the visible surface. The AO value is defined through the ratio of invisible samples to the overall number of generated samples. Hoang and Low [Hoang10] suggested the *multi-resolution SSAO* that can be combined with other SSAO methods, the main idea of which is to measure the AO term for different radii and merge obtained information.

Loos and Sloan [Loos10] introduced the *volumetric obscurance* method (see Figure 5) that calculates the cumulative length of the unoccluded rays (depicted by thick gray lines on Figure 5) coming from the camera position instead of their number (as in the SSAO approach of Mittring [Mitt07]). The work of McGuire *et al.* [McGuire11] describes the application of a similar method in the industrial Alchemy framework.

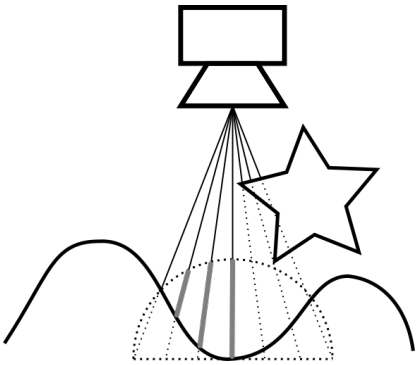


Figure 5: Volumetric obscurance.

3 OVERVIEW OF THE HOLOVIZIO SYSTEM

Probably, one of the most commonly accepted ways to parameterize light field is the 4D *two-plane parameterization* (see Figure 6). Each ray of light in light field is defined by the points of it's intersection with two parallel planes (*screen plane* and *observer plane*), and each point of intersection is parameterized by two coordinates.

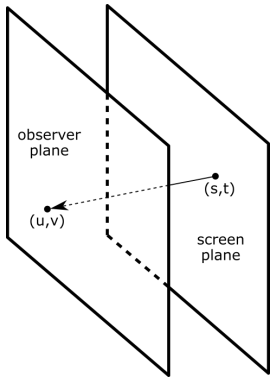


Figure 6: Two-plane 4D parameterization.

Two-plane parameterization is commonly used for full parallax light field displays. But for horizontal-only parallax displays (like HoloVizio system) the observer plane can be substituted by *observer line* [Balazs14] (see Figure 7). In this case, 4D two-plane parameterization can be replaced with 3D *plane-line parameterization* (two coordinates for the screen plane, one for the observer line). Analogously to two-plane parameterization, the observer line can be imagined as the possible positions of the viewer with respect to the actual screen.

Introduced plane-line parameterization is actually used in the most simple architecture of the HoloVizio system. In this form, HoloVizio system consists of actual physical screen (represented by the screen plane) with special diffuse properties, and a row of optical modules placed behind the screen. Each ray of light emitted by an optical module traverses through the screen without obstacles in horizontal direction, but diffused uniformly in all vertical directions to be able to hit the ob-

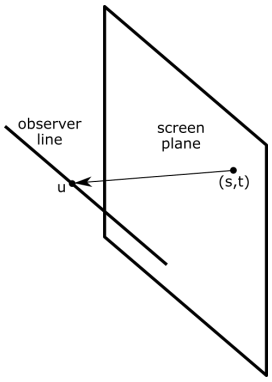


Figure 7: Horizontal parallax 3D parameterization.

server line (see Figure 8). More advanced architectures of the HoloVizio system may include convex display or different placing of optical modules, but the main concept remains the same. Real-life HoloVizio system may contain different number of optical modules (e.g., it is 80 for HoloVizio C80 cinema system).

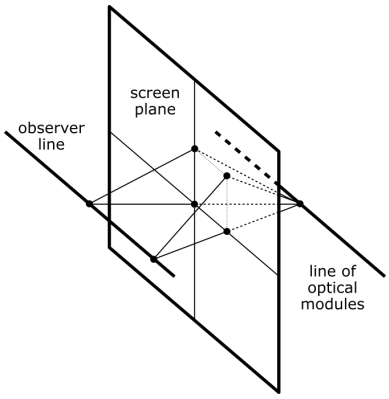


Figure 8: Distortion of particular rays of light coming from a single optical module.

For the practical reasons, we differentiate the following Euclidean 3D spaces: the *world space*, the *physical space*, and the *image space*. In the world space, the coordinates of the virtual scene are expressed. In the physical space, X axis is parallel to the observer line, Y axis goes vertically up, and Z axis is perpendicular to the screen plane. Physical space is introduced to simplify the calculation of the distortion of light that comes through the screen. Image space is defined for each optical module separately. In this space, the actual input 2D texture of the optical module is computed. X and Y axes in the image space are X and Y axes of the texture, and Z axis is the same as Z axis in the physical space.

The transformation from the world space to the physical space is affine and invertible. It can be set up by defining a perpendicular parallelepiped in each space (also known as the *region of interest*, ROI), and calculating the transformation from one to another. While the transformation from the physical space to the image space is not affine and not invertible. The relations between all three mentioned spaces is depicted on Figure 9.

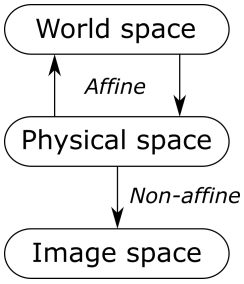


Figure 9: Relations between the world space, the physical space, and the image space.

Although the contribution of this paper is specifically made for HoloVizio light field displays, the introduced algorithm can be extended to other types of light field systems. The main reason to apply the introduced solution to a concrete system would be the necessity to use additional hierarchy of 3D spaces, similar to one depicted on Figure 9.

4 THE CONVENTIONAL SSAO ALGORITHM

In 2D case, the physical space does not exist, and the image space is substituted with the *screen space*. X and Y coordinates in the screen space are the same as X and Y coordinates of the pixel in the final image, and Z coordinate (*depth*) corresponds to the distance from the viewer to the corresponding point of the virtual scene. The exact way of computation of Z coordinate depends on whether orthographic or perspective projection is used. Note that there is an invertible homogeneous transformation between the world space and the screen space. This transformation allows to easily determine the precise position of each visible point on the scene, if both pixel coordinates and depth are known.

Our algorithm is based on the approach published by Mittring [Mitt07] (see Algorithm 1). This algorithm can be implemented as an OpenGL compute shader that is executed per each pixel of the final image. It takes the depth texture as the input, as well as two 4-by-4 matrices that correspond to the world-to-screen and the screen-to-world transformations. First, it calculates the position of a point of the visible surface that corresponds to the current pixel in the screen and the world space coordinates. Then it spawns samples (random 3D points) within a predefined radius `R` of calculated point in the world space. Furthermore, there is a check for each sample whether it is visible (i.e., positioned in front of the visible surface) or not. If the sample is not visible, the algorithm adds a value to the AO variable. In our implementation, the value of AO variable varies from zero to one.

Algorithm 1 Conventional SSAO algorithm. Variables `pntScrn` and `smpScrn` represent coordinates in the screen space, variables `pntWrld` and `smpWrld` – in the world space.

```
1:  pntScrn.xy ← texture coordinate of given pixel
2:  pntScrn.z ← depthTexture(pntScrn.xy)
3:  pntWrld ← screen-to-world(pntScrn)
4:  S ← set of random samples around pntWrld
5:  for each sample from S do
6:      smpWrld ← world coordinates of sample
7:      smpScrn ← world-to-screen(smpWrld)
8:      surDepth ← depthTexture(smpScrn.xy)
9:      if surDepth < smpScrn.z then
10:         increase AO variable
```

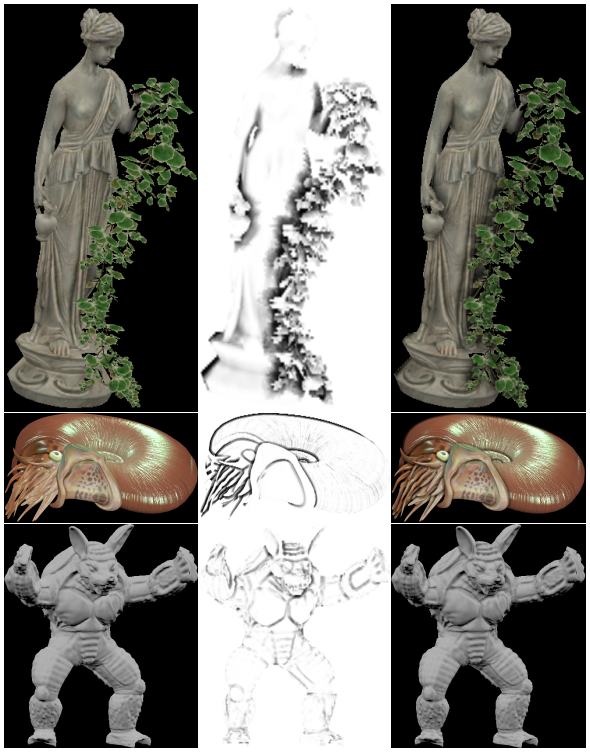


Figure 10: Rendered models of statue (top), ammonite (center) and armadillo (bottom) for one optical module. The left column represents the images without the SSAO effect. Images in the right column are after the SSAO effect was applied. The grayscale images with $(1-\texttt{ao})$ values are in the center.

The easiest way to apply the desired AO effect is to multiply each channel of the resulting RGB image by the $(1-\texttt{ao})$ term, where `ao` is the appropriate value in the computed AO texture. Figure 10 shows an example of this effect.

5 THE THREE-DIMENSIONAL SSAO ALGORITHM

The concept of the image space in HoloVizio system is very close to the concept of the screen space in conventional 2D graphics pipeline. X and Y coordinates of a point in each of these spaces are the pixel coordinate of the output texture of the shading algorithm, while Z coordinate represents the notion of depth in it's particular way. The main problem of application of the algorithm introduced by Mittring [Mitt07] is that the image-to-world transformation (analogous to screen-to-world transformation from algorithm 1) cannot be easily defined for the HoloVizio system.

The solution of the mentioned problem is to compute the world-space positions of the visible surface before the application of SSAO algorithm, and to store them in a separate texture (*positions texture*). In this case, the application of the image-to-world transformation for a point `pntImg` on the visible surface in the image space, can be substituted with getting the value of positions texture at `pntImg`.*xy* coordinates (symbol *.xy* designates the first two coordinates of a 3D vector). The final modification of the algorithm is presented in Algorithm 2.

Algorithm 2 SSAO algorithm for HoloVizio. Variables `pntImg` and `smpImg` represent coordinates in the image space, variables `pntWrld` and `smpWrld` – in the world space.

```
1:  pntImg.xy ← texture coordinate of given pixel
2:  pntWrld ← positionsTexture(pScn.xy)
3:  S ← set of random samples around pntWrld
4:  for each sample from S do
5:      smpWrld ← world coordinates of sample
6:      smpImg ← world-to-image(smpWrld)
7:      surDepth ← depthTexture(smpImg.xy)
8:      if surDepth < smpImg.z then
9:          increase AO variable
```

Algorithm 2 looks very similar to the Algorithm 1. The most noticeable difference is that rows 2 and 3 of Algorithm 1 are substituted with row 2 in Algorithm 2. However, there also are two other important distinctions. First, the world-to-image transformation in Algorithm 2 also includes the physical-to-image transformation described in Section 3, which is not invertible and cannot be expressed as a simple combination of homogeneous 4-by-4 matrices. Second, depth texture used in Algorithm 2 contains values obtained with the help of the mentioned world-to-image transformation. Alternatively, taking a value from the depth texture (row 7 in Algorithm 2) can be substituted with two operations: taking the world-space coordinates from positions texture, and applying world-to-image transformation to these coordinates.

6 IMPROVING THE QUALITY

It is possible to improve the quality of the introduced algorithm by the following methods.

Generation of samples

In most of SSAO implementations, samples are generated as a set of points inside a unit sphere (or a ball). Perhaps the most straightforward way to do this is to generate the points with uniform distribution within a bigger volume (cube), and to use the rejection method to leave only the necessary ones. However, we consider this method not to be flexible enough. The main reason for this is that on a small number of samples (around 8) the generated set can be not very well represented in all parts of the sphere.

In our implementation, we use the cylindrical parameterization to generate points inside the sphere uniformly. Algorithm 3 shows the usage of this parameterization. To use this approach to implement the conventional random uniform distribution on the spherical surface, one have to substitute the values ξ_1 and ξ_2 in this algorithm with the uniformly distributed random values from -1 to 1.

Algorithm 3 Generation of points on the spherical surface in cylindrical coordinates.

1: $\xi_1, \xi_2 \leftarrow$ values from -1 to 1;
2: $z \leftarrow \xi_1$;
3: $\varphi \leftarrow \pi\xi_2$;
4: $r \leftarrow \sqrt{1-z^2}$;
5: $x \leftarrow r\cos\varphi$;
6: $y \leftarrow r\sin\varphi$;

Algorithm 4 introduces the way to generate the coordinates of the point with uniform distribution in the inner volume of the ball with a given radius R, based on pregenerated points on the unit sphere. Thus, to get the uniformly distributed set of samples within a ball of radius R, one has to supply Algorithm 3 with ξ_1 and ξ_2 uniformly distributed on $[-1,1]$, and Algorithm 4 with $r_{min} \leftarrow 0$ and $r_{max} \leftarrow R$. In our implementation, we found that the choice $r_{max} \leftarrow R$ and $r_{min} \leftarrow 0.8 \cdot R$ is suitable enough.

Algorithm 4 Generation of random radius uniformly inside the given boundaries.

1: $x, y, z \leftarrow$ coordinates of a point on the unit sphere (output from Algorithm 3);
2: $r_{min}, r_{max} \leftarrow$ min and max boundaries of the radius;
3: $t \leftarrow$ uniformly distributed value from r_{min}^3 to r_{max}^3;
4: $r \leftarrow \sqrt[3]{t}$;
5: $x \leftarrow r \cdot x; \quad y \leftarrow r \cdot y; \quad z \leftarrow r \cdot z;$

The most flexible part in the introduced way to generate random samples using cylindrical coordinates lies in different methods to choose ξ_1 and ξ_2 values from Algorithm 3. One can easily notice that the parametric space of ξ_1 and ξ_2 values is nothing more than the $[-1,1] \times [-1,1]$ square in ordinary case. Thus, one can try to generate the random points in this parametric space in more sophisticated ways. We implemented the jittered sampling (subdividing space into grid cells and generating equal number of uniformly distributed points inside each cell), the Latin square approach (subdividing space into grid cells and generating only one random sample in each row and column), and the Poisson disk sampling [Brid07]. Although, we have not yet performed any subjective test that would reflect the significant difference in the perceived quality of the resulting images for each approach.

Additional blur pass

The algorithmic complexity of the introduced SSAO approach depends linearly on the number of samples being used. The common method to speed up the algorithm is to use a smaller number of samples, but compensate it with proper placement (e.g., by the methods described in the previous subsection). We found that it is reasonable to use approximately 8 samples. However, if we use a small number of samples, some visual artifacts in the final image may appear. To avoid these artifacts, we can do an additional Gaussian blur pass with a small kernel radius after the SSAO texture is rendered (see Figure 11).

Figure 11: Ambient texture before (left) and after (right) the blur pass was applied. Note that the artifacts in form of single white pixels on the left image that disappear after the blur pass.

Multi-resolution SSAO

There are usually many details of different size and shape in the virtual scene. This means that when the SSAO algorithm is applied, the radius of the sphere which incorporates the random samples should depend on the surrounding geometry. However, using additional information about the needed level of detail would add some complexity to the algorithm, making it slower in the end. To solve this problem, we implemented the *multi-resolution* ambient occlusion algorithm (see Figures 12, 13) introduced by Hoang and Low [Hoang10]. The main idea of this solution is to calculate the AO term for a couple of different radii (5 in our framework) and to merge the obtained information (we take the maximal AO contribution).

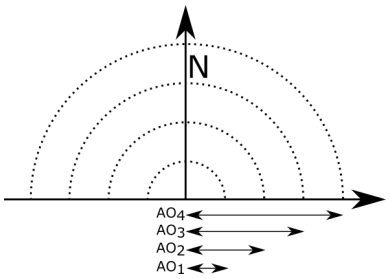


Figure 12: Multiresolution ambient occlusion. Labels $AO_1, ..., AO_4$ depict the areas in which the AO term is computed separately.

Figure 13: Ambient texture with single (left) and multiple AO radii (right).

7 PERFORMANCE

All mentioned techniques were implemented in a single framework for the HoloVizio system. For this purpose, we used the deferred rendering scheme. In total, there are six render passes for each frame:

1. Z prepass – preliminary checking of visibility;
2. G-buffer – generation of diffuse color texture, position texture, etc.;
3. SSAO texture generation;
4. horizontal Gaussian blur for SSAO texture;
5. vertical Gaussian blur for SSAO texture;
6. post processing – application of lights and SSAO effect.

We measured performance of the HoloVizio framework on a machine with Intel i7-5820K 3.30GHz CPU and GeForce GTX 960 video card. The tested HoloVizio system was virtual, with only one optical module of 1024-by-768 resolution. The rendered models are shown on Figure 10, and include:

1. *statue* (5011 faces, 3 textures);
2. *ammonite* (29520 faces, 4 textures);
3. *armadillo* (212574 faces, 0 textures).

For each model, we calculated the average time to render a single frame as the arithmetic average of rendering time of 1000 frames (see Table 1) with and without the application of the SSAO algorithm. Results from Table 1 show that the time to apply the SSAO effect itself can be considered as constant (for one HoloVizio system), and in the tested virtual system it equals to approximately 6.45 milliseconds.

Model	No SSAO	SSAO	Difference
Statue	0.79	7.25	6.46
Ammonite	1.62	8.05	6.43
Armadillo	2.48	8.96	6.48

Table 1: Average time to render a single frame in milliseconds.

An example of the model of statue displayed on a real HoloVizio system with the SSAO effect is shown on Figure 14. The photo displayed in the figure was captured by a regular digital camera.

Figure 14: The model of statue rendered on the HoloVizio 640RC system[1]. Left image is rendered without SSAO effect, right image – with. Note that the images above are rendered with the help of multiple optical modules, not a single one. These images are not the same with the images for the optical modules from Figure 10.

8 CONCLUSIONS

In this paper, we proposed the modification of the method used for generating the SSAO effect, in order to make it suitable for the state-of-the-art HoloVizio light field system, and described several ways to improve its quality and performance. This algorithm can also be applied to different light field systems or systems that aim to generate immersive 3D environments

[1] Mentioned model of HoloVizio system is out of the market now, and the closest analogue to it is the HoloVizio 722RC system by the time of this paper.

(e.g., virtual reality, augmented reality, stereoscopic displays, etc.). Further research on this topic shall include modifications of other techniques (e.g., global illumination) of conventional 2D frameworks to make them compatible with light field systems and 3D visualization in general.

9 ACKNOWLEDGEMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 676401. The work in this paper was funded from the European Union's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant agreement No 643072, Network QoE-Net.

10 REFERENCES

[Balazs14] Balázs, Á., Barsi, A., and Kovács, P. T., and Balogh, T. Towards mixed reality applications on light-field displays, in 3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), 2014, pp. 1–4, IEEE, 2014.

[Balogh07] Balogh, T., Kovács, P.T., and Megyesi, Z. Holovizio 3d display system, in Proceedings of the First International Conference on Immersive Telecommunications, in ImmersCom '07 (ICST), Brussels, Belgium), pp. 1–5, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[Brid07] Bridson, R. Fast poisson disk sampling in arbitrary dimensions., in SIGGRAPH sketches, p. 22, 2007.

[Bun05] Bunnell, M. Dynamic ambient occlusion and indirect lighting. Gpu gems, Vol. 2, No. 2, pp. 223–233, 2005.

[Chris10] Christensen, P.H. Point-based global illumination for movie production, in ACM SIGGRAPH 2010 Course Notes, 2010.

[Cras11] Crassin, C., Neyret, F., Sainz, M., Green, S., and Eisemann, E. Interactive indirect illumination using voxel cone tracing, in Computer Graphics Forum, Vol. 30, pp. 1921–1930, Wiley Online Library, 2011.

[Dim08] Dimitrov, R., Bavoil, L., and Sainz, M. Horizon-split ambient occlusion, in Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games, I3D '08, (New York, NY, USA), p. 1, ACM, 2008.

[Hoang10] Hoang, T.D., and Low, K.L. Multi-resolution screen-space ambient occlusion, in Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology, VRST '10, New York, USA, pp. 101–102, ACM, 2010.

[Kaj86] Kajiya, J.T. The rendering equation, in ACM Siggraph Computer Graphics, Vol. 20, pp. 143–150, ACM, 1986.

[Laf93] Lafortune, E.P., and Willems, Y.D. Bi-directional path tracing, in Proceedings of the 3rd International Conference on Computational Graphics and Visualization Techniques (Compugraphics), Vol. 10, (Alvor, Portugal), pp. 145–153, 1993.

[Laine10] Laine, S., and Karras, T. Two methods for fast ray-cast ambient occlusion, in Proceedings of the 21st Eurographics Conference on Rendering, EGSR'10, Aire-la-Ville, Switzerland, pp. 1325–1333, Eurographics Association, 2010.

[Lan00] Langer, M.S., and Bülthoff, H.H. Depth discrimination from shading under diffuse lighting. Perception, Vol. 29, No. 6, pp. 649–660, 2000.

[Loos10] Loos, B.J., and Sloan, P.P. Volumetric obscurance, in Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games, pp. 151–156, ACM, 2010.

[McGuire11] McGuire, M., Osman, B., Bukowski, M., and Hennessy, P. The alchemy screen-space ambient obscurance algorithm, in Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics, pp. 25–32, ACM, 2011.

[Mitt07] Mittring, M. Finding next gen: Cryengine 2, in ACM SIGGRAPH 2007 Courses, SIGGRAPH '07, New York, USA, pp. 97–121, ACM, 2007.

[Shan07] Shanmugam, P., and Arikan, O. Hardware accelerated ambient occlusion techniques on gpus, in Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games, I3D '07, New York, USA, pp. 73–80, ACM, 2007.

[Tab04] Tabellion, E., and Lamorlette, A. An approximate global illumination system for computer generated films. ACM Trans. Graph., Vol. 23, pp. 469–476, Aug. 2004.

[Veach97] Veach, E., and Guibas, L.J. Metropolis light transport, in Proceedings of the 24th annual conference on Computer graphics and interactive techniques, pp. 65–76, ACM Press/Addison-Wesley Publishing Co., 1997.

[Wald09] Wald, I., Mark, W. R., Günther, J., Boulos, S., Ize, T., Hunt, W., Parker, S.G., and Shirley, P. State of the art in ray tracing animated scenes, in Computer Graphics Forum, Vol. 28, pp. 1691–1722, Wiley Online Library, 2009.

[Whit79] Whitted, T. An improved illumination model for shaded display, in ACM SIGGRAPH Computer Graphics, Vol. 13, p. 14, ACM, 1979.

fastGCVM: A Fast Algorithm for the Computation of the Discrete Generalized Cramér-von Mises Distance

Johannes Meyer
Karlsruhe Institute of Technology,
Vision and Fusion Laboratory,
Adenauerring 4
76131 Karlsruhe,
Germany
johannes.meyer@kit.edu,
www.meyer-research.de

Thomas Längle
Fraunhofer Institute of Optronics,
System Technologies and Image Exploitation IOSB,
Fraunhoferstr. 1
76131 Karlsruhe,
Germany
thomas.laengle
@iosb.fraunhofer.de

Jürgen Beyerer
Fraunhofer Institute of Optronics,
System Technologies and Image Exploitation IOSB,
Fraunhoferstr. 1
76131 Karlsruhe,
Germany
juergen.beyerer
@iosb.fraunhofer.de

ABSTRACT

Comparing two random vectors by calculating a distance measure between the underlying probability density functions is a key ingredient in many applications, especially in the domain of image processing. For this purpose, the recently introduced generalized Cramér-von Mises distance is an interesting choice, since it is well defined even for the multivariate and discrete case. Unfortunately, the naive way of computing this distance, e.g., for two discrete two-dimensional random vectors $\tilde{\mathbf{x}}, \tilde{\mathbf{y}} \in [0,\dots,n-1]^2, n \in \mathbb{N}$ has a computational complexity of $O(n^5)$ that is impractical for most applications. This paper introduces fastGCVM, an algorithm that makes use of the well known concept of summed area tables and that allows to compute the generalized Cramér-von Mises distance with a computational complexity of $O(n^3)$ for the mentioned case. Two experiments demonstrate the achievable speed up and give an example for a practical application employing fastGCVM.

Keywords

Distance of random vectors, summed area tables, speed up, histogram comparison, localized cumulative distributions, generalized Cramér-von Mises distance.

1 INTRODUCTION

Applications and algorithms from different fields require to compute a distance between two random vectors, respectively, between the corresponding probability density functions in order to measure their similarity [Cha07]. For example, histogram distances are employed by content based image retrieval systems to find images similar to the query image [MGW10, CS02, DNK03]. Furthermore, such distance measures can be used as an optimization criterion to obtain a Dirac mixture approximation of probability distributions, for interpolations, for parameter estimation or for tracking [PHB13]. Whenever one of the two random vectors is of discrete type, the cumulative distribution functions are usually employed for further processing steps. However, this is only possible for the one-dimensional case since the cumulative distribution of a multivariate random vector is not unique.

In [HK08], the authors have introduced a novel formulation of the cumulative distribution, the so-called localized cumulative distribution. The LCD of a random vector can be imagined as a rectangular kernel transform of the underlying probability density function. Based on the definition of the LCD, [HK08] introduces a generalized formulation of the Cramér-von Mises (CVM) distance that can be used to calculate distances between two (multivariate) random vectors of whom one (or both) may even be of discrete type. However, as it is shown in the following sections, the computation of the LCD and the CVM distance for two discrete two-dimensional random vectors $\tilde{\mathbf{x}}, \tilde{\mathbf{y}} \in [0,\dots,n-1]^2, n \in \mathbb{N}$, as it is typical for image processing applications, has a complexity of $O(n^5)$.

This paper shows, how the concept of so-called summed area tables [Cro84, VJ04] can be used to obtain a novel and more efficient algorithm for computing the LCD and the CVM distance. For the mentioned example random vectors, the proposed algorithm has a reduced complexity of $O(n^3)$.

The paper is structured as follows: Sec. 2 lists and discusses related work performed by other research groups. In Sec. 3, the localized cumulative distributions and the generalized Cramér-von Mises distance are introduced and their computational complexity is analyzed. Furthermore, the concept of summed area tables is described. Section 4 is dedicated to the proposed fast algorithm for the computation of the Cramér-von Mises distance and Sec. 5 covers the performed experiments and the respective results. A summary of the paper and an outlook concerning further research topics is provided in Sec. 6.

2 RELATED WORK

This section describes major work performed by other researches in which either the generalized Cramér-von Mises distance plays an important role or summed area tables have been used. To the knowledge of this paper's authors, neither the concept of summed area tables nor any other acceleration technique has yet been employed to reduce the computation time of the generalized Cramér-von Mises distance.

Franklin Crow was the first to introduce the concept of summed area tables [Cro84]. The respective paper is concerned with the task of reducing the computational costs of the texture mapping problem. Crow shows that it is possible to precompute a certain data structure, a summed area table (see Sec. 3.1), for a given image using linear time and space so that afterwards the sum of the pixel values inside any arbitrary query rectangle can be obtained in constant time.

In [VJ04], Viola and Jones introduce a processing framework for face detection. They employ a large set of features in concert with a classifier cascade trained using the AdaBoost learning algorithm. By adapting the idea of summed area tables to digital images, they can achieve a fast computation of the used features.

Hanebeck et al. and Gilitschenski et al. show in [HHK09, GH13], how the concept of localized cumulative distributions and the generalized Cramér-von Mises distance can be employed to obtain Dirac mixture approximations of multivariate Gaussian distributions. Such approximations are important ingredients, e.g., for state estimation in dynamic systems. In their presented work, the authors achieve an efficient implementation by analytically obtaining a closed-form solution for the LCD and the generalized CVM distance for multivariate Gaussian densities and Dirac mixtures. Since the approach for accelerating the calculation of the LCD and the CVM distance provided in this paper does not rely on any specific form of a probability distribution, it can be used to speed up applications, where no analytical solution can be found.

3 PREREQUISITES

Before the details of the acceleration approach can be described, this section introduces the necessary definitions and data structures beginning with the concept of summed area tables. For the sake of simplicity, the formulas and solutions shown in this paper are often limited to the case of discrete two-dimensional probability distributions since this is the common case for image processing applications.

3.1 Summed Area Tables

Summed area tables denote a data structure that can be precomputed for two-dimensional input arrays allowing to calculate the sum of the array entries inside arbitrary rectangular regions of the array [Cro84].
Let $i(x,y) \in \mathbb{R}, \quad x,y \in [0,\dots,n-1]$ denote a two-dimensional array like data structure. The corresponding summed area table I is defined by

$$I(x,y) := \begin{cases} 0 & \text{if } \min\{x,y\} \leq 0\,, \\ \sum\limits_{x_\mathrm{f}=1}^{x} \sum\limits_{y_\mathrm{f}=1}^{y} i(x_\mathrm{f},y_\mathrm{f}) & \text{otherwise}\,. \end{cases} \tag{1}$$

The data structure I can be calculated in a single sweep over i by employing the iterative formulation

$$I(x,y) = i(x,y) + I(x-1,y) + I(x,y-1) - I(x-1,y-1)\,. \tag{2}$$

By this means, the sum of array entries of i inside an interval $x \in [x_\mathrm{f},x_\mathrm{t}]\,,\ y \in [y_\mathrm{f},y_\mathrm{t}]$ can be obtained via three arithmetic operations on I only:

$$\begin{aligned} \sum_{x=x_\mathrm{f}}^{x_\mathrm{t}} \sum_{y=y_\mathrm{f}}^{y_\mathrm{t}} i(x,y) = \quad & I(x_\mathrm{t},y_\mathrm{t}) \\ & - I(x_\mathrm{f}-1,y_\mathrm{t}) \\ & - I(x_\mathrm{t},y_\mathrm{f}-1) \\ & + I(x_\mathrm{f}-1,y_\mathrm{f}-1)\,. \end{aligned} \tag{3}$$

Figure 1 provides a visualization motivating the formula. Since the iterative expression (2) involves only four constant time arithmetic operations and four constant time array accesses and has to be performed for every element in i, i.e., $n \cdot n = n^2$ times, the computation of I has a complexity of $O(n^2)$ [MS08, Cor09]. By using formula (3), the calculation of a sum of array entries of i in an arbitrary rectangular region has a complexity of $O(1)$. The concept of summed area tables is used in later sections in order to accelerate the computation of the generalized Cramér-von Mises distance.

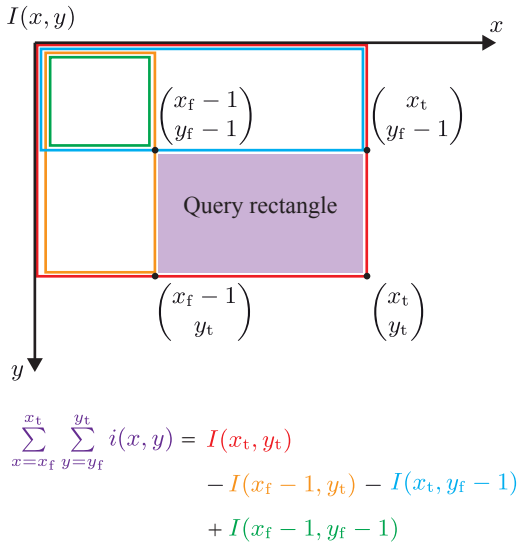


Figure 1: Calculating the sum of the array entries inside the query rectangle $x \in [x_\mathrm{f}, x_\mathrm{t}]$, $y \in [y_\mathrm{f}, y_\mathrm{t}]$ (purple) by employing a summed area table: from the red component $I(x_\mathrm{t}, y_\mathrm{t})$, first the orange component $I(x_\mathrm{f}-1, y_\mathrm{t})$ and the blue component $I(x_\mathrm{t}, y_\mathrm{f}-1)$ have to be subtracted. Since the area represented by the green component has now been subtracted twice, $I(x_\mathrm{f}-1, y_\mathrm{f}-1)$ has to be added.

3.2 Localized Cumulative Distribution

As mentioned before, cumulative distributions are often employed in order to be able to calculate a distance between two random vectors. However, the conventional definition of the cumulative distribution is not unique for the multivariate case. This issue is tackled by the so-called localized cumulative distribution LCD introduced in [HK08]. For a given random vector $\tilde{\mathbf{x}} \in \mathbb{R}^N, N \in \mathbb{N}$ and the corresponding probability density function $f : \mathbb{R}^N \to \mathbb{R}_+$, the respective LCD $F(\mathbf{x}, \mathbf{b})$ is defined as

$$F(\mathbf{x},\mathbf{b}) := P\left(|\tilde{\mathbf{x}} - \mathbf{x}| \leq \frac{1}{2}\mathbf{b}\right), \quad (4)$$
$$F(\cdot,\cdot) : \Omega \to [0,1], \Omega \subset \mathbb{R}^N_+ \times \mathbb{R}^N_+, \mathbf{b} \in \mathbb{R}^N_+,$$

with $\mathbf{x} \leq \mathbf{y}$, $\mathbf{x}, \mathbf{y} \in \mathbf{R}^N_+$ denoting a component-wise relation that only holds if $\forall j \in [1, \ldots, N] : x_i \leq y_i$. The LCD $F(\mathbf{x}, \mathbf{b})$ can be imagined as an integral transform with the rectangular kernel $\mathbf{b}$ providing the limits of the integration for the different dimensions. Based on the probability density function $f(\mathbf{x})$ corresponding to $\tilde{\mathbf{x}}$, the respective LCD $F(\mathbf{x}, \mathbf{b})$ is calculated by

$$F(\mathbf{x},\mathbf{b}) = \begin{cases} \int_{\mathbf{x}-\frac{1}{2}\mathbf{b}}^{\mathbf{x}+\frac{1}{2}\mathbf{b}} f(\mathbf{t})\,d\mathbf{t} & \text{, if } \tilde{\mathbf{x}} \text{ continuous,} \\ \sum_{\mathbf{t}=\max\{\mathbf{0},\mathbf{x}-\lfloor\frac{1}{2}\mathbf{b}\rfloor\}}^{\min\{\mathbf{x}_\mathrm{max},\mathbf{x}+\lfloor\frac{1}{2}\mathbf{b}\rfloor\}} f(\mathbf{t}) & \text{, if } \tilde{\mathbf{x}} \text{ discrete,} \end{cases} \quad (5)$$

with $\mathbf{0} = (0, \ldots, 0)^\mathrm{T}$ denoting the zero vector, $\mathbf{x}_\mathrm{max}$ denoting the vector representing the upper limit of the spatial support and $\max\{\mathbf{x}\}$, $\min\{\mathbf{x}\}$ and $\lfloor\mathbf{x}\rfloor$ denoting elementwise operations. In contrast to the conventional definition of the cumulative distribution function, the LCD of a multivariate random vector is unique. Since this paper is especially focused on the case of discrete random vectors, the discrete case of Eq. (5) will be considered in further sections.

3.2.1 *Complexity of localized cumulative distribution evaluation*

In order to determine the computational complexity of one evaluation of the LCD $F(\mathbf{x}, \mathbf{b})$ for a given probability density function of a discrete random vector $\tilde{\mathbf{x}} \in \mathbb{R}^N$, the kernel vector $\mathbf{b}$ is considered to hold the same value in all dimensions, i.e., $\mathbf{b} = (b, \ldots, b)^\mathrm{T}$, what is always the case for its usage in the context of the generalized Cramér-von Mises distance. Since evaluating $F(\mathbf{x}, \mathbf{b})$ requires b array accesses and summation operations along each dimension, the corresponding complexity is $O(b^N)$ and hence $O(b^2)$ for the two-dimensional case common in image processing applications.

3.3 Generalized Cramér-von Mises Distance

Based on the introduced localized cumulative distribution (4), the generalized Cramér-von Mises distance between two multivariate random vectors can now be defined employing their LCDs as shown in [HK08]. For the LCDs $F(\mathbf{x}, \mathbf{b}), G(\mathbf{x}, \mathbf{b})$ corresponding to the probability density functions $f(\mathbf{x}), g(\mathbf{x})$ of two random vectors $\tilde{\mathbf{x}}, \tilde{\mathbf{y}} \in \mathbb{R}^N, N \in \mathbb{N}$, their generalized Cramér-von Mises distance is given by

$$D(f,g) := \int_{\mathbb{R}^N} \int_{\mathbb{R}^N_+} (F(\mathbf{x},\mathbf{b}) - G(\mathbf{x},\mathbf{b}))^2 \, d\mathbf{b}\, d\mathbf{x}. \quad (6)$$

In the case of discrete random vectors, the integrals are replaced by summations resulting in:

$$D(f,g) = \sum_{\mathbf{x}\in\Omega_s} \sum_{b=0}^{b_\mathrm{max}} \left(F(\mathbf{x},(b,\ldots,b)^\mathrm{T}) - G(\mathbf{x},(b,\ldots,b)^\mathrm{T})\right)^2, \quad (7)$$

with Ω_s denoting the spatial support of the probability density functions and $b_{\max}$ representing the absolute maximum component value of Ω_s, i.e., the maximum kernel size necessary to capture the whole probability density function.

3.3.1 Complexity of generalized Cramér-von Mises distance calculation

This section deals with the computational complexity of the calculation of the discrete CVM distance (7). As it is the most important case for image processing applications, the CVM distance for two discrete two-dimensional random vectors $\tilde{\mathbf{x}}, \tilde{\mathbf{y}} \in [0,\dots,n-1]^2$ and the corresponding probability density functions f, g is considered and the resulting complexity is determined in successive steps. As shown in Sec. 3.2.1, one evaluation of the LCD $F(\mathbf{x},\mathbf{b})$ has a complexity of $O(b^2)$ and thus every loop of the inner summation of Eq. (7) also has a complexity of $O(b^2)$. Since b is increased by 1 from 0 to $n-1$, the computation of the whole inner sum of Eq. (7) requires a number of

$$\sum_{b=0}^{n-1} b^2 = \frac{n^3}{3} - \frac{n^2}{2} + \frac{n}{6} \tag{8}$$

computations and hence has a complexity of $O(n^3)$. Conclusively, as all these computations have to be performed over the whole spatial support of the underlying probability density functions, i.e., a total of n^2 times, the overall complexity of the computation of $D(f,g)$ is in $O(n^5)$:

$$D(f,g) = \underbrace{\sum_{\mathbf{x}\in\Omega_s} \underbrace{\sum_{b=0}^{n-1} \overbrace{\left(F(\mathbf{x},(b,b)^{\mathrm{T}}) - G(\mathbf{x},(b,b)^{\mathrm{T}})\right)^2}^{\in O(b^2)}}_{\in O(n^3)}}_{\in O(n^5)} . \tag{9}$$

4 FAST GENERALIZED CRAMÉR-VON MISES DISTANCE

For the case of two two-dimensional discrete random vectors, the calculation of the generalized CVM distance can be accelerated. Therefore, the concept of summed area tables is employed in the evaluation of the localized cumulative distribution (5). Since for the referenced case, equation (5) represents a summation over a rectangular region, a summed area table can be precomputed for $f(\mathbf{x})$ in order to speed up the evaluation of $F(\mathbf{x},\mathbf{b})$. Algorithm 1 lists the pseudo code for the proposed fast calculation of the generalized CVM distance. After building the summed area tables $\mathfrak{f}, \mathfrak{g}$ for the discrete input probability density functions f and g, the squared difference between the localized cumulative distributions of f and g are computed for every spatial position $(i,j)^{\mathrm{T}}$ and for every sensible kernel size $b \in [0,\dots,n-1]$. The LCDs are evaluated by Algorithm 2 that employs Eq. (3) in concert with the provided summed area table $\mathfrak{s}$, the spatial position $(i,j)^{\mathrm{T}}$ and the kernel size $(b,b)^{\mathrm{T}}$ to obtain the summation of the values of the probability density function inside the respective rectangle.

Algorithm 1 Fast algorithm for calculating the generalized Cramér-von Mises distance for two two-dimensional discrete random vectors $\tilde{\mathbf{x}}, \tilde{\mathbf{y}}$ represented by their corresponding probability density functions $f(\mathbf{t}), g(\mathbf{t})$.

```
fastGCVM(f(x), g(x))
𝔣 ← generateSummedAreaTable(f(x))
𝔤 ← generateSummedAreaTable(g(x))
D ← 0
for i = 0,...,n−1 do
    for j = 0,...,n−1 do
        for b = 0,...,n−1 do
            D ← D+
                (fastLCD(𝔣, (i,j)^T, (b,b)^T) −
                 fastLCD(𝔤, (i,j)^T, (b,b)^T) )^2
        end for
    end for
end for
return D
```

Algorithm 2 Algorithm for evaluating the localized cumulative distribution $LCD((i,j)^{\mathrm{T}},(b,b)^{\mathrm{T}})$ of a two-dimensional discrete random vector at position $(i,j)^{\mathrm{T}}$ with kernel sizes $(b,b)^{\mathrm{T}}$ based on the summed area table $\mathfrak{s}$ of the underlying probability density function.

```
fastLCD(𝔰, (i,j)^T, (b,b)^T)
x_f ← max{0, i − ⌊½b⌋}
x_t ← min{x_max, i + ⌊½b⌋}
y_f ← max{0, j − ⌊½b⌋}
y_t ← min{y_max, j + ⌊½b⌋}
return 𝔰(x_t, y_t) − 𝔰(x_f − 1, y_t) − 𝔰(x_t, y_f − 1)
       +𝔰(x_f − 1, y_f − 1)
```

4.1 Complexity of fastGCVM Algorithm

The fastGCVM algorithm shown in the listing Algorithm 1 has three nested loops with n iterations each. The inner loop calculates a squared difference between the results of two evaluations of the fastLCD algorithm shown in listing Algorithm 2. Since fastLCD requires only a constant number of max, min, array lookups and arithmetic operations, it has a constant complexity of $O(1)$. Consequently, the complexity of the inner loop

Table 1: Execution times resulting from the performed experiment. The size of the input histograms is denoted by n and t_{naiv}, t_{fast} represent the mean measured execution times in milliseconds $\pm$ standard deviation for the naive, respectively, the fast algorithm.

n =	10	20	30	40	50	60	70	80	90	100
t_{naiv} in ms	7.2 $\pm$ 0.6	183.5 $\pm$ 0.3	1,311 $\pm$ 2	5,379 $\pm$ 4	16,145 $\pm$ 36	49,379 $\pm$ 138	105,820 $\pm$ 218	201,940 $\pm$ 1,550	375,010 $\pm$ 1,433	634,180 $\pm$ 390
t_{fast} in ms	0.23 $\pm$ 0.02	0.75 $\pm$ 0.01	2.18 $\pm$ 0.01	4.96 $\pm$ 0.02	9.7 $\pm$ 0.3	16.29 $\pm$ 0.06	25.7 $\pm$ 0.1	38.35 $\pm$ 0.08	54.6 $\pm$ 0.2	75.1 $\pm$ 0.1

of Algorithm 1 is in $O(1)$ too. As the three loops run n iterations each, the algorithm fastGCVM has a total complexity of $O(n^3)$.

5 EXPERIMENTS

Two experiments have been designed in order to demonstrate the effectiveness of the proposed speed-up technique. In the first experiment, multiple pairs of two-dimensional histograms with numbers of bins ranging from 10^2 to 100^2 have been randomly generated and used as the discrete input probability density functions. Between each pair, the generalized Cramér-von Mises distance has been calculated using a naive implementation of Eq. 7 and the proposed fastGCVM method shown in listing Algorithm 1. The algorithms have been implemented in C# using the Accord.NET framework [Sou14] and by not making use of any parallelism. The measured execution times are listed in Table 1. The results clearly show that the fast algorithm fastGCVM outperforms the naive implementation even for the case of small problem instances. For the largest employed example histogram with $100 \times 100 = 10,000$ bins, the naive implementation requires more than 10 minutes for calculating the desired result whereas fastGCVM needs only about 75 milliseconds.

For the second experiment, the code of an existing application has been extended by the fastGCVM algorithm. In [MLB16a, MLB16b], so-called light deflection maps are processed in order to visually inspect transparent objects for material defects. Deflection maps are spatially resolved data structures similar to discrete histograms containing information about the angles by which light rays get deflected while propagating through a transparent test object. Strong spatial discontinuities between adjacent deflection maps provide an indication of present scattering material defects that lead to changes in the distribution of the light's propagation direction. Such discontinuities cause high values of the generalized Cramér-von Mises distance between spatially adjacent deflection maps. In the context of the second experiment, the execution time was measured that has been required for processing the deflection maps [MLB16a] of a transparent cylindrical lens using both the naive and the fast implementation. For each input data set, the generalized CVM distance had to be calculated 3042 times between histograms having $9 \times 9 = 81$ bins. Figure 2 shows the resulting inspection images in pseudo colors. The naive implementation of the generalized CVM had an execution time of 15,750 ms $\pm$ 2 ms and the fastGCVM algorithm finished after 180 ms $\pm$ 2 ms.

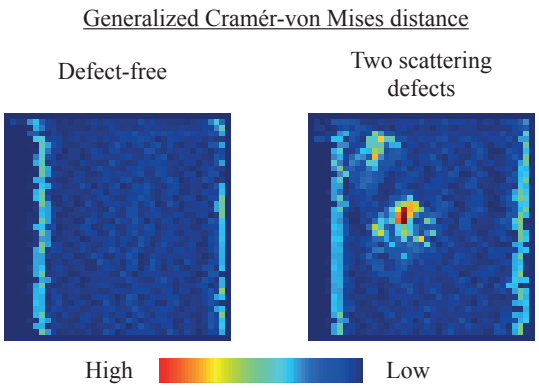


Figure 2: Pseudo color inspection images for plano-convex cylindrical lenses resulting from calculating the generalized Cramér-von Mises distance for spatially adjacent deflection maps. The left image corresponds to a defect-free test object instance and the right image corresponds to a test object instance affected by two scattering surface defects. The two defects are clearly indicated by the two regions of higher intensities in the image's upper left corner.

In summary, the experiments show that using the naive implementation of the Cramér-von Mises distance is not suitable for any practical application where execution time plays a critical role. Especially for visual inspection systems—as shown in the second example—which often have to fulfill real-time requirements, the fastGCVM algorithm allows to employ the generalized Cramér-von Mises distance in the image processing pipeline due to its reduced computational complexity.

6 SUMMARY

The generalized Cramér-von Mises distance is a helpful tool for comparing multivariate random vectors. However, for the frequent case of two discrete two-dimensional random vectors $\tilde{\mathbf{x}}, \tilde{\mathbf{y}} \in [0, \ldots, n-1]^2$, $n \in \mathbb{N}$, the naive implementation of the CVM distance

has a computational complexity of $O(n^5)$, what leads to execution times impractical for many applications. After introducing the reader to the idea of summed area tables, which is a common speed up technique in the domain of image processing, the generalized Cramér-von Mises distance and the underlying concept of localized cumulative distributions have been introduced. The paper then proposes to employ summed area tables in order to obtain fastGCVM, a fast algorithm for calculating the generalized Cramér-von Mises distance with a reduced computational complexity of only $O(n^3)$. By means of two experiments it could be shown that fastGCVM clearly outperforms the naive implementation of the generalized CVM distance—in some cases, fastGCVM's execution time is four magnitudes lower than the time required by the naive implementation. It should be mentioned, that the execution time of fastGCVM can be further reduced by adequately employing simple parallelization techniques. As further steps, the authors plan to provide their implementation as an open source library and to theoretically show, how summed area tables can be used to also speed up the calculation of the generalized CVM for higher dimensional discrete random vectors.

7 REFERENCES

[Cha07] Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007.

[Cor09] Thomas H Cormen. *Introduction to algorithms*. MIT press, 2009.

[Cro84] Franklin C. Crow. Summed-area tables for texture mapping. *ACM SIGGRAPH computer graphics*, 18(3):207–212, 1984.

[CS02] Sung-Hyuk Cha and Sargur N. Srihari. On measuring the distance between histograms. *Pattern Recognition*, 35(6):1355–1370, 2002.

[DNK03] Dietrich Van der Weken, Mike Nachtegael, and Etienne Kerre. Using similarity measures for histogram comparison. In *Fuzzy Sets and Systems-IFSA 2003*, pages 396–403. Springer, 2003.

[GH13] Igor Gilitschenski and Uwe D. Hanebeck. Efficient deterministic dirac mixture approximation of Gaussian distributions. pages 2422–2427. IEEE, 2013.

[HHK09] Uwe D. Hanebeck, Marco F. Huber, and Vesa Klumpp. Dirac mixture approximation of multivariate gaussian densities. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 3851–3858. IEEE, 2009.

[HK08] Uwe D. Hanebeck and Vesa Klumpp. Localized Cumulative Distributions and a Multivariate Generalization of the Cramér-von Mises Distance. In *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, pages 33–39. IEEE, 2008.

[MGW10] Yu Ma, Xiaodong Gu, and Yuanyuan Wang. Histogram similarity measure using variable bin size distance. *Computer Vision and Image Understanding*, 114(8):981–989, August 2010.

[MLB16a] J. Meyer, T. Längle, and J. Beyerer. Acquiring and processing light deflection maps for ransparent object inspection. In *2016 2nd International Conference on Frontiers of Signal Processing (ICFSP)*, pages 104–109, Oct 2016.

[MLB16b] Johannes Meyer, Thomas Längle, and Jürgen Beyerer. About the acquisition and processing of ray deflection histograms for transparent object inspection. In *IRISH MACHINE VISION & IMAGE PROCESSING Conference proceedings*, 2016.

[MS08] Kurt Mehlhorn and Peter Sanders. *Algorithms and data structures: The basic toolbox*. Springer Science & Business Media, 2008.

[PHB13] Alexey Pak, Marco F. Huber, and Andrey Belkin. On weak distance between distributions in application to tracking. In *Sensor Data Fusion: Trends, Solutions, Applications (SDF), 2013 Workshop on*, pages 1–6. IEEE, 2013.

[Sou14] César Souza. The Accord.NET Framework. `http://accord-framework.net`, 2014. Accessed: March 2017.

[VJ04] Paul Viola and Michael J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

MAELab: a framework to automatize landmark estimation

LE Van Linh
LaBRI-CNRS 5800
ITDLU, Dalat Univ-V
linhlv@dlu.edu.vn/
van-linh.le@labri.fr

BEURTON-AIMAR Marie
LaBRI-CNRS 5800
Bordeaux University
33400 Talence-F
beurton@labri.fr

KRAHENBUHL Adrien
LaBRI-CNRS 5800
Bordeaux University
33400 Talence-F
adrien.krahenbuhl@labri.fr

PARISEY Nicolas
IGEPP
INRA 1349
35653 Le Rheu-F
nparisey@rennes.inra.fr

Abstract

In biology, the morphometric analysis is widely used to analyze the inter-organisms variations. It allows to classify and to determine the evolution of an organism's family. The morphometric methods consider features such as shape, structure, color, or size of the studied objects. In previous works [8], we have analyzed beetle mandibles by using the centroid as feature, in order to classify the beetles. We have shown that the Probabilistic Hough Transform (PHT) is an efficient unsupervised method to compute the centroid. This paper proposes a new approach to precisely estimate the landmark geometry, points of interest defined by biologists on the mandible contours. In order to automatically register the landmarks on different mandibles, we defined patches around manual landmarks of the reference image. Each patch is described by computing its SIFT descriptor. Considering a query image, we apply a registration step performed by an Iterative Principal Component Analysis which identify the rotation and translation parameters. Then, the patches in the query image are identified and the SIFT descriptors computed. The biologists have collected 293 beetles to provide two sets of mandible images separated into left and right side. The experiments show that, depending on the position of the landmarks on the mandible contour, the performance can go up to 98% of good detection. The complete workflow is implemented in the MAELab framework, freely available as library on GitHub.

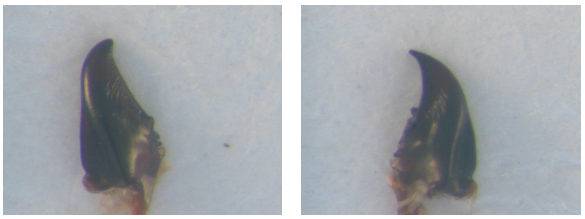
Keywords

Morphology, image registration, SIFT descriptor, beetle, mandible.

1 INTRODUCTION

Phenotype of beetle species is characterized by information like age, sex, morphological criteria or environmental parameters. Biologists are used to proceed to manual measurements in case of analysis at macro level as for example tissues or animal members [6] [3]. They can directly measure the geometrical characteristics of elements on the body of the animal: length, width, diameter, angles, etc. Another way to obtain morphological measures is to take pictures of the members and to apply image processing algorithms. In order to evaluate a population of beetles from Brittany lands, a collection of 293 beetles has been established. For each beetle, biologists took images of the left and right mandibles (see Fig. 1) and a set of landmarks has been manually determined by experts. A morphometric landmark is a specific point, directly linked to the animal anatomy.

The landmarks are used in many domains, not only in biological studies. It is an application field of image processing [9] that appears in works of computer vision, mainly in face recognition [15] but also in human orthodontic [5] or morphometric analysis [1].



(a) Left mandible (b) Right mandible

Figure 1: The two mandibles of a beetle captured by biologists.

In this paper, we focus on the automatic identification of landmarks in 2D mandible images. The proposed method consists of three main steps: a segmentation of the mandible based on the Canny algorithm, an Iterative Principal Component Analysis to register a query image on a model image, and finally a landmark estimation on the query image by comparison of SIFT descriptors. Section 2 presents the complete workflow then section 3 details the experiments and analyzes the results.

2 METHOD

The addressed problem is the automatic detection of morphologic landmarks on mandible pictures to replace the manual operation made by an expert operator. We detail hereafter a workflow (resumed Fig. 2) including (1) the segmentation of a query image, (2) a registration step on a model image then (3) the detection of landmark positions. It is worth to note that all pictures have been taken in the same conditions with the same camera at the same resolution. Moreover, the model image is randomly chosen from the set of all images.

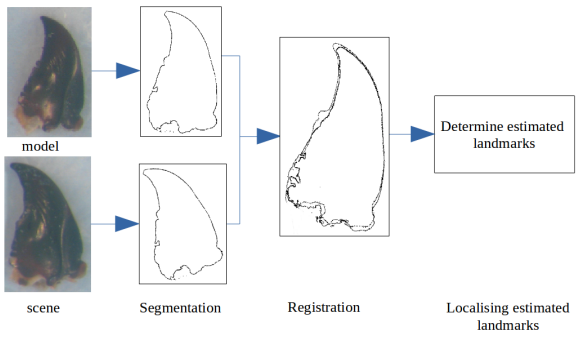


Figure 2: Overview of the proposed method.

2.1 Image segmentation

The segmentation step is the first task of a large number of image processing chains. A contour-based algorithm, the Canny algorithm [4], has been chosen to extract the contour belonging to the shape of the mandible. To use this method, two threshold values have to be set. As it is mentioned in [14], determine the right thresholds could be difficult. The mandatory *threshold value* is determined by analyzing the image histogram (see [8] for details). Most often authors define these thresholds as a lower and an upper one with a usual ratio of $T_{lower} = (1/2) \times T_{upper}$. In order to consider a larger range of values, we defined $T_{lower} = 1/3 \times T_{upper}$. Note that to optimize the computing time, the direction of the gradient of each pixel belonging to the mandible contour is also computed during the Canny algorithm but will be used later (see Sec. 2.3). To obtain the segmentation of the mandible, the contours obtained with Canny are discriminated to only keep the mandible contours. As shown in Fig. 3, the Canny algorithm generates some contours which do not belong to the mandible shape. A simple algorithm parses the contour image to suppress the edges inside the biggest contour.

(a) Contours identified by the Canny algorithm.

(b) Contour selected by post-processing.

Figure 3: Detection of the mandible contours.

2.2 Image registration

As previously mentioned, all images have been captured at the same scale. However, the mandible size can vary from a beetle to another one, as their orientation and position can differ from a picture to another one. This point is taken into account in this registration step from a query image (the scene) to a reference image (the model).

We have chosen to apply a method based on the Principal Component Analysis (PCA) [12, 13] to determine the rotation and translation parameter between the two images. In input, we consider the two lists of contour points defined from the segmentation of the two images. Firstly, the centroid and the principal axis of each image are computed. The centroid corresponds to the mean point of all contour points. The principal axis is the line connecting the centroid to a contour point, determined with algorithm 1.

Algorithm 1: Algorithm to find the principal axis of a list of contour points

Input : Centroid c, list of contour points l
Output: Principal axis a

```
for all points p_i in l do
    for all points p_j in l do
        if p_i ≠ p_j then
            Compute the orthogonal distance d_ij
            between the line (c, p_i) and p_j.
        end
    end
    Compute d_mean as the average distance of all d_ij
    distances.
    if d_mean is minimal then
        p_min = p_i;
    end
end
The principal axis is: a = (c, p_min).
```

The translation between the scene image and the model image is computed from the distance between their centroids. The rotation is computed from the angle between the principal axes of these two images. Translation then rotation operations are applied to register the

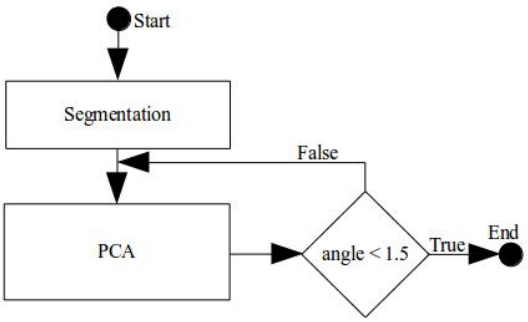


Figure 4: Workflow of the PCAI, allowing to refine the rotation angle between the scene and the model.

scene on the model. However, the translation and rotation can be imprecise when the result of the segmentation contains noise. In order to prevent these cases, we remarked and used the image specificity that implies that the tip of the mandible is less noisy than its base. So, we have sorted the contour points according to their y-value to build the subset containing the half upper part of contour points. Then, we have enhanced the PCA by an Iterative process on this subset until stabilization (PCAI). At each iteration, the rotation value is refined. The procedure stop when the new computing angle is lower than 1.5 degrees (see Fig. 4).

The Fig. 5 shows an example of the successive results obtained with PCAI. The red contour belongs to the model, the black one corresponds to the scene contour registered according to the angle find after one iteration, and the blue contour is the final contour obtained at the end of the PCAI process.

Figure 5: Iterations of the registration step between the model contour (in red) and the contours of the scene image.

2.3 Landmark detection with SIFT

The last step of the workflow consists of estimating the position of the landmark in scene image from the manual ones of the model. We relied on the SIFT method [10] that has been used in a lot of computer vision methods to identify points of interests. We modified some aspects of the initial method to defined a process specialized to the landmark identification. In order to reduce the computing time and the possible errors of location, we do not consider all points of the image but only the area around each landmark on the model. Firstly, the patch around each landmark of the model is computed and extracted at the same position in the scene image. Then, the SIFT descriptor is computed. The orientation and the gradient magnitude are calculated for each pixel by using the gradient values computed during contour detection (see Sec. 2.1) by applying the classical equations 1:

$$\begin{aligned} m_I(x,y) &= \sqrt{v_x^2+v_y^2} \\ \theta_I(x,y) &= tan^{-1}(v_y/v_x) \end{aligned} \tag{1}$$

Where:

- $v_x = I(x+1,y) - I(x-1,y)$
- $v_y = I(x,y+1) - I(x,y-1)$
- $I(x,y)$ is the intensity of I at position (x,y),
- $m_I(x,y)$ is the gradient magnitude of in I at position (x,y),
- $\theta_I(x,y)$ is the orientation in I at position (x,y).

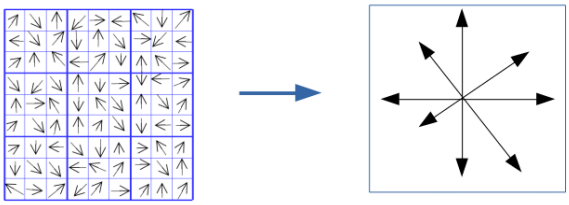


Figure 6: The SIFT descriptor of a patch. In the right figure, the arrow length corresponds to the gradient value.

For each patch, the SIFT descriptor is the histogram of the sum of gradients of each considered direction. As usually, eight direction classes are considered: $[0^o - 45^o[$, $[46^o - 90^o[$, $[91^o - 135^o[$, $[136^o - 180^o[$, $[181^o - 225^o[$, $[226^o - 270^o[$, $[271^o - 315^o[$, $[316^o - 360^o[$). The feature vector is normalized to reduce the effects of illumination changes.

The Fig. 6 shows a patch of 9×9 pixels centered in each landmark on the model. The size of 9×9 has been retained after several tests where patch sizes 18×18, 36×36 and 54×54 have given unsatisfactory results. From the histogram, we obtain the local gradient value for each direction.

The comparison between two SIFT descriptors is done by using the $L2$-distance with the following equation (2):

$$L(D1,D2) = \sum_{i=0}^{n} \sqrt{(D1_i - D2_i)^2} \tag{2}$$

Where:

- n is the number of directions
- $D1$ and $D2$ are two descriptors of size n,
- $D1_i$ and $D2_i$ are the i^{th} descriptor value.

The Fig. 7 illustrates how we have applied the SIFT method into our workflow. IN order to detect the scene landmarks, the patches P_m of the model and P_s of the scene are initialized with the size of P_m smaller than the size of P_s. After experiments, we have kept 36×36 pixels as the size of P_s. For each pixel in the patch P_s, a sub-patch P'_s is extracted with the same size than P_m. When P'_s have a part outside P_s, the outside pixels are also considered. Then, the distance $L(P_m, P'_s)$ is computed using equation (2). The estimated position of the landmark corresponds to the position of the sub-patch P'_s with the smallest distance L to P_m. Finally, the position of the estimated landmarks are positioned in the original scene image by applying the reverse operations of rotation and translation computed in registration step (see Sec. 2.2).

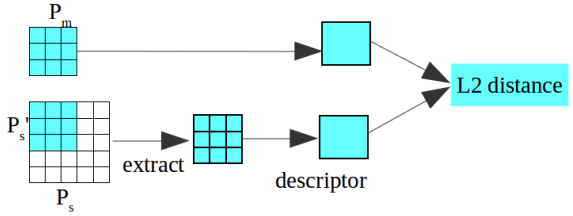


Figure 7: Comparison of descriptors between the patch P_m of the model image and the patches P'_s of the scene image.

3 EXPERIMENTS AND RESULT

The complete method is implemented in the framework MAELab[1]. The left and the right mandibles of the beetles has been analyzed separately. After verifying the quality of the image, it remains 290 usable images of right mandibles and 286 images of left mandibles. The removed images include the images without mandible or with broken mandibles. In all valid images, a set of manual landmarks is indicated by biologists: 18 for right mandibles, 16 for left mandibles, which constitutes our ground truth.

We have run the full workflow on all the usable images. The results have shown differences in algorithm accuracy: estimated landmarks are well positioned on some scene images but not satisfying on others. As we mentioned before, mandibles images can exhibit different sizes because beetles have also different sizes of mandible. We detected that our method is sensible to this parameter. To improve the results, we have inserted a pre-processing step to estimate the scale of a scene image and the model before the computing of the SIFT descriptors. The bounding boxes of the mandible of the model image and the scene image are computed and the scales in the x- and y-directions are determined by the ratio between the corresponding sides of the bounding boxes. Then, the scene contours are rescaled to fit the model contours.

[1] MAELab is a free software written in C++. It can be directly and freely obtained by request at the authors.

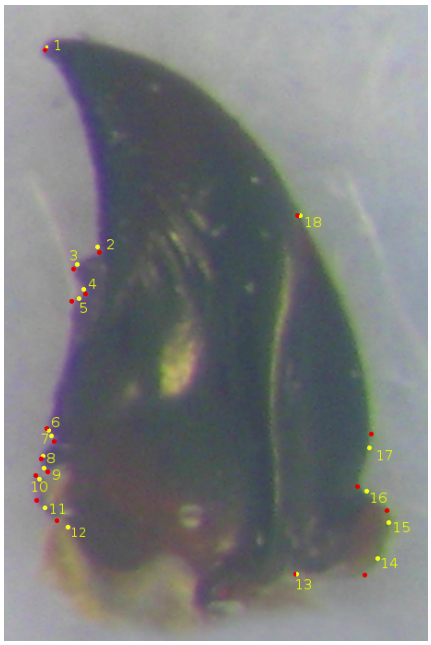


Figure 8: The manual (in red) and estimated (in yellow) landmarks on a right mandible.

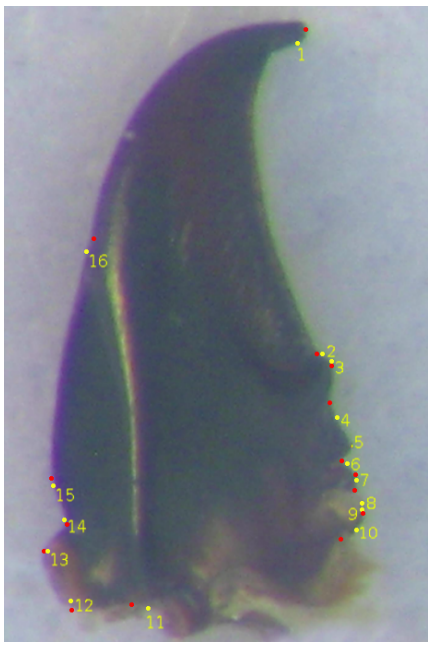
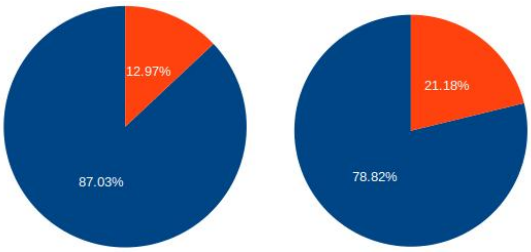


Figure 9: The manual (in red) and estimated (in yellow) landmarks of a left mandible.

Figs. 8 and 9 show the final results for a right and a left mandible with the manual and estimated landmarks. The estimated landmarks are quite near with the manual landmarks, as it is shown in the following statistical evaluation.

The statistics have been computed for all landmarks of the scene images. We have compared the positions between the manual and estimated landmarks by accepting an error from 1% to 2% of the bounding box's size. According to this way, a global statistic compares all pairs of corresponding landmarks on all images as presented in Fig. 10. It shows the global results with a score of well-positioned landmarks equal to **87.03**% for right mandibles and **78.82**% for left mandibles.



(a) Set of right mandibles. (b) Set of left mandibles.

Figure 10: The mean proportion of well and bad landmark locations of the two sets of left and right mandibles.

Besides the global results, we are also interested by the accuracy of the individual positions of the estimated landmarks. We have computed the distance between the manual landmarks and their corresponding estimated landmarks in order to examine the proportion of well-positioned landmarks. The Fig. 11 and 12 show the proportion of well-estimated landmarks for each landmark of the model. With 18 landmarks of right mandible, the position of the 1^{st} estimated landmarks is very accurate with **98.62**%. The lowest proportion is **74,48**% for the 14^{th} landmark. The remaining landmarks are also estimated with an accuracy greater than 75%. For left mandibles, the highest and lowest success rates are **93,01**% for the 1^{st} landmark and **60,14**% for the 16^{th} landmark. The statistic is done on each estimated landmark of all the images with a standard deviation error [2]. As we can see in Fig. 3, the noise of the contour part located at the base of a mandible is higher than the noise located at the tip of the mandible. This explains why the correct proportion on 11^{th} and 12^{th} landmarks of the left mandible and 13^{th} and 14^{th} landmarks of the right one are less accurate than other landmarks. Moreover, when we reconsider the datasets, the left mandible images have bigger scale values than the right mandible images. This could explain that the success rate of the right mandibles is always greater than this one of the left in all experiments.

In a previous work, we have tried to apply a set of procedures coming from an article of Palaniswamy [11] who tried to find automatically a specific point of interest into a Drosophila wing. We have succeeded to fix the centroid of the mandibles by using these procedures (mainly based on the computation of a Probabilistic Hough Transform accumulator). But this way has not been enough efficient to set precisely the landmarks. D. Houle et al [6] have more recently described a method to estimate automatically the landmarks on Drosophila wings (with 12 landmarks). This method is mainly based on the use of a curve analysis (with splines) belonging to the wing shape. The method has been evaluated on 535 wing images. The average proportion of all 12 points is 82%. They have been able to improve their results by suppressing the least accuracy point (47% of right results) that leads to a better parameter fitting. Y. Ke et al. [7] have proposed to combine SIFT descriptor with PCA analysis to characterize images belonging to a Graffiti dataset. They also obtained good performances close to 95% of correct results. The results presented in this article can be considered as in the same order of correctness than these works, but it concerns a problem, precise fixing of a lot of landmarks, more difficult to solve. One can note that this chain of treatments dedicated to the estimation of landmarks on 2D images of mandibles is from now, user-friendly available.

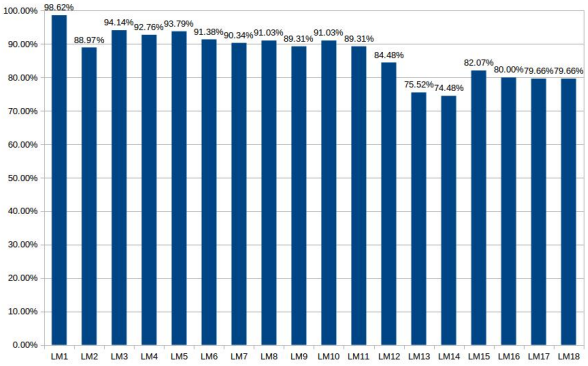


Figure 11: The proportion of well estimated landmarks of right mandibles.

4 CONCLUSION

The morphometric analysis is a powerful tool to analyze and to classify species. In this paper, we have designed a method to segment the beetle mandibles and to automatically locate landmarks which have been determined manually on a model image, by biologists. Each mandible has been segmented by using the Canny algorithm before to be registered using PCAI to align the images. The estimation step of the landmark position uses the SIFT descriptor to find the best matching position. The results show that the method succeeds in locating the landmarks for all images. The accuracy of the method is sufficient to be proposed to biologists as

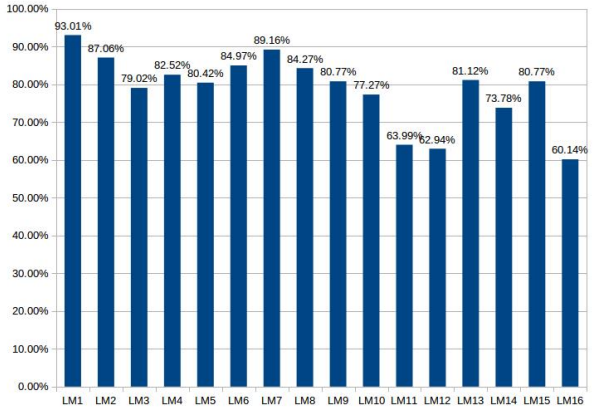


Figure 12: The proportion of well estimated landmarks of left mandibles.

an alternative to the manual measures. Moreover, considering the previous work in [8], this method reduces the drastically the number of outlier landmarks and the MAELab implementation also reduce the global computing times and memory cost. From now, the next stage consists of improving the registration step in order to increase the matching step accuracy and completely remove manual interventions. For example, we could investigate deep learning methods, more precisely Convolutional Neural Networks computing, which has risen up in image processing recently. Biologists are interested in the large-scale analysis of their species collections, automatic classification is one of the bottlenecks to solve towards a better integration of informatic procedures in their current way to work.

5 REFERENCES

[1] José Maria Becerra and Antonio G Valdecasas. Landmark superimposition for taxonomic identification. *Biological Journal of the Linnean Society,*, 81:page 267–274, 2004.

[2] J Martin Bland and Douglas G Altman. Statistics notes: measurement error. *Bmj*, 313(7059):744, 1996.

[3] Paul A Bromiley, Anja C Schunke, Hossein Ragheb, Neil A Thacker, and Diethard Tautz. Semi-automatic landmark point annotation for geometric morphometrics. *Frontiers in Zoology*, 11(1):61, 2014.

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

[5] Leila Favaedi and Maria Petrou. Cephalometric landmarks identification using probabilistic relaxation. In *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, pages 4391–4394. IEEE, 2010.

[6] David Houle, Jason Mezey, Paul Galpern, and Ashley Carter. Automated measurement of drosophila wings. *BMC evolutionary biology*, 3(1):25, 2003.

[7] Yan Ke and Rahul Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–II. IEEE, 2004.

[8] L Lê Vãnh, M Beurton-Aimar, JP Salmon, A Marie, and N Parisey. Estimating landmarks on 2d images of beetle mandibles. *WSCG*, 2016.

[9] Yunpeng Li, David. J Crandall, and Daniel P. Huttenlocher. Landmark classification in large-scale image collections. pages 1957–1964. Kyoto, Japan, 29 sept-2 oct 2009.

[10] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[11] Sasirekha Palaniswamy, Neil A Thacker, and Christian Peter Klingenberg. Automatic identification of landmarks in digital images. *IET Computer Vision*, 4(4):247–260, 2010.

[12] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901.

[13] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[14] Jun Zeng and Dehua Li. An adaptive canny edge detector using histogram concavity analysis. *International Journal of Digital Content Technology and Its Applications*, 5(6):172–181, 2011.

[15] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision*, pages 94–108. Springer, 2014.

Support Vector Machine Optimized by Firefly Algorithm for Emphysema Classification in Lung Tissue CT Images

Eva Tuba
Faculty of Mathematics,
University of Belgrade
Studentski trg 16,
11000 Belgrade, Serbia
etuba@acm.org

Milan Tuba
Faculty of Computer Sci.,
John Naisbitt University
Bulevar umetnosti 29,
11070 Belgrade, Serbia
mtuba@acm.org

Dana Simian
Faculty of Science
Lucian Blaga University,
Ion Ratiu Street 5-7,
550012, Sibiu, Romania
dana.simian@ulbsibiu.ro

ABSTRACT

Digital images and digital image processing facilitated significant progress in numerous areas where medicine is an important one of them. Computer-aided detection and diagnostics systems are used to assist specialists in interpretation of medical digital images. One of the important research issues is detection and classification of the chronic obstructive pulmonary disease in lung CT images. In this paper we proposed a method for emphysema classification based on texture and intensity features. Only six different characteristics of the uniform local binary pattern and intensity histogram were used as input vector for support vector machine that was used as classifier. Feature vector was significantly reduced compared to the other state-of-the-art methods while the classification accuracy was increased. On images from standard dataset global accuracy of our proposed algorithm was 98.18% compared to 95.24% and 93.9% of two other compared algorithms.

Keywords

Support vector machines, lung tissue classification, CT images, image processing, firefly algorithm, swarm intelligence

1 INTRODUCTION

Digital images and digital image processing have been proven to be of great use in medicine. Numerous sources have been used for gathering digital images of various tissues and organs, such as X rays, ultrasound, magnetic resonance imaging (MRI), computed tomography (CT), positron emission tomography (PET). Each of these image types have good and bad characteristics, thus they are all used for different organs, diseases and anomalies. Since MRI, CT, PET and other medical devices and procedures produce a large number of images that need to be analyzed in short time, computer-aided detection and diagnosis systems (CAD) are very useful to speed up the process, highlight suspicious parts, enhance image quality for easier diagnostic, etc.

One of the extensively studied topic is detection of chronic obstructive pulmonary disease (COPD). Usual method for early stage detection of COPD is by using CT imaging which enables detection of emphysema. Example of lung slice CT image is shown in Fig. 1.

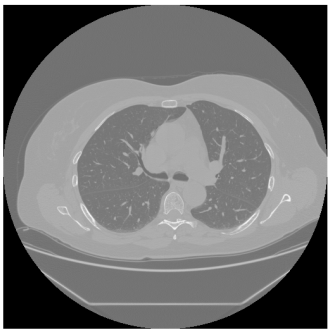
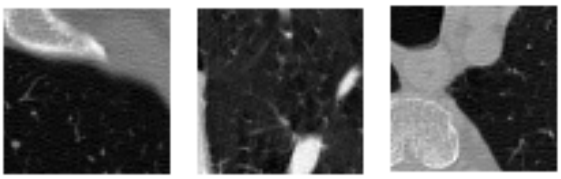


Figure 1: Lung CT slice

Emphysema stands for a long-term and progressive disease of the lungs that causes shortness of the breath. In CT images emphysemas can be recognized as regions close to the air with low attenuation values. Three different types of emphysemas can be recognized, centrilobular emphysema (CLE), paraseptal emphysema (PSE) and panlobular emphysema (PLE). Examples of normal tissue (NT), CLE and PSE are shown in Fig. 2.

In recent years different approaches were proposed for emphysema detection and classification. Some of the well known classifiers were used for labeling the types of the tissues such as k nearest neighbor [IM14], artificial neural networks [BSBY14], convolution neural



(a) NT - normal tissue (b) CLE - centrilobular emphysema (c) PSE - paraseptal emphysema

Figure 2: Examples of the normal tissue and different emphysemas

network [SSK15], [ACE+16], etc. Besides the classification method, feature selection is a very important part of the CAD system for emphysema classification, i.e. diagnosis determination. Numerous papers propose different texture features combined with the image intensity, while the other use some other features such as invariant moments, structural co-occurrence matrix and others.

In this paper, we proposed a method for emphysema lesion detection and classification. For classification support vector machine (SVM) was used and the lung tissues were described by uniform local binary pattern (LBP) and intensity histogram features. Only six different characteristics of the uniform local binary pattern and intensity histogram were used as input vector for support vector machine that was used as classifier. Feature vector was significantly reduced compared to the other state-of-the-art methods while the classification accuracy was increased.

The remainder of this paper is organized as follows. In Section 2 literature review on recent proposed methods for emphysema lesion classification are discussed. In Section 3 our proposed method is described. Experimental results along with comparison with other state-of-the-art methods are presented in Section 4. At the end, conclusion and possible further work are given in Section 5.

2 LITERATURE REVIEW

Computer-aided diagnosis systems were exhaustively studied in the recent years [TTD17]. In [DBF+13] a CAD system for breast cancer detection in mammographic images was proposed, while in [KY14] CAD for brain tumor detection in MRI images was developed. In [STT16] a method for brain MRI abnormality detection was proposed. Another computer-aided diagnosis systems from literature deal with Alzheimer's type dementia classification [RGSG+13], detection of Parkinson's disease [HAB15], cancer detection [LC15], etc. Different image processing techniques were used such as edge detection [NTT16], multilevel thresholding [Tub14], etc.

CAD for chronic obstructive pulmonary disease detection, especially emphysema, represents one of the important research topics. In general, there are two main steps in these systems: feature extraction and classification.

In [DSS15] features based on local diagonal extrema pattern (LDEP) for CT images were proposed. The relationship between central pixel and diagonal neighbors was described by using the first order local diagonal derivatives. Indexes and values of the local diagonal extremes were found and compared with the intensity of the central pixel. Features were formed based on these indexes and comparisons. Feature dimension was reduced by using only diagonal neighbors and it was shown that it represents an efficient feature descriptor.

A cascade method that combines multi-fractal features and alpha histograms descriptors was proposed in [IM15]. Multi-fractal features were useful to differentiate normal tissue and abnormalities while alpha histogram descriptor additionally improved classification accuracy. For classification naive Bayes classifier was used.

In [NERCE14] emphysema classification framework based on complex Gabor filters and local binary patterns was proposed. By combining these two features global characteristics as well as local information were described and additionally kernel Fisher analysis was used to reduce the size of feature vector. For classification k-nearest neighbor classifier was used and it was reported that combination of the mentioned descriptors represents effective features for emphysema classification.

Quality of a learning technique to classify emphysema based on embedded probabilistic principal component analysis (PPCA) was tested in [ZCKR+13]. The first step of the proposed method was to find the most discriminant linear space for each emphysema pattern against the remaining patterns where lung CT image patches can be embedded. PPCA model was trained for each pattern. Contribution of the mentioned paper is the ability to compute the class membership posterior probability for each emphysema pattern. Proposed method has shown competitive results against different texture-based approaches.

In [KWE14] five classes of increasing disease severity were considered. Three different methods were exploited for this multi-class classification problem. The first method used a global rankSVM for ranking, while hierarchical SVM was used for classification. Finally combination of these two classifiers was proposed and named hierarchical rankSVM. Results have shown that hierarchical approaches were computationally efficient and classification achieved accuracies were a little better in the case of hierarchical SVM.

Physician-in-the-loop feedback approach was proposed in [RRK+14] in aim to minimize ambiguity in the selected training samples. Multiple metrics for features

were used and seven unoptimized support vector models were built for each of them. The training samples were classified by using these seven SVM models. Label of the region of interest was determined by majority voting.

In [YFAL16] texton-based method for classification of emphysema was proposed. Lung textures were described by the nearest-texton frequency histograms. Experiments were done for characterizing lung textures with sparse decomposition from texton dictionaries while different regularization strategies were used. Additionally, the sparsity-inducing constraints were introduced to the construction of the dictionaries. Accuracy higher than 90% was reported.

In [KI15] convolution neural network (CNN) was proposed for emphysema classification. As input for the proposed classifier raw high resolution CT images of lung were used. It was reported that promising accuracy has been achieved. Another method that uses CNN was proposed in [Pei15]. Feature extraction was done by CNN as well as classification. It was reported that for differentiation of normal tissue and one type of emphysema high accuracy was achieved.

Local binary pattern was frequently used for emphysema description. In [NKOS15] comparative study of basic LBP and two variations, completed LBP (CLBP) and local ternary pattern (LTP), was presented. Joint histogram of density and texture histogram was used as input vector for linear SVM. Histograms were calculated for regions of different sizes and the best results were 81.36%, 82.99% and 83.29% for LBP, CLBP and LTP respectively. Another method that uses LBP was proposed in [SSdB10]. Combined histogram of the LBP and intensity histogram was used as input vector for adjusted k-nearest neighbor classifier. Classification accuracy was 95.4%.

This review shows that emphysema classification systems represent an important and active research area. In this paper we try to develop faster and more accurate algorithm based on fewer features, hence appropriate for further improvements.

3 OUR PROPOSED METHOD

In this paper we propose a computer-aided system for emphysema classification. The proposed method consists two main parts, feature extraction and classification. In the past it has been proven that different types of emphysema can be well distinguished by the texture features [SSdB10], [YFAL16]. We propose rotation invariant version of local binary pattern as a texture descriptor. For better classification results these features were combined with intensity features. Appropriate features are mandatory for high classification accuracy but accuracy also depends on the quality of the classifier. In this paper we propose very successful classifier, support vector machine, optimized by fireworks swarm intelligence algorithm.

3.1 Feature extraction

Local binary patterns (LBP) are well known texture feature descriptors. Originally, LBP were proposed as an invariant measure for local structure in 3×3 neighborhoods. Different modifications were proposed later and nowadays are used as a simple but good texture descriptor. Binary patterns are obtained by thresholding circular neighbor pixels according to the central pixel. LBP value of the central pixel (x,y) is defined by the following equation:

$$LBP_{P,R} = \sum_{p=0}^{P} s(I(x_p,y_p) - I(x,y))2^P \qquad (1)$$

where P is the number of neighbor pixels at distance R that will be considered, $I(x,y)$ is the intensity of the central pixel (x,y) and function s is defined as:

$$s(k) = \begin{cases} 1 & \text{if } k \geq 0, \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

Fig. 3 represents illustration of Eqs. 1 and 2.

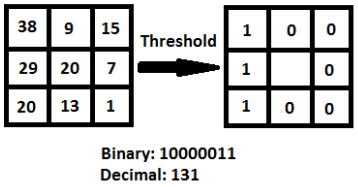


Figure 3: LBP

After obtaining the LBP which are actually binary codes, histogram is made. Size of the histogram is determined by the number of the possible binary codes which is defined by the number of considered neighbor pixels P.

One of the LBP characteristic is that it is invariant to any monotonic gray-scale transformation. As long as relative relation between pixel intensities is saved after transformations, LBP remains the same. LBP has been widely studied in the past thus numerous modifications of LBP can be found in literature. One of the highly desirable characteristic of texture descriptors is rotation invariance and various changes of LBP were proposed to achieve this. In this paper we used uniform LBP. In uniform LBP, number of the possible patterns is reduced. Instead of considering all possible patterns only those that contain less than two transitions from 0 to 1 (or 1 to 0) are used while all others are considered as the same pattern. If 8 neighbor pixels

were used for forming LBP ($P = 8$, $R = 1$), the number of patterns is reduced from 2^8 =256 to 59. In this paper, uniform local binary patterns were transformed into decade numbers and normalized histogram of occurrence frequencies was made.

Besides texture descriptor, LBP histogram, normalized intensity histogram was used. In [SSdB10] joint LBP and intensity histogram were also used and it has been proven that intensity information significantly increases classification accuracy.

Histograms are usually used as input feature vectors and since intensity histogram is a 256-dimensional vector and uniform LBP histogram is 59-dimensional vector the input vector would be 315-dimensional. Larger dimension of feature vectors can make classification more difficult. Instead of using complete histograms in this paper we propose to use just some histogram characteristics. Numerous metrics are used to provide useful information based on the histogram. In this paper we used six common metrics. For each histogram, uniform LBP and intensity histogram mean (μ), standard deviation (σ), skewness, kurtosis, energy and entropy were calculated and used as input vector. If normalized histogram is annotated by p and N represents the number of histogram values then these metrics can be calculated by the following equations:

$$\mu = \sum_{i=0}^{N} i * p(i) \tag{3}$$

$$\sigma = \sqrt{\sum_{i=0}^{N} (i-\mu)^2 * p(i)} \tag{4}$$

$$\text{skewness} = \frac{1}{\sigma^3} \sum_{\text{i}=0}^{\text{N}} (\text{i}-\mu)^3 * \text{p(i)} \tag{5}$$

$$\text{kurtosis} = \frac{1}{\sigma^4} \sum_{\text{i}=0}^{\text{N}} (\text{i}-\mu)^4 * \text{p(i)} - 3 \tag{6}$$

$$\text{energy} = \sum_{\text{i}=0}^{\text{N}} \text{p(i)}^2 \tag{7}$$

$$\text{entropy} = -\sum_{\text{i}=0}^{\text{N}} \text{p(i)} * \log_2 \text{p(i)} \tag{8}$$

By using these metrics the 12-dimensional feature vector is used (6 dimensions for the LBP histogram and 6 dimensions for the intensity histogram) instead of earlier mentioned 315-dimensional vector hence input vector for classifier is significantly reduced.

3.2 Classification

In this paper we used one of the widely used and very successful binary classifier: support vector machine (SVM). Emphysema classification represents multiclassification problem. In order to use binary classifier for multiclassification two method were proposed in literature - *one against one* and *one against all*. In this paper we used *libsvm* package for building SVM models where *one against one* method was implemented and the final decision was made by voting.

Accuracy of the SVM classification, besides depending on the chosen features, also depends on selection of the two parameters, soft margin parameter C and kernel function parameter. SVM model is defined by solving quadric optimization problem:

$$\frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \varepsilon_i, \tag{9}$$

where w is normal vector to the hyperplane and ε_i are slack variables.

Kernel function is used for nonlinear SVM and Gaussian radial basis function kernel (RBF) is usually a good choice. RBF is defined by the following equation:

$$K(X_i, X_j) = \exp(-\gamma||X_i - X_j||^2). \tag{10}$$

where X_i and X_j represent two feature vectors. C is the parameter for the soft margin cost function and controls the influence of each individual instance. Kernel function is used for enabling non-linear separable data to be classified by SVM. One of the most used kernel function is RBF which has a free parameter γ. Large γ will lead to high bias and low variance model, and vice-versa.

For building an optimal SVM model, values for the pair (C, γ) need to be found. The simplest method to find a good parameter pair is to use grid search. SVM model is build for numerous pair values and the best one is chosen for SVM training. Grid search is simple but it is also computationally expensive and represents discretized search for continuous parameters. Finding the optimal parameters is a hard optimization problem and one of the methods for solving them is usage of different metaheuristics. In the last decades nature inspired algorithms, especially swarm intelligence algorithms were proposed and successfully used for such problems.

In literature different swarm intelligence algorithms were applied to finding optimal SVM parameters problem. In [BHX13] particle swarm optimization (PSO) and pattern search based memetic algorithm were used for SVM parameters tuning while in [LH15] PSO was hybridized by artificial bee colony and adjusted for

SVM optimization. Fireworks algorithm for SVM optimization was proposed in [TTB16], [TTS16].

In this paper we adjusted a novel firefly algorithm (FA) for tuning SVM parameters for emphysema classification. Firefly algorithm was proposed in [Yan09] by Yang and it is inspired by social and flashing behavior of fireflies. It has been successfully used in numerous applications [BT14a], [BT14b], [TB14]. Fireflies are attracted by their lights where attractiveness of one firefly is directly proportional to its brightness. The brightness is defined by the following equation:

$$I(X) = \begin{cases} \frac{1}{f(X)} & \text{if } f(X) > 0 \\ 1 + |f(X)| & \text{othewise.} \end{cases} \quad (11)$$

where X represents position of the firefly and f is objective function. Attractiveness β depends on the distance between fireflies:

$$\beta(r) = \frac{\beta_0}{1 + \gamma r^2} \quad (12)$$

The position of a firefly i attracted to more attractive or brighter firefly j is updated by the following equation:

$$x_i^{t+1} = x_i^t + \beta e^{\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha_t \varepsilon_i^t \quad (13)$$

where β_0 represents attractiveness at $r = 0$, α is randomization parameter, ε_i^t represents a vector of random numbers drawn from a Gaussian distribution or uniform distribution at time t, and r_{ij} is distance between two fireflies (i, j).

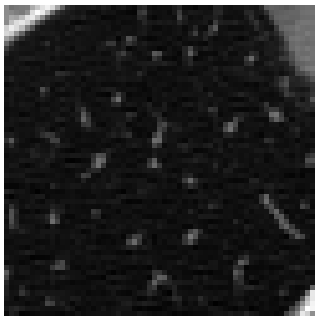
In our previous work [TMT16] we proposed FA for SVM optimization and the proposed method was tested on standard benchmark classification problems. The proposed method achieved better results comparing to the other state-of-the-art algorithms. FA algorithm was used for 2-dimensional problem, where search range for the first parameter C was set to be $[2^{-5}, 2^{15}]$ while the second parameter γ was in range $[2^{-15}, 2^5]$. Objective function was accuracy from 10-fold cross validation.

4 EXPERIMENTAL RESULTS

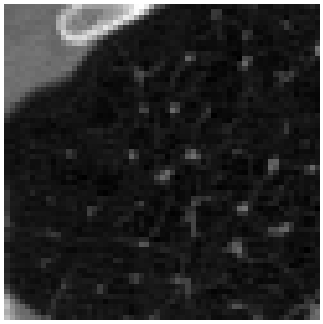
Proposed algorithm was implemented in Matlab R2016a while SVM was built by LIBSVM (Version 3.22) [CL11]. Experiments were performed on the platform with the following features: Intel ® Core™i7-3770K CPU at 4GHz, 8GB RAM, Windows 10 Professional OS.

Parameters for the FA were set empirically as follows: $\alpha = 0.2$, $\gamma = 0.23$ and $\beta_0 = 1$. Population size was 50, while the maximal number of iteration was set to 30. Additionally, if the accuracy did not improve in 10 consecutive iterations, algorithm stops. SVM parameters C and γ were searched in exponential space, search range for exponent for C was set to $[-10, 20]$ and the SVM model was built for $C = 2^x$ where x is parameter selected by FA, while exponent for parameter γ was searched in interval $[-20, 10]$ and similarly for SVM model $\gamma = 2^y$ was used where y was generated by FA.

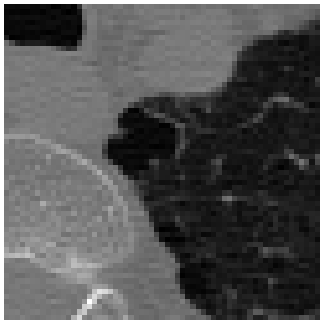
For the experiments we used standard public computed tomography emphysema database [SSdB10] available for download at *image.diku.dk/emphysema_database/*. Database contains 168 manually labeled 2D regions of interest (ROI) of size 61×61, slice thickness 1.25 mm and in-plane resolution of 0.78 mm $\times$ 0.78 mm. Three different types of ROI exist in the database, normal tissue (NT), centrilobular emphysema (CLE) and paraseptal emphysema (PSE). Examples of the patches from the database are shown in Fig. 4.



(a) NT



(b) CLE



(c) PSE

Figure 4: Examples of the regions of interst (normal tissue and emphysema) from the used database

The first part is to train support vector machine model for region classification based on the patches in the used

dataset. Because of the small number of patches (only 168) in the dataset, in most papers that used this dataset (including the papers that we compare with), all images were used for training. Since in this paper we proposed region based classification method with the region size of 31×31, for each patch 961 overlapping blocks were obtained (each patch size is 61×61), thus we were able to provide training set and test set based on the patches in the database. SVM model was trained with 1000 randomly chosen blocks for each class from different patches (3000 training regions in total). Test set had 1500 regions (500 from each class).

The best SVM model was for $C = 2^{18.23}$ and $\gamma = 2^{-12.36}$ where global accuracy was 98.18. In Table 1 confusion matrix of classification is presented. The rows represent true labels and in columns are labels assigned by SVM model. The results are in percents.

	NT	CLE	PSE
NT	100	0	0
CLE	0	100	0
PSE	0	5.47	94.53

Table 1: Confusion matrix for ROI classification by our proposed method (accuracies in %)

In order to prove quality of our proposed algorithm, we compared our results with two other methods. The first one was proposed in [SSdB10] where different rotation invariant LBP were used also along with the intensity histogram. Classification was done by *k*-nearest neighbor. Input vector for the classifier was joint LBP and intensity histogram. The second algorithm that was used for comparison was proposed in [YFAL16]. In [YFAL16] lung textures were described with sparse decomposition from texton dictionaries when different regularization were used. Sparsity was further expanded by adding constraints to the dictionaries.

In [SSdB10] global accuracy of the classifier was 95.24%. Global accuracy of patches classification by our proposed method was 98.18%. In [SSdB10] the largest error was made for classification NT class, where 93.22% was correctly classified while 6.88% was classified as PSE. Our proposed method had the lowest accuracy for classification of the PSE class - 94.53%. Instances from PSE class were classified as CLE in 5.47%. Instances of NT and CLE classes were classified with 100% accuracy. In [YFAL16] only global accuracies for different proposed methods were reported. The highest accuracy was 93.9%, while other methods achieved accuracies of 92.1%, 91.8%, 92.4% and 91.7%. All reported results in [YFAL16] were lower than the accuracy of our proposed method that achieved 98.18%. The comparison results are summarized in Table 2.

Our proposed method also achieved better results for classification patches compared to [NKOS15] where complete joint histograms of three different LBP variants and intensity were used as input for linear SVM. Reported accuracies were 81.36%, 82.99% and 83.29% while accuracy of our proposed method was 98.18%.

	THM	LBP-kNN	GFWA
NT	-	93.22	100
CLE	-	98.00	100
PSE	-	93.91	94.53
global acc.	93.9	95.24	98.18

Table 2: Comparison results of our proposed method and methods proposed in [SSdB10] and [YFAL16]

5 CONCLUSION

In this paper a method for lung tissue classification was proposed. Different lung tissues were described by simple and powerful texture features, uniform local binary patterns. Beside texture features, intensity features were used. In this paper feature vector was reduced comparing to the other state-of-the-art approaches by using histogram characteristics such as mean, standard deviation, skewness, kurtosis, energy and entropy, rather than the whole histogram. Compared to the other methods from the literature our proposed method achieved higher accuracy for classification of regions of interest provided in standard benchmark dataset. In further work feature selection can be included in the proposed algorithm and it can be additionally adjusted for CT image segmentation by introducing the fourth class that will distinguish lung tissue from the background.

6 ACKNOWLEDGMENTS

M. Tuba was supported by the Ministry of Education, Science and Technological Development of Republic of Serbia, Grant No. III-44006.

D. Simian was supported by the research grant LBUS-IRG-2015-01, project financed by Lucian Blaga University of Sibiu.

7 REFERENCES

[ACE+16] Marios Anthimopoulos, Stergios Christodoulidis, Lukas Ebner, Andreas Christe, and Stavroula Mougiakakou. Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. *IEEE transactions on medical imaging*, 35(5):1207–1216, 2016.

[BHX13] Yukun Bao, Zhongyi Hu, and Tao Xiong. A PSO and pattern search based memetic algorithm for SVMs parameters optimization. *Neurocomputing*, 117:98–106, 2013.

[BSBY14] Zahra Beheshti, Siti Mariyam Hj Shamsuddin, Ebrahim Beheshti, and Siti Sophiayati Yuhaniz. Enhancement of artificial neural network learning using centripetal accelerated particle swarm optimization for medical diseases diagnosis. *Soft Computing*, 18(11):2253–2270, 2014.

[BT14a] Nebojsa Bacanin and Milan Tuba. Firefly algorithm for cardinality constrained mean-variance portfolio optimization problem with entropy diversity constraint. *The Scientific World Journal*, 2014, 2014.

[BT14b] Ivona Brajevic and Milan Tuba. Cuckoo search and firefly algorithm applied to multilevel image thresholding. In Xin-She Yang, editor, *Studies in Computational Intelligence, Cuckoo Search and Firefly Algorithm: Theory and Applications*, volume 516, pages 115–139. Springer, 2014.

[CL11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, May 2011.

[DBF+13] C Dromain, B Boyer, R Ferre, S Canale, S Delaloge, and C Balleyguier. Computed-aided diagnosis (CAD) in the detection of breast cancer. *European journal of radiology*, 82(3):417–423, 2013.

[DSS15] S. R. Dubey, S. K. Singh, and R. K. Singh. Local diagonal extrema pattern: A new and efficient feature descriptor for CT image retrieval. *IEEE Signal Processing Letters*, 22(9):1215–1219, 2015.

[HAB15] Thomas J Hirschauer, Hojjat Adeli, and John A Buford. Computer-aided diagnosis of Parkinson's disease using enhanced probabilistic neural network. *Journal of medical systems*, 39(11):179, 2015.

[IM14] Musibau Ibrahim and Ramakrishnan Mukundan. Multi-fractal techniques for emphysema classification in lung tissue images. *International Conference on Environment, Chemistry and Biology*, 78:115–119, 2014.

[IM15] Musibau Ibrahim and Ramakrishnan Mukundan. Cascaded techniques for improving emphysema classification in computed tomography images. *Artificial Intelligence Research*, 4(2):112–118, 2015.

[KI15] E. M. Karabulut and T. Ibrikci. Emphysema discrimination from raw HRCT images by convolutional neural networks. In *9th International Conference on Electrical and Electronics Engineering (ELECO)*, pages 705–708, 2015.

[KWE14] S. Kurugol, G. R. Washko, and R. S. Estepar. Ranking and classification of monotonic emphysema patterns with a multi-class hierarchical approach. In *11th IEEE International Symposium on Biomedical Imaging (ISBI)*, pages 1031–1034, 2014.

[KY14] Dong-Hyun Kim and Soo-Young Ye. CAD for detection of brain tumor using the symmetry contribution from MR image applying unsharp mask filter. *Transactions on Electrical and Electronic Materials*, 15(4):230–234, 2014.

[LC15] Howard Lee and Yi-Ping Phoebe Chen. Image based computer aided diagnosis system for cancer detection. *Expert Systems with Applications*, 42(12):5356–5365, 2015.

[LH15] Kuan-Cheng Lin and Yi-Hsiu Hsieh. Classification of medical datasets using SVMs with hybrid evolutionary algorithms based on endocrine-based particle swarm optimization and artificial bee colony algorithms. *Journal of medical systems*, 39(10):119, 2015.

[NERCE14] Rodrigo Nava, Boris Escalante-Ramírez, Gabriel Cristóbal, and Raúl San José Estépar. Extended Gabor approach applied to classification of emphysematous patterns in computed tomography. *Medical and biological engineering and computing*, 52(4):393–403, 2014.

[NKOS15] Mizuho Nishio, Hisanobu Koyama, Yoshiharu Ohno, and Kazuro Sugimura. Classification of emphysema subtypes: Comparative assessment of local binary patterns and related texture features. *Advances in Computed Tomography*, 4(03):47–55, 2015.

[NTT16] Marina Nikolic, Eva Tuba, and Milan Tuba. Edge detection in medical ultrasound images using adjusted canny edge detection algorithm. In *24th Telecommunications Forum (TELFOR)*, pages 691–694. IEEE, 2016.

[Pei15] Xiaomin Pei. Emphysema classification using convolutional neural networks. In Honghai Liu, Naoyuki Kubota, Xiangyang Zhu, Rüdiger Dillmann, and Dalin Zhou, editors, *International Conference on Intelligent Robotics and Ap-*

plications (ICRA): Proceedings, Part I, pages 455–461. Springer International Publishing, 2015.

[RGSG+13] Javier Ramirez, JM Gorriz, Diego Salas-Gonzalez, A Romero, Miriam Lopez, Ignacio Alvarez, and Manuel Gomez-Rio. Computer-aided diagnosis of Alzheimer's type dementia combining support vector machines and discriminant set of features. *Information Sciences*, 237:59–72, 2013.

[RRK+14] Sushravya Raghunath, Srinivasan Rajagopalan, Ronald A Karwoski, Brian J Bartholmai, and Richard A Robb. Active relearning for robust supervised training of emphysema patterns. *Journal of digital imaging*, 27(4):548–555, 2014.

[SSdB10] L. Sorensen, S. B. Shaker, and M. de Bruijne. Quantitative analysis of pulmonary emphysema using local binary patterns. *IEEE Transactions on Medical Imaging*, 29(2):559–569, 2010.

[SSK15] Hayaru Shouno, Satoshi Suzuki, and Shoji Kido. A transfer learning method with deep convolutional neural network for diffuse lung disease classification. In *International Conference on Neural Information Processing*, pages 199–207. Springer, 2015.

[STT16] Aleksandar Stojak, Eva Tuba, and Milan Tuba. Framework for abnormality detection in magnetic resonance brain images. In *24th Telecommunications Forum (TELFOR)*, pages 687–690. IEEE, 2016.

[TB14] Milan Tuba and Nebojsa Bacanin. JPEG quantization tables selection by the firefly algorithm. In *International Conference on Multimedia Computing and Systems (ICMCS 2014)*, pages 153–158, April 2014.

[TMT16] Eva Tuba, Lazar Mrkela, and Milan Tuba. Support vector machine parameter tuning using firefly algorithm. In *26th International Conference Radioelektronika*, pages 413–418. IEEE, 2016.

[TTB16] Eva Tuba, Milan Tuba, and Marko Beko. Support vector machine parameters optimization by enhanced fireworks algorithm. In Ying Tan, Yuhui Shi, and Ben Niu, editors, *LNCS, Advances in Swarm Intelligence: 7th International Conference on Swarm Intelligence, Proceedings, Part I*, volume 9712, pages 526–534. Springer International Publishing, 2016.

[TTD17] Eva Tuba, Milan Tuba, and Edin Dolicanin. Adjusted fireworks algorithm applied to retinal image registration. *Studies in Informatics and Control*, 26(1):33–42, 2017.

[TTS16] Eva Tuba, Milan Tuba, and Dana Simian. Adjusted bat algorithm for tuning of support vector machine parameters. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 2225–2232. IEEE, 2016.

[Tub14] Milan Tuba. Multilevel image thresholding by nature-inspired algorithms: A short review. *Computer Science Journal of Moldova*, 22(3), 2014.

[Yan09] Xin-She Yang. Firefly algorithms for multimodal optimization. *Stochastic Algorithms: Foundations and Applications, LNCS*, 5792:169–178, 2009.

[YFAL16] J. Yang, X. Feng, E. D. Angelini, and A. F. Laine. Texton and sparse representation based texture classification of lung parenchyma in CT images. In *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1276–1279, 2016.

[ZCKR+13] T. Zulueta-Coarasa, S. Kurugol, J. C. Ross, G. G. Washko, and R. San Jose Estepar. Emphysema classification based on embedded probabilistic PCA. In *35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3969–3972, 2013.

Improving the ergonomics of hand tracking inputs to VR HMD's

Scott Devine
Queen's University Belfast
Northern Ireland
sdevine08@qub.ac.uk

Chris Nicholson
Carleton University
Ottawa, Canada
chrisnicholson@cmail.carleton.ca

Karen Rafferty
Queen's University Belfast
Northern Ireland
K.Rafferty@ee.qub.ac.uk

Chris Herdman
Carleton University
Ottawa, Canada
chrisherdman@cunet.carleton.ca

ABSTRACT

This study improves the ergonomics of using the Leap Motion hand tracking device with an Oculus Rift. The improvements were realised through the use of a 3D printed mount that angled the Leap Motion down by 30 degrees. This allowed for users to interact with a virtual environment in which their arms may be held in a biomechanically less stressful location, rather than up and in front of their face. To validate the configuration, 15 participants completed a specially designed task which involved pressing virtual buttons in a given location. The button pressing task was performed in three configurations that compared the angled mount against the standard forward facing mount. Results indicate that the angled mount eliminates tracking loses, whilst producing comparable accuracy against the control condition and allowing the participant to interact in a more natural arm posture.

Keywords

Human Computer Interaction; Hand Tracking; Virtual Reality; Leap Motion; Oculus Rift;

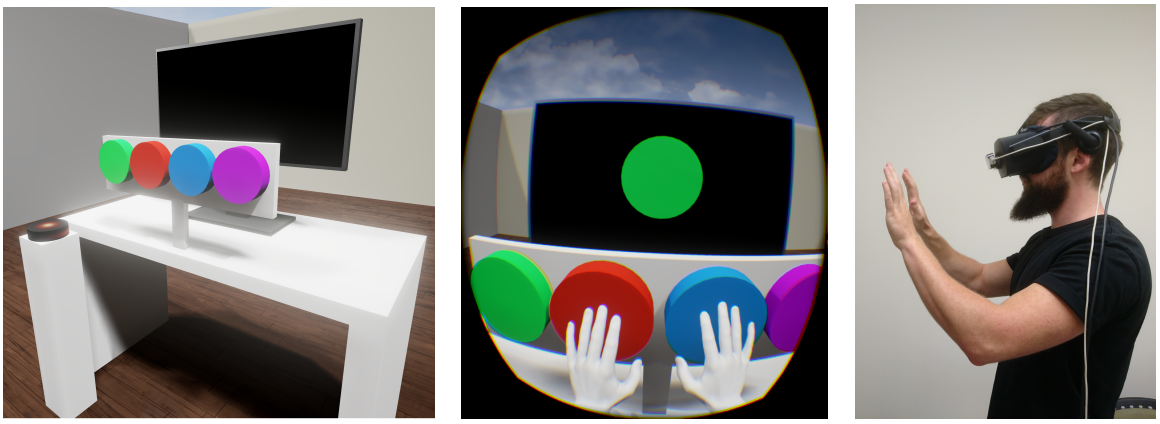
1 INTRODUCTION

In recent years the VR (Virtual Reality) industry has grown considerably with the mainstream adoption and release of multiple high fidelity HMD's (Head Mounted Displays) including the Oculus Rift [1] and the HTC Vive [2]. Both of these HMD's use physical hand controllers that allow users to interact with the virtual environment. These controllers are very accurate, but do not allow for the natural inputs afforded by the entire hand and fingers. An ideal solution would involve high fidelity low latency hand tracking that allows the user to manipulate 3D representations of his/her own hands in VR.

The Leap Motion [3] device was designed to allow users to interact in virtual environments using only their hands. It is an optical tracking device that uses time of flight calculations from emitted infrared light to track both the hands and fingers in 6 DOF. Though originally designed as a desktop mounted device, the potential to solve the limitations of inputs in VR with physical controllers was realised. Leap Motion's Orion update allowed the device to be mounted directly on a HMD and track the hands in front of the face. The Leap's software has undergone serious optimisations whilst the hardware remains the same. The Leap Motion now tracks a participants hands with the highest accuracy with the palms facing away and parallel to the device, this can be seen in Figure 1c.

However, the limited 120 degree horizontal FOV (Field Of View) produces a new set of challenges when tracking users hands in VR. A user has to bring their hands directly into view (Figure 1c) or point the HMD toward what they are interacting with (Figure 2c). Both of these movements are unnatural and have the potential to cause user discomfort. Persistently elevated arm positions in user interfaces causes arm fatigue [4, 5] and looking down at the hands causes discomfort in the shoulders and neck when used for extended periods. Ideally, the FOV of the Leap Motion would be large enough to allow the user to look forward and interact with his/her forearms at a more ergonomic 90 degree to his/her body (Figure 3c).

The research documented within this paper was designed to investigate the potential for an angled mount to mitigate the aforementioned limitation of current Leap Motion design. The mount developed for this research, angles the Leap Motion down, toward the users hands by 30 degrees. This changes the optimal tracking area, allowing a user to look up and forward at the surrounding environment, whilst their hands are still tracked. As shown in Figure 3c this allows users to interact with the environment in a more comfortable manner. It was hypothesized that moving the effective



(a) Scene view of desk and monitor with forward pressing buttons

(b) User view of Forward condition

(c) Interaction posture of Forward condition with original mount

Figure 1: Forward Condition

tracking range down would eliminate tracking loses, improve user comfort and provide a better VR interaction experience. Section 2 details related literature, however, to the knowledge of the authors no one has attempted to angle the Leap Motion whilst attached to a HMD before. Section 3 covers the device and mount set-up and Section 4 details the conducted user study. The results are detailed and discussed in Section 5, with Conclusions in Section 7.

2 RELATED WORK

The Leap Motion has been used for numerous human computer interaction studies from numerical gesture recognition [6] to hands-free immersive image navigation [7]. The Leap Motion has allowed researchers to acquire 6 DOF tracking data of both the hands and finger at millimetre accuracy [8, 9]. Though still using hardware from 2013, the Leap Motion controller continues to under go software revisions, most recently receiving the Orion update. This update was specifically designed to improve use when mounted on VR headsets. However, little research uses the Leap Motion mounted on the front of the HMD, it is instead used in a desktop fashion. Both desktop and HMD mounted configurations of the Leap Motion can cause fatigue and this is reflected in the literature.

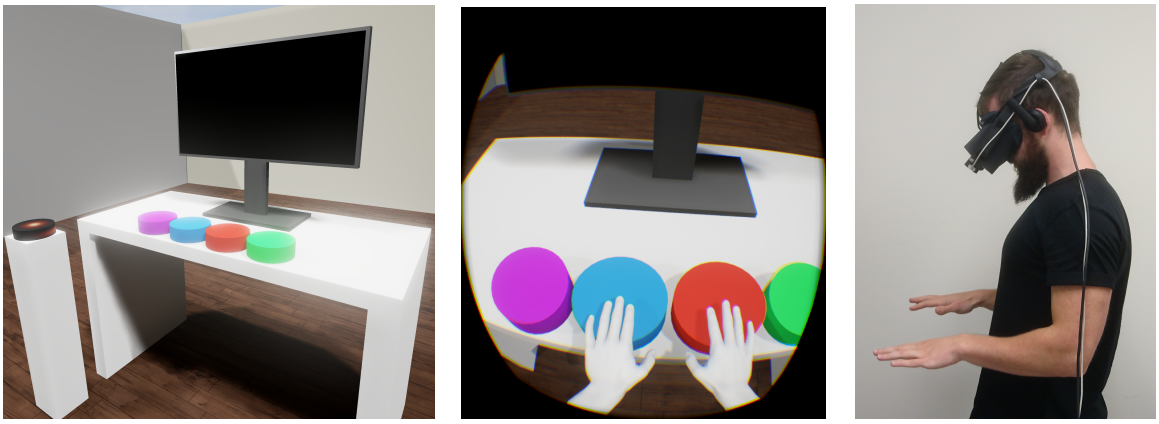
This is the case in [10], where they highlighted that users appeared to become tired when using the Leap Motion, having to hold their hands above the Leap Motion when mounted in a desktop configuration.

To improve driving pleasure in autonomous vehicles Manawadu et al. [11] used the Leap Motion to give the driver some lateral and longitudinal control of the vehicle. However, they observed that drivers found it difficult to input the gestures exactly inside the interaction space. The interaction space, which was desk mounted, was not in a comfortable position, from the upper chest to the users chin.

Vosinakis et al. [12] found that half of the participants reported arm fatigue during an engraving task when using the Leap Motion for Cycladic Sculptur. The Leap Motion was mounted in desktop mode with participants having to hold their hands above the device. The authors recommended that participants not hold their hands out at a stretched position for more than 20 seconds at a time. This recommendation is not practical for interacting in VR, interaction with VR should not be contained to 20 second bursts.

Lee et al. [13] evaluated adding a touch screen to the front of a GearVR headset [14] as a means of input to the mobile HMD. They attached the touch screen to the front of the HMD pointing outward, a participant raised their hands in front of their face and touched the screen to interact. They noted that 15 out of 20 participants experienced neck and arm fatigue when interacting with the touch screen on front of their face.

Al-Megren et al. [15] conducted a shoulder fatigue study, using both subjective and objective measures. The objective measures involved using surface Electromygraphy (SEMG) to monitor three muscles on each arm. They tested horizontal and vertical multi touch displays for arm fatigue when interacting, subjective measures showed arm fatigue for both horizontal and vertical interactions. However, objective measures showed a significant level of muscle fatigue on the middle deltoids and the non-dominant extensor digitorum for the vertical configuration only. This shows that participants shoulders were suffering from fatigue when interacting on a vertical panel, or up in front of their face. The authors noted that their results have clear implications on interaction design for large interactive displays, noting that vertical displays, while acceptable for short periods of time, are not suitable for frequent use. Whilst this study was conducted using interactive touch screens, the same applies to interacting with tracked hands in VR, specifically with



(a) Scene view of desk and monitor with down pressing buttons

(b) User view of Down condition

(c) Interaction posture of Down condition with original mount

Figure 2: Down Condition

the Leap Motion. This is because to achieve the best tracking a participant must interact with a vertical plane parallel to the device, provided that participant is looking forward.

To help reduce shoulder fatigue, Vuibert et al. [16] designed the width of the tracking volume to reside between the hip and the shoulder of the participants. In other work, researchers at NASA incorporated pointing rather than hand positioning to move the cursor in a VR menu. This allowed the user greater flexibility in the positioning of the arms and hands. It also reduced fatigue and improved accuracy [17].

In sum, there are motivations for making the interaction with VR and the use of the Leap Motion more ergonomic or user friendly. The literature shows that participants interacting with their arms raised will get fatigued, this also happens to be where the Leap's tracking is most accurate.

3 DEVICE AND MOUNT SETUP

In order to shift the optimal tracking volume of the Leap Motion to a position that is more comfortable for the user, the device was angled downward. In order to do this a custom angled mounting bracket was created. Using a DaVinci 1.0 [18] 3D printer various degrees of angled mounting brackets were printed. These brackets attached into the clip that holds the original, official Leap Motion mount. Initially 30, 45, 60 and 90 degree angled mounts were created. Though through preliminary testing, all but the 30 degree mount were deemed too large. The standard Leap Motion mount can be seen in Figures 1c and 2c, the 30 degree angled mount can be seen in Figure 3c. The mounts were attached onto an Oculus Rift CV1.

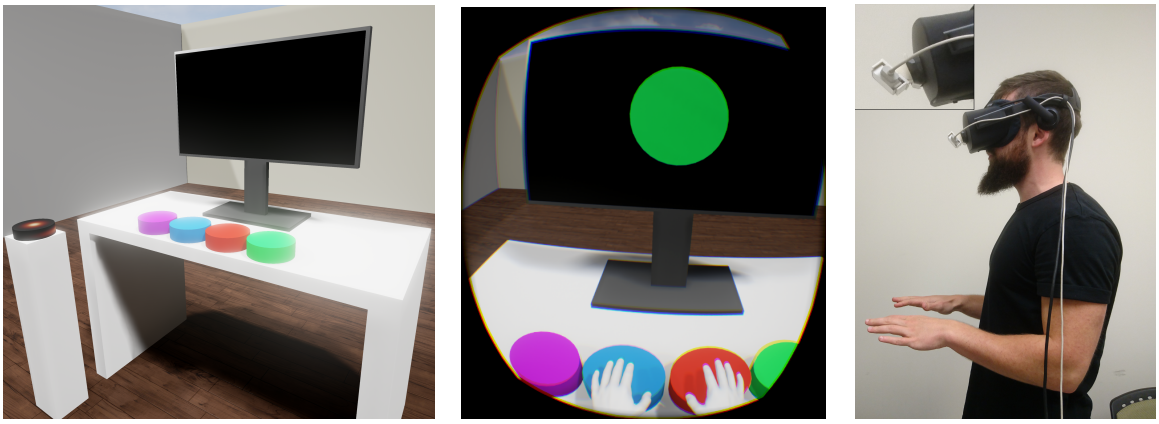
4 USER STUDY

A total of 15 people took part in the user study with an average age of 27. 60% were male and the remaining 40% female. 74% of participants were right handed with the remaining 26% left handed

Interaction with VR is task dependant [19], so the task in the present study was designed to mirror a generic task where the user is providing inputs dictated by the information being presented on a display. Because of the nature of the original Leap mount this type of task becomes difficult as in order to insure minimal tracking losses the user interaction has to be moved up, in front of the user. The input task required participants to press virtual buttons whose colours corresponded with the colour presented on a virtual display as quickly and accurately as possible. The scene was developed using Unreal Engine 4.13. Within the scene there was a rendered computer monitor, (Figures 1a,2a,3a), circles of a specific colour would appear on the monitor, and the user would have to hit the corresponding button. There were four colours available, these were randomly generated to appear on the monitor. Each colour was displayed on the monitor for 500ms. Participants had one second to respond in total. The locations of the four coloured buttons were randomised for each condition. All trials were conducted while participants were standing.

The user study consisted of three conditions counterbalanced across participants, with 100 trials per condition, these were spilt into blocks of 20. The participants started each block of 20 by pressing a start button to their left (Shown in Figure's 1a,2a,3a).

Condition one consisted of buttons that faced the participant (Forward pressing buttons), raised on a stand, this can be seen in Figure 1. These buttons were parallel to the screen and to the Leap Motion. For condition one the Leap Motion was mounted in its original official mount. A view from the participants perspective can be seen in Figure 1b. It should be noted here that both the displayed colour and the interaction buttons are within the participants FOV. We will refer to condi-



(a) Scene view of desk and monitor with down pressing buttons

(b) User view of Angled condition

(c) Interaction view of the Angled condition, with the angled mount

Figure 3: Angled Condition

tion one as the Forward (Buttons) condition. Condition two consisted of buttons on the desk that the rendered computer monitor was placed on, shown in Figure 2a. This is more a comfortable position for the participant to hit the buttons, again for this trial the Leap motion was mounted in its original position. However, from the participants perspective (Figure 2b), only one of either the displayed colour or the buttons can be seen at a time, the FOV is not large enough to capture both. We will refer to condition two as the Down (Buttons) condition. Lastly, the third condition (Figure 3a) used the 30 degree angled mount, like the second trial the buttons where placed on the desk for a more comfortable experience. In contrast to the Down condition, our modified setup with the angled mount allows for both the buttons and the displayed screen colour to be seen at the same time, within the participants FOV (Figure 3b), whilst also providing a more comfortable interaction environment. Condition three will be referred to as the Angled (Mount) condition.

Participants completed a short survey after the completion of each experimental condition to subjectively assess each condition. Participants rated their level of agreement from 1 (disagree) to 7 (agree) on the following statements:

1. I could easily press the correct buttons with the virtual fingers.
2. I often hit buttons I did not mean to hit.
3. It was easy to move the virtual hands where I wanted.
4. It was frustrating trying to hit the correct buttons in the virtual environment.
5. The virtual hands often disappeared when I was trying to press the buttons.

The expected results are shown in Table 1. Accuracy is defined as the correct number of displayed colours matched with a hit on the correctly coloured button, before a new colour has appeared on the monitor, only the first hit counts. Tracking losses were measured as the number of times one or more hands was not visible in VR, this is because both hands were required for the task. User comfort was quantified subjectively, asking the participants if they felt fatigued after each condition. The direction view vector of the HMD was recorded to allow for tracking of gaze direction and height. Response times were also recorded for each condition.

5 RESULTS

Dependant measures were each analysed in separate one-factor repeated measures ANOVAs with three levels (Condition: Forward, Down, Angled). Comparisions across the levels were made using 95% CI's (Confidence Intervals) generated from the ANOVAs [20].

Tracking Loss. Of primary interest was that the angled mount substantially reduced tracking losses relative to the other conditions, $F(2,28) = 13.874, p < .001, part-eta = .4498$. As shown in Figure 4, there were significantly fewer tracking losses in the Angled condition than in the Forward and Down conditions. In fact, the number of tracking losses in the Angled condition was negligible with a mean of 0.66 times, indicating that the Leap Motion system with the angled mount was able to maintain near 100% tracking of the users hands.

Accuracy. The Analysis of the mean accuracy scores showed a significant effect of condition, $F(2,28) = 11.228, p < .001, part-eta = .445$ As shown in Figure 5 participants were significantly less accurate at hitting the correct virtual button in the Down condition than the Forward and Angled conditions. The finding that the accuracy did not differ between the Forward and the Angled conditions is important as it shows that the most effective tracking volume has been moved.

Table 1: Expected Results

Forward Condition	Down Condition	Angled Condition
High accuracy	Low accuracy	High accuracy
Low tracking loss	High tracking loss	Low tracking loss
User discomfort (Arms)	User discomfort (Neck)	User comfort
Low gaze variation	High gaze variation	Low gaze variation

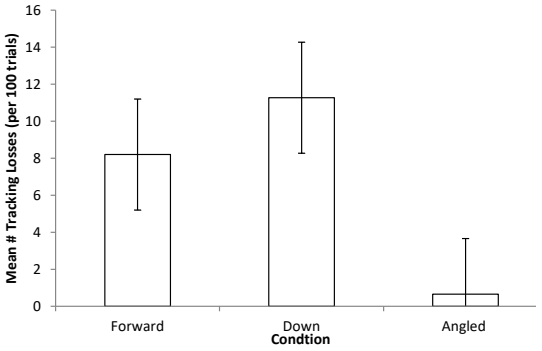


Figure 4: Tracking Losses

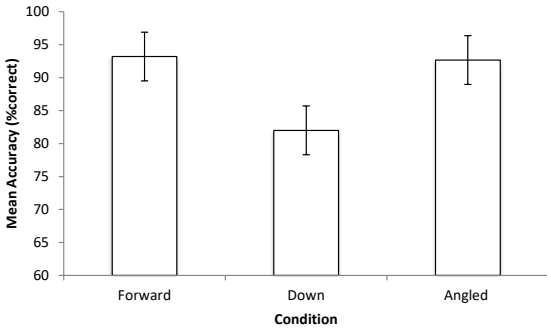


Figure 5: Accuracy

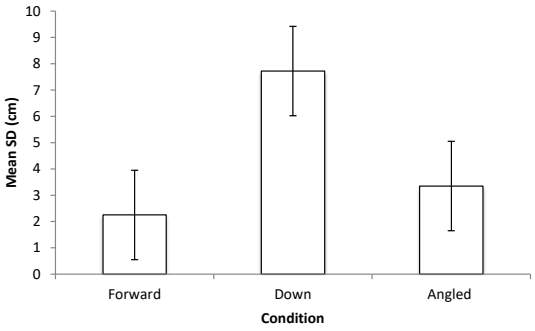


Figure 6: Gaze Variation

Response Times. Response times were measured from the initial presentation of a target to the point at which a virtual button was hit. Overall, 7.4% of the response time data was lost due to mechanisms used to make the buttons 'bounce'. The buttons work by linear interpolating between two positions. If the participant did not move their hand away quick enough and were late in reacting to the displayed colour, the system registered a hit for the next displayed colour, at an instant response time of zero seconds. It is interesting to note that the more data was lost in the Down (12.3%) condition as compared to 5.3% of the Forward condition and the Angled condition at 4.4%.

An analysis of the mean correct response times showed a significant effect of condition, $F(2,28) = 5.827, p = .008, part - eta = .294$. Response times were faster in the Forward (714 ms) condition than in the Down (733 ms) and the Angled (755 ms) conditions. This was expected as in the Forward condition the response buttons and the target were both presented in the same forward field of view. Response times did not differ significantly between the Angled and the Down conditions.

Gaze Variation. Gaze Variation was calculated as the standard deviation of gaze location across each of the 100 trials for each condition. The gaze variation is measured as z height variation from the HMD view vector. There was a significant effect of condition on gaze variation, $F(2,28) = 23.383, p < .001, part - eta = .625$. As shown in Figure 6, gaze variation was larger in the Down than the Forward and Angled conditions. This reflects the fact that in the Down condition participants had to move their gaze from the targets in the up position in order to see the virtual buttons in the down location. Gaze variation did not differ significantly between the Forward and the Angled conditions. Gaze variation was understandably low in the Forward condition as both the targets and the virtual buttons were positioned in the same forward field of view. The low gaze variation in the Angled condition suggests that with the 30 degree mount participants were able see their hands while also seeing the target stimuli.

Survey Responses. Analysis of the survey results indicated that two of the questions indexed a significant impact of condition. The first question asked the participants how easily they could press the buttons with the virtual tracked fingers. Participants indicated that the Forward condition had the highest score (Higher is better) out of 7 at 5.29, with the Angled condition following closely at 4.96, the Down condition trailed both these with an average answer of 3.79. The second question asked how often the participants felt the hands disappeared when they were trying to press a button, again out of 7. The Angled condition achieved the lowest score (Lower is better) at 2.29, with the Forward condition receiving 3.04 and the Down condition the highest at 3.96.

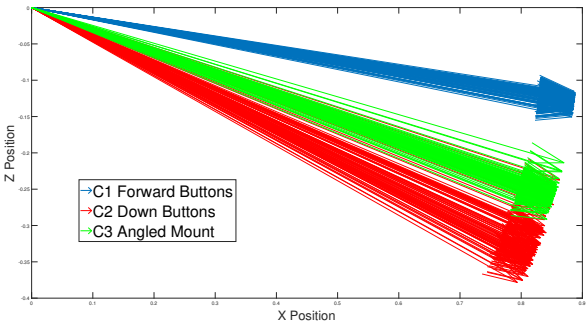


Figure 7: View Vectors

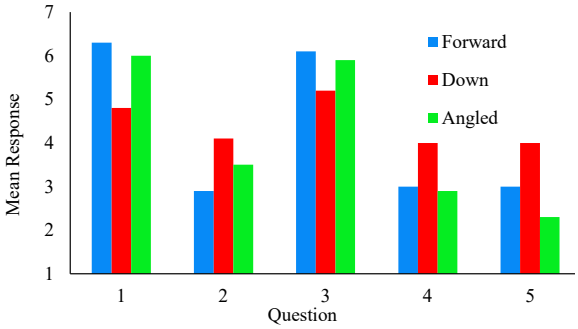


Figure 8: Survey questions results.

6 DISCUSSION

Due to the high variability in gaze location in the Down condition, there was a dramatic difference in the numbers of times that tracking was lost (Figure 4) over the 100 trials in each condition. For the angled condition, there was an average of 0.66 tracking losses over 100 trials. In contrast, there was an average of 11.27 tracking losses per 100 trials on the Down condition. Because the Leap is mounted on the head, every head movement changes the location of the Leap tracking volume thus increasing the likelihood of a tracking loss occurring.

As predicted the Down condition had the lowest accuracy while there was no difference between Forward and Angled conditions. The lack of difference in the accuracy between the Forward and Angled conditions is because the angled mount allows for the emulation of similar characteristics of the Forward condition, in a more comfortable manor. In the Forward condition the forward buttons, target stimuli and hands are all located in the users FOV. Using the angled mount allows for the buttons, target stimuli and hands to all be within the users field of view when interacting with down buttons, which they are not in the Down condition. The accuracy on the Down condition was well above chance so although more challenging, the task was of reasonable difficulty.

Gaze location can be used as a metric for task comfort. Moving the head up and down or standing with the neck angled down for long periods can lead to a host of health problems [21]. For this metric, we are concerned with the height component of the view vector. In this case the z component represents gaze height. Ergonomists recommend that one's gaze be aligned parallel to the floor. This insures the neck in aligned vertically with the spine. In relation to this, a higher z value, or as the value approaches zero (See Figure 7) the more optimal the value is. In Figure 7 the view vectors from an arbitrary point (HMD Origin) are displayed. All 100 trials per condition are averaged across participants. Blue arrows represent the Forward condition. As predicted, the Forward condition has the highest gaze height with little variability which means participants did not move their head much. Ergonomically, the Forward condition is the best position for the neck. However, the Forward condition requires participants to hold their arms in front of their face for the majority of the trials.

The Down Condition (Red) had the lowest gaze height and highest gaze variability making it the least ergonomic. The use of the angled mount resulted in an improvement in the gaze height and a reduction of gaze variability. It was initially assumed that use of an angled mount of 30 degrees would allow for an improvement of 15 degrees between the Down and Angled conditions. However, when all the trials are averaged out, and the angle between the upper x axis (parallel to the floor) and these gaze vectors are calculated, this assumption was shown to be incorrect. The angle between the x axis and the Forward conditions gaze vector is -8.29 degrees. The angle for the Down condition is notably larger, at -21.71 degrees. The angled mount improves this for the Angled condition to -16.89 degrees, an improvement of 4.82 degrees (4.87cm z height difference).

Response times were significantly faster in the Forward condition when the buttons were elevated up in front of the participants and close to where the targets were displayed. We hypothesized that the fastest response times would occur in the Angled condition because participants hands were in the most comfortable position while still being tracked. However, it is likely that due to the proximity of the buttons and the target stimulus, although less comfortable, the Forward condition allowed for quicker responses. We hypothesized that the Down condition would result in the slowest response times because participants had to move their heads from trial to trial in order to keep the Leap hands in view. The results reveal this not to be the case. It is possible that there was a speed accuracy trade-off in the Down condition with participants being relatively quick but making more errors and resulting in more tracking losses.

Although subjective, a number of participants (three), mentioned explicitly that their shoulders were feeling fatigued, also worth noting is that these participants ran the study in one of the counter balanced orders resulting in them completing the forward condition last, the most taxing.

In the Forward condition, the optimal position for tracking (Figure 1c) consisted of the user looking straight ahead and bringing their hands into view. However, after a number of trials participants began to lower their arms and look down slightly. They lowered their arms into what we call the "T-Rex" position, that is their elbows tucked tight into their sides for support with their hands up and pulled back parallel to their shoulders. From this position participants in the Forward condition raised their hands up into the field of view to push a virtual button. Had a default resting position been set, or a line drawn in front of them in VR, where the participant had to keep their hands above for the duration of the condition, there may have been a similar amount of tracking losses to the Angled condition.

The participants were asked to answer five questions after each condition. Answers ranged from 1-7, 1 meaning disagree, 4 neither agree nor disagree and 7 agree. Comparing the participants survey answers for both down button conditions (Down and Angled) it becomes apparent that they agreed the angled mount made it easier to press the correct buttons. This is also clear for the second significant question relating to tracking issues, there is a huge difference in the amount of times tracking is lost across all conditions, the participants also acknowledged this. The participants answers for the Down and Angled condition are almost exactly opposite ends of the agree and disagree scale, with scores of 4.96 and 3.29 (1 to 7 range, 4 neither agree nor disagree), they both edge into strongly disagree and strongly agree for how well they thought the hands were tracked, this is shown in Figure 8. The participants answers reflect the results measured objectively, they recognised the benefits of the angled mount.

During the Forward condition the hit location on the button was recorded, this was recorded as we hypothesized that the vertical location (z) of the hit would drop if they fatigued, resulting in an objective measure for fatigue. There was no significant relationship between the location of button press and the trial number, meaning the data does not support the hyposthsis that peoples hits lowered as the trials went on. This is not to say they were not fatigued but this measure does not capture it. We know that participants will get fatigued based on previous SEMG monitoring studies [15], and while anecdotal we believe the "T-rex" position they default to during the Forward condition reflects this.

7 CONCLUSIONS AND FUTURE WORK

This paper introduced and evaluated a custom 3D printed angled mount for use with the Leap Motion and HMD's. Changing the orientation of the optimal tracking volume for a Leap Motion attached to a HMD improves user comfort and allows a more natural posture to be adopted when using such devices for a prolonged period of time. The mount allows for a person to have their arms at a comfortable 90 degrees to their body, rather than holding them raised in front of their face, which can result in fatigue. 15 participants completed a task to evaluate the effectiveness of the new angled mount. Our hypotheses of the improvements the angled mount would bring were correct. Not only did the angled mount achieve the accuracy of the optimal tracking Forward condition (Figure 5), it eliminates tracking losses, in comparison to both the Forward and Down conditions (Figure 4).

Alongside the expected behavioural improvement, the analysis provides evidence that the biomechanical mechanism by which the test manipulation is expected to improve performance is detectable itself in gaze measurements, with the angled mount cutting gaze variation by over 50%. This reduction in gaze variation can improve neck comfort. Using the angled mount allows for optimal tracking, comfortable interaction and reduced head movement.

The angle of the mount studied may have actually been too large. We believe the optimal angle to be somewhere between 15 and 30 degrees, so as to reduce the impact of any possible interactions above the headset. As interaction with VR is task dependant [19] we foresee a use for an adjustable mount in the future. This mount could feed its angle into Unreal Engine, allowing for on the fly customisation based on the task at hand.

Virtual reality is centred around experiencing new sights and environments, this is impacted negatively when having to look down at what is being interacted with, rather than being engaged with a greater portion of what is displayed in the environment. In everyday life interaction is mostly completed in the peripherals of our vision. The use of the angled mount affords more natural interactions utilising more of the peripheral vision, like that of everyday interactions.

ACKNOWLEDGEMENTS

To the ACE lab, many thanks for an awesome Canadian adventure, without whom this would not be possible.

REFERENCES

[1] OCULUS Inc, 2015. URL `https://www.oculus.com/en-us/`.

[2] HTC Corporation, 2016. URL https://www.htcvive.com/uk/.

[3] Leap Motion Inc. Leap Motion Inc, 2015. URL https://www.leapmotion.com/.

[4] Bengt Ahlström, Sören Lenman, and Thomas Marmolin. Overcoming touchscreen user fatigue by workplace design. In *Posters and Short Talks of the 1992 SIGCHI Conference on Human Factors in Computing Systems*, CHI 92, pages 101–102, New York, NY, USA, 1992. ACM.

[5] Andrew Sears. Improving touchscreen keyboards: design issues and a comparison with other devices. *Interacting with computers*, 3(3):253–269, 1991.

[6] J. K. Sharma, R. Gupta, and V. K. Pathak. Numeral gesture recognition using leap motion sensor. In *2015 International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 411–414, Dec 2015.

[7] M. Sreejith, S. Rakesh, S. Gupta, S. Biswas, and P. P. Das. Real-time hands-free immersive image navigation system using microsoft kinect 2.0 and leap motion controller. In *2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*, pages 1–4, Dec 2015.

[8] Jože Guna, Grega Jakus, Matevž Pogačnik, Sašo Tomažič, and Jaka 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.

[9] Frank Weichert, Daniel Bachmann, Bartholomäus Rudak, and Denis Fisseler. Analysis of the accuracy and robustness of the leap motion controller. *Sensors*, 13(5):6380–6393, 2013.

[10] D. E. Holmes, D. K. Charles, P. J. Morrow, S. McClean, and S. M. McDonough. Using fitt's law to model arm motion tracked in 3d by a leap motion controller for virtual reality upper arm stroke rehabilitation. In *2016 IEEE 29th International Symposium on Computer-Based Medical Systems (CBMS)*, pages 335–336, June 2016.

[11] U. E. Manawadu, M. Kamezaki, M. Ishikawa, T. Kawano, and S. Sugano. A haptic feedback driver-vehicle interface for controlling lateral and longitudinal motions of autonomous vehicles. In *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 119–124, July 2016.

[12] S. Vosinakis, P. Koutsabasis, D. Makris, and E. Sagia. A kinesthetic approach to digital heritage using leap motion: The cycladic sculpture application. In *2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, pages 1–8, Sept 2016.

[13] Jihyun Lee, Byungmoon Kim, Bongwon Suh, and Eunyee Koh. Exploring the front touch interface for virtual reality headsets. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 2585–2591. ACM, 2016.

[14] Samsung. GearVR, 2016. URL http://www.samsung.com/global/galaxy/gear-vr/.

[15] Shiroq Al-Megren, Ahmed Kharrufa, Jonathan Hook, Amey Holden, Selina Sutton, and Patrick Olivier. *Comparing Fatigue When Using Large Horizontal and Vertical Multi-touch Interaction Displays*, pages 156–164. Springer International Publishing, Cham, 2015. ISBN 978-3-319-22723-8.

[16] Vanessa Vuibert, Wolfgang Stuerzlinger, and Jeremy R Cooperstock. Evaluation of docking task performance using mid-air interaction techniques. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction*, pages 44–52. ACM, 2015.

[17] Stephen R Ellis. Using virtual menus in a virtual environment richard h. jacoby sterling software nasa ames research center, m/s 262-6 moffett field, ca. 94035. *Visual Data Interpretation: 10-11 February 1992, San Jose, California*, 1668: 39, 1992.

[18] AYZ Printing. da vinci 1.0, 2016. URL http://us.xyzprinting.com/us_en/Product/da-Vinci-1.0.

[19] Kay M. Stanney, Ronald R. Mourant, and Robert S. Kennedy. Human factors issues in virtual environments: A review of the literature. *Presence: Teleoper. Virtual Environ.*, 7(4):327–351, August 1998. ISSN 1054-7460.

[20] Jerzy Jarmasz and Justin G Hollands. Confidence intervals in repeated-measures designs: The number of observations principle. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 63(2):124, 2009.

[21] Dennis R Ankrum and Kristie J Nemeth. Head and neck posture at computer workstations-what's neutral? In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 44, pages 5–565. SAGE Publications, 2000.

Extraction of Sliding Collision Area of Knee-Form for Automobile Safety Inspections

Masatomo Inui
Dept. of Intelligent Systems Engr.
Ibaraki Univ.
Nakanarusawa 4-12-1
316-8511, Hitachi, JAPAN
masatomo.inui.az@vc.ibaraki.ac.jp

Kenta Gunji
Dept. of Intelligent Systems Engr.
Ibaraki Univ.
Nakanarusawa 4-12-1
316-8511, Hitachi, JAPAN
17nm914h@vc.ibaraki.ac.jp

Nobuyuki Umezu
Dept. of Intelligent Systems Engr.
Ibaraki Univ.
Nakanarusawa 4-12-1
316-8511, Hitachi, JAPAN
nobuyuki.umezu.cs@vc.ibaraki.ac.jp

ABSTRACT

The United Nations Economic Commission for Europe defines a safety regulation based on the possible collision between the driver's knee and an automobile's interior parts. The "knee-form" apparatus is used to evaluate compliance with this regulation. Current software for analyzing possible collisions of the knee-form is not applicable to the part whose surface is vertical and near parallel to the knee-form approaching direction. In this paper, we propose a novel algorithm named "push-and-slide" for extracting the knee-form colliding area on the console part and door panel. In the first step of the algorithm, the target surface of the part is transformed to grid-like points in a high resolution. The knee-form models in various positions and orientations are prepared in the second step. Each knee-form model is pushed to the grid-like points. The model is then moved along the part surface to detect possible collisions between the knee-form and the grid-like points. An experimental system is developed and some computational experiments are performed.

Keywords

UNECE inspection, automobile safety, collision detection, knee-form, CAD.

1. INTRODUCTION

In driving an automobile, speed control is basically realized by the driver's leg and foot motion. If interior parts locating around the driver, such as instrument panel, console part and door panel, hinder the smooth motion of the leg, the operability of the automobile is significantly impaired. If the knee touches electric switches while driving, it may cause traffic accidents in the worst case. To prevent such problems, the United Nations Economic Commission for Europe (UNECE) defines a safety regulation based on the possible collision between the knee and the interior parts [GAR]. Similar regulations are defined in other countries including Japan and China.

Figure 1 explains the UNECE regulation No. 21 concerning the knee collision. In the automobile design, CAD model of the automobile body, which is an assembly of exterior and interior part models, is positioned so that its front part faces the negative direction of the x-axis of the object's coordinate frame. An apparatus imitating the knee shape during driving is defined in the regulation. This "knee-form" has an equilateral triangle-like shape and a thickness of 120 mm. It has a rounded corner that can be represented by a cylinder with a radius of 60 mm as shown in Figure 1(a). To inspect for compliance with the regulation, the knee-form is positioned in the driver's space as its round corner faces towards the front part of the automobile. It is moved linearly in the negative direction of the x-axis, where the front surface of the knee-form eventually collides to an interior part, for example instrument panel. This inspection is repeated by changing the initial position

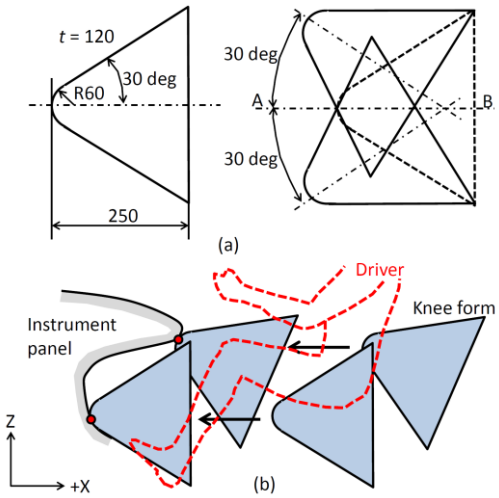


Figure 1. Knee-form collision analysis as defined in the UNECE regulation No. 21.

of the knee-form. In this way, as shown in Figure 1(b), surface areas on the interior parts are detected where the knee can make collision. Before starting the movement, the apparatus can be rotated above and below the horizontal by up to 30 degrees to represent different orientations. In the actual driving, seat position, position and orientation of the driver on the seat, and his/her height affect the knee collision on the interior part. The UNECE regulation, however, do not consider such parameters in the collision detection.

Because interior components significantly affect the appearance and comfort of the automobile, they are often designed in terms of their function and aesthetics. Knee-form collision detection is manually inspected by specialists in the final design stage with a physical apparatus and models. This work is difficult, time consuming, and prone to human errors, and detection of safety problems at this stage causes costly re-works of the designing. Therefore, a fast and automatic inspection method that allows the designers themselves to check the regulations during the shape design is desirable.

2. OUTLINE OF CONTRIBUTION

Although UNECE regulation inspections are an important topic for automobile manufacturers, technologies for automating the inspection tasks have rarely been studied, and many manufacturers still use manual inspection methods.

Some commercial systems are known for verifying the UNECE regulations, such as the CAVA system [CAVA]. The Toyota Motor Corporation submitted some patents relevant to the automatic inspection of UNECE regulations 17, 21, and 25, concerning collisions of a large sphere representing an infant's head [JP06]. These systems detect only the sphere contacting area on automobile parts and do not support the regulations concerning the knee-form.

Yamazaki et al. proposed a UNECE regulation inspection system based on detecting the intersection between a CAD model and spheres placed on the model surface [Yam11]. We developed an improved method for the rapid detection of the sphere-contacting shape for evaluating the UNECE regulations [Inu15]. These systems use the parallel processing capability of a Graphics Processing Unit (GPU) for accelerating the computation [CUDA]. They do not support the collision detection with the knee-forms.

We proposed a method for automatically detecting the contacting area of the knee-form on the instrument panel [Inu17]. The "knee-form" apparatus can be modeled as a Minkowski sum [Li11] shape of a vertical equilateral triangle and a horizontal cylinder with a radius of 60 mm and a thickness of 120 mm. The knee-form contacting condition is geometrically equivalent to that of an equilateral triangle contacting a Minkowski sum shape of the instrument panel and a horizontal cylinder. With the parallel computation capability of a GPU, our system can detect and output the knee-form contacting area in a practical time period.

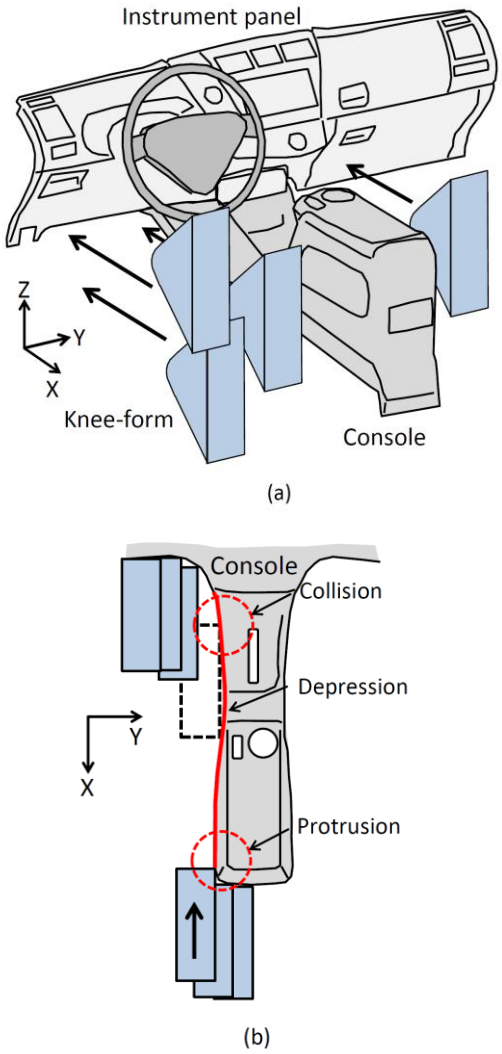


Figure 2. Limitation of our prior method [Inu17].

In this method, it is assumed that the knee-form approaches the parts from a sufficiently far position from the parts. This assumption is acceptable for detecting the knee-form contacting area on the instrument panel, however it is not appropriate for detecting the area on the console part and door panel. Figure 2 illustrates this problem. Most surface of the instrument panel faces towards the positive direction of the x-axis. Since there is no obstacles between the knee-form in a far position and the instrument panel, the approaching knee-form can directly collide any part of the instrument panel as shown in Figure 2(a).

On the other hand, most surface area of the console part and door panel is vertical and near parallel to the approaching direction of the knee-form. The knee-form can make a collision at a shallow depression in the side surface of the console part as shown in Figure 2(b). Such collision is not detected in our prior method because a knee-form approaching from a far position make a collision at a protrusion and it cannot reach the depression locating behind the protrusion.

One possible method of extracting the knee-form colliding area may be the application of 3D collision detection technology; established technologies in this field are described in [Mol99], and [Fau08] presents the use of the GPU's parallel processing technology for accelerating the collision detection. In this method, a manual inspection with physical apparatus is replaced by a virtual process with solid models in computers. Although this approach is simple and easy to implement, its cost is estimated to be very large because a tremendous number of collision detections are required, with knee-form models in different orientations and initial positions.

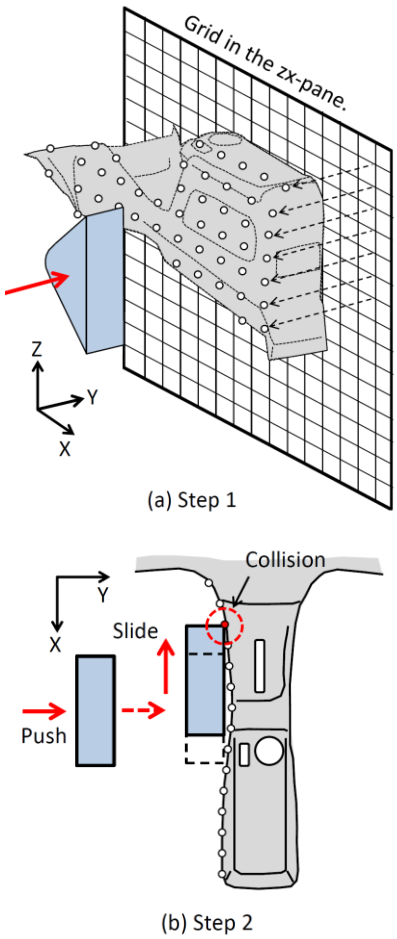


Figure 3. Outline of push-and-slide algorithm.

In this paper, we propose a novel algorithm named "push-and-slide" specialized for detecting the knee-form colliding area in the surface of the interior part which is vertical and near parallel to the knee-form approaching direction. The input data for the method consists of a polyhedral model that approximates the interior part shape. Most commercial CAD systems provide a function to output the model data as a group of triangular polygons, such as in the STL format.

Figure 3 illustrates the outline of the algorithm. In the first step, the target surface area of the part is transformed to grid-like points in a high resolution (see Figure 3(a)). The knee-form models in various positions and orientations are prepared in the second step. Each knee-form model is moved in the y-axis direction and pushed to the grid-like points. The knee-form model is then moved in the negative x-axis direction to detect possible collision with the grid-like points (see Figure 3(b)). The system finally visualizes the knee-form colliding area on the display by converting the detected collision points to a set of small polygons.

Our method has similarities to the collision detection based method, however it can realize better performance by properly using the geometric characteristic of the grid structure of the points on the part surface. In the following sections, details of our knee-form colliding area extraction algorithm are explained. In section 4, techniques for improving the performance are given. Experimental extraction result from a CAD model of a console part is given in Section 5, and we summarize our conclusions in Section 6.

3. BASIC ALGORITHM

In this section, details of the basic processing flow of our push-and-slide algorithm is explained. In the following discussion, we consider a collision analysis of the knee-form to the left side surface of the console part as an example. The same method is applicable for detecting knee-form colliding area in the other side of the console part or in the surface of the door panel by modifying some conditions, for example pushing direction of the knee-form to the part surface.

Step 1 Sampling of Grid-Like Points

The first step of the algorithm is the sampling of grid-like points on the near vertical surface of the console part. The console part is oriented parallel to the zx-plane and the left side surface of the part faces to the negative direction of the y-axis. An axis-aligned regular grid with sufficient width and resolution is prepared in the zx-plane. From each grid point, a ray is extended along the negative direction of the y-axis, and the intersection points between the ray and the left side surface of the console part are computed as

shown in Figure 3(a). The intersection point with the smallest y coordinate is selected and recorded as the surface point corresponding to the grid point. This operation is repeated for all grid points, and an approximated representation of the left side surface of the console part is obtained as grid-like points.

Such surface points are obtained by rendering a hidden-surface eliminated picture of the console part using the orthogonal projection as illustrated in Figure 4 [Mol99]. In a preparation, a local coordinate frame for the orthogonal projection is placed so that its x-axis and y-axis are aligned with respect to the x-axis and z-axis of the object's coordinate frame. The negative z-axis direction of the projection's coordinate frame corresponds to the viewing direction. Six clipping planes (*left*, *right*, *top*, *bottom*, *near*, *far* in the figure) defining the visible regions in the object's coordinate frame are given such that they confine the console part with a sufficient margin around the part so that any knee-forms contacting the part surface can be contained within.

After the preparation, component polygons of the console model are rendered into the frame buffer in an arbitrary order. During the rendering, the color and the depth value of points on a polygon corresponding to each pixel of the frame buffer is updated, and the result is the pixels with the colors of points on the visible surfaces of the console part and the entries of the points in the depth buffer.

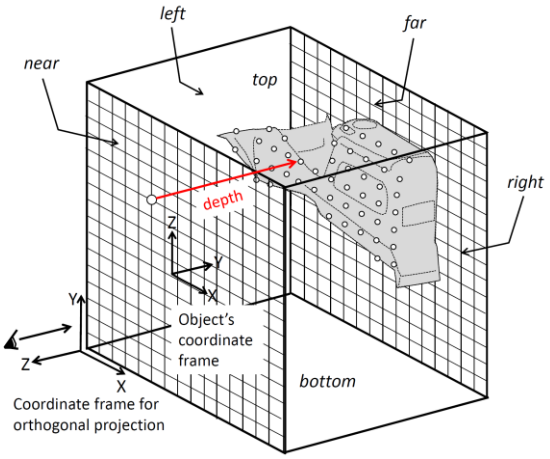


Figure 4. Depth buffer-based method for sampling grid-like points on the part surface.

Data stored in the depth buffer can be transferred to a 2D array in a C language program using a buffer manipulation function, for example glReadPixels( ) in OpenGL [Kes16]. Based on the clipping plane definition and the pixel position in the display, the x and z coordinates of the visible point at the pixel in the object's coordinate frame can be computed. The depth value at the pixel corresponds to the y coordinate of the point. The left side surface of the console part is thus derived as a set of grid-like points aligned with respect to the pixel grid of the display.

Step 2 Push-and-Slide Operation

In the definition of UNECE No.21, the knee-form can be rotated above and below the horizontal by up to 30 degrees. In our current implementation, the knee-form can be rotated in every one degree around the y-axis in the range. The rotated knee-form model is set to a certain initial position. We consider that the center point of the round corner part of the knee-form corresponds to the reference point of the knee-form. Points in the axis-aligned regular grid in the zx-plane are traced in row by row order. For each grid point, the rotated knee-form is placed so that its reference point (green point in Figure 5(a)) has the same x and z coordinates of the grid point and a sufficiently small y coordinate. The knee-form in Figure 5(a) is placed at (*i*, *j*) grid point.

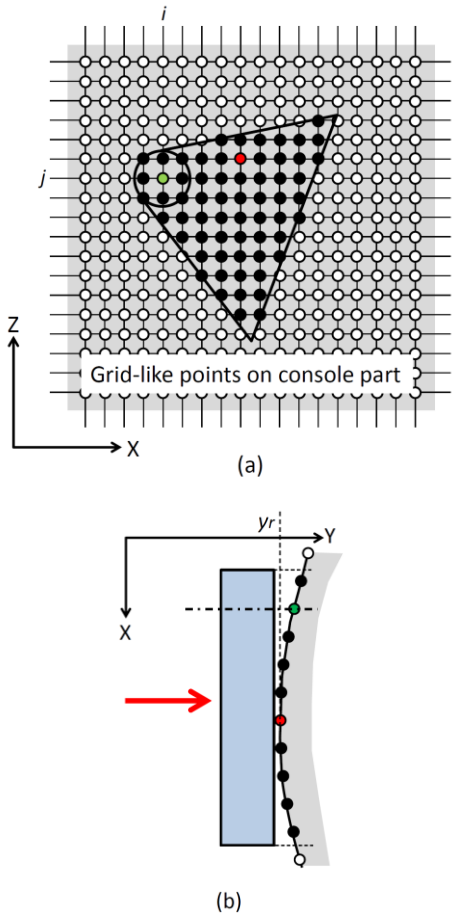


Figure 5. Determination of the knee-form position by a pushing operation.

3.1.1 Pushing Operation

The knee-form is then pushed linearly in the positive direction of the y-axis until it realizes a contact at a grid point in the left side surface of the console model. The y coordinate of the knee-form model in the

contacting condition is determined by projecting the knee-form shape to the left side surface of the console model and by checking the y coordinates of the grid-like surface points contained within the projection. In Figure 5(a), the black points represent such points within the projection. In these points, a point with the smallest y coordinate *yr* (red point in Figure 5(b)) corresponds to a point where the knee-form realizes a contact after a pushing operation.

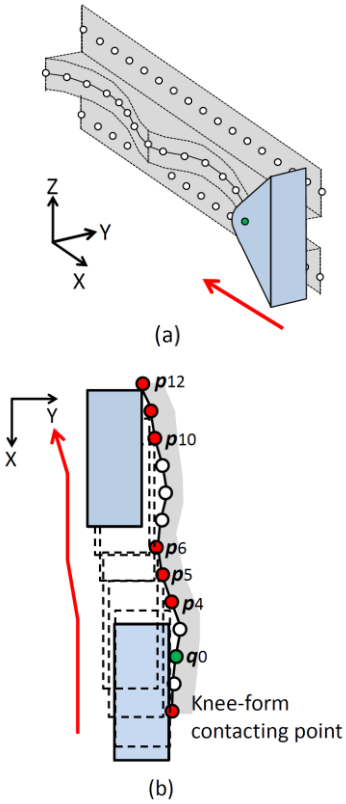


Figure 6. Sliding operation for detecting colliding points in a single row.

3.1.2 Sliding Operation

After the placement of the knee-form model on the console surface, the model is moved linearly in the negative x-axis direction until it collides to the console surface on its front surface. Since we represent the surface of the console part as a set of grid-like points, the collision between the knee-form in the sliding motion and the grid-like points is actually checked.

The knee-form continues the sliding operation after detecting the first collision for finding other colliding points in the part surface. Figure 6 illustrates our strategy. In this figure, a simple narrow and wave-like protrusion is given as the side surface of the console part. A single row of points is placed on the narrow "passage" on the protrusion. Other surface points are placed on the bottom part of the surface as shown in Figure 6(a). The knee-form model is already placed on the surface of the protrusion as its reference point locates above a grid point $\boldsymbol{q}_0$. Because of this placement, the knee-form can collide only points in the row on the protrusion in the sliding motion.

The knee-form starts sliding in the negative x-axis direction to detect a first colliding point $\boldsymbol{p}_4$ as shown in Figure 6(b). After the detection, the knee-form continues the sliding operation along the leftward curve to detect new colliding point $\boldsymbol{p}_5$ on the same row of the grid-like points. After colliding $\boldsymbol{p}_6$ at the peak point of the curve, the knee-form stops the curve-following because motion along the rightward curve causes gauges between the knee-form and the part surface. Otherwise, it starts the straight motion along the negative direction of the x-axis from $\boldsymbol{p}_6$ to detect new colliding point $\boldsymbol{p}_{10}$ after getting over a depression between $\boldsymbol{p}_6$ and $\boldsymbol{p}_{10}$. The knee-form starts the leftward sliding operation again from $\boldsymbol{p}_{10}$ until it reaches the edge of the console part.

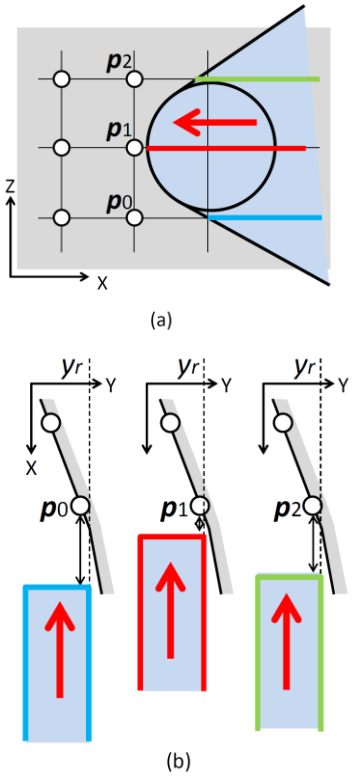


Figure 7. Detection of the colliding points in multiple rows.

Side surface of the actual console part generally has much complex curved shape. Consider rows of grid-like points locating in front of the knee-form. In Figure 7(a), three such rows are illustrated. These rows are easily determined based on the range of knee-form shape in the z-axis direction. Figure 7(b) illustrates three sections of the knee-form and the part surface obtained by horizontally slicing them along blue, red, and green lines given in Figure 7(a).

Y coordinates of the points on the rows and distances between the points and the front surface of the knee-form in the x-axis direction are measured. The first colliding point in the sliding motion of the knee-form is a point whose y coordinate is smaller than the y coordinate *yr* of the side surface of the knee-form contacting to the console part, and the distance between the point and the front surface of the knee-form is the smallest. As shown in Figure 7(b), y coordinates of points $\boldsymbol{p}0$, $\boldsymbol{p}1$, and $\boldsymbol{p}2$ locating in front of the knee-form are all smaller than *yr*. In these points, the knee-form collides to $\boldsymbol{p}1$ because it is the nearest point to the front surface of the knee-form in the x-axis direction. The knee-form then collides to $\boldsymbol{p}2$ and $\boldsymbol{p}0$ in its following sliding motion.

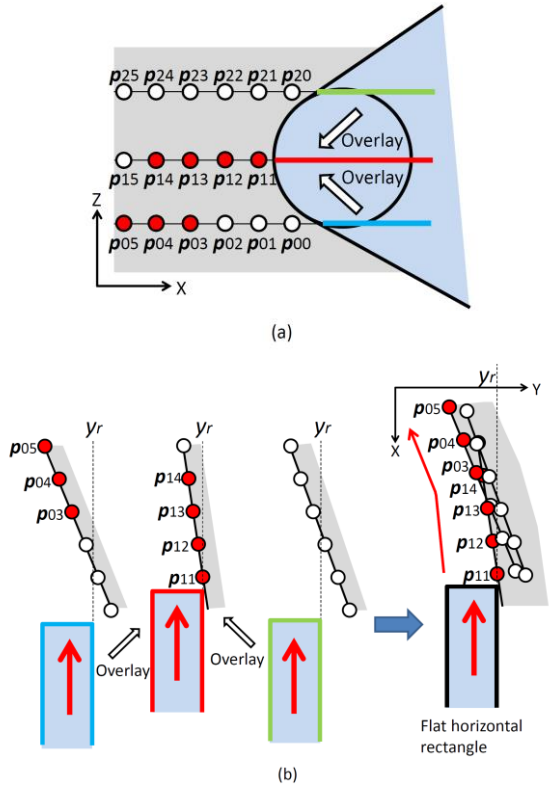


Figure 8. Sliding operation using sorted points.

To detect all colliding points in multiple rows in the sliding motion, we realize an algorithm based on the sorting operations of grid-like points. Figure 8(a) illustrates 3 rows of points existing in front of the knee-form. The knee-form realizes collisions at $\boldsymbol{p}11$, $\boldsymbol{p}12$, $\boldsymbol{p}13$, $\boldsymbol{p}14$, $\boldsymbol{p}03$, $\boldsymbol{p}04$ and $\boldsymbol{p}05$ in its sliding motion in the negative x-axis direction.

After placing the knee-form model on the console surface by a pushing operation, the grid-like points locating in front of the knee-form model are collected. For each point, the distance between the point and the front surface of the knee-form in the x-axis direction is measured, then the points are sorted according to their distances. The sorted points correspond to the result of overlaying the rows of points in front of the knee-form so that the 3 sections of the knee-form are superimposed to be a flat horizontal rectangle as shown in Figure 8(b).

The sorted points are scanned to find out a point whose y coordinate is smaller than *yr* (see point $\boldsymbol{p}11$ in Figure 8(b)). This point corresponds to the first colliding point of the knee-form to the surface. The sliding operation is executed by using the sorted points to collect the knee-form colliding points. As shown in Figure 8(b), the sorted points are traced along the leftward curve to collect colliding points $\boldsymbol{p}12$, $\boldsymbol{p}13$, $\boldsymbol{p}14$, $\boldsymbol{p}03$, $\boldsymbol{p}04$ and so on.

The pushing and sliding operations mentioned above are repeated for all the knee-form placements above the grid-points in the zx-plane to determine the colliding points for the knee-form in a specific orientation. The final result is obtained by repeating the operation for each rotation angle in the range between -30 degree to 30 degree. The obtained colliding points are properly connected based on the adjacency information of the grid, and converted to an equivalent polyhedral surface representing the knee-form colliding area on the part. This result is used for displaying and for outputting to the data file in STL format.

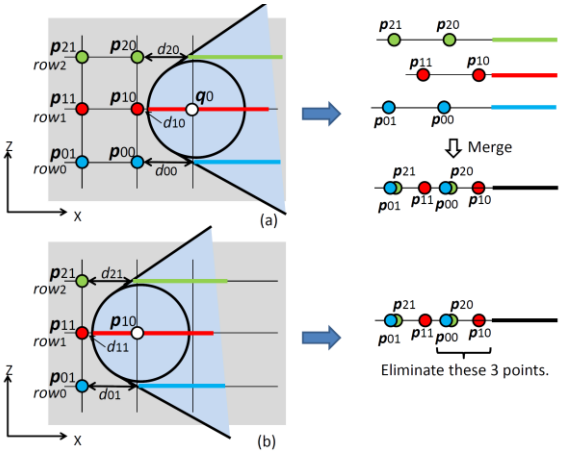


Figure 9. Merging of 3 rows of points for obtaining the sorting result.

4. PERFORMANCE IMPROVEMENT

In our method, sorting operation in the sliding operation is the most time consuming task. Since points on the part surface is defined based on the regular square grid in the zx-plane, horizontal distance between two adjacent points in a row is equal for all rows. Sorting is thus realized by a simple merging operation of rows of points. Consider 3 rows of points as shown in Figure 9(a). For each row, select a point closest to the front surface of the knee-form. They are $\boldsymbol{p}00$ for *row*0, $\boldsymbol{p}10$ for *row*1, and $\boldsymbol{p}20$ for *row*2. Their distances to the knee-form in the x-axis

direction are d_{00}, d_{10}, and d_{20} respectively where $d_{10} < d_{20} < d_{00}$. In this case, sorting result of points in the 3 rows is obtained by merging 3 rows of points as $\{\boldsymbol{p}_{10}, \boldsymbol{p}_{20}, \boldsymbol{p}_{00}, \boldsymbol{p}_{11}, \boldsymbol{p}_{21}, \boldsymbol{p}_{01}, \ldots, \boldsymbol{p}_{1i}, \boldsymbol{p}_{2i}, \boldsymbol{p}_{0i}, \ldots\}$.

In our knee-form collision analysis, the knee-form model in a same orientation is positioned for each grid point of the regular square grid in the zx-plane in row by row order. In each row, the knee-form is placed on a grid point in the row from right to left order. In this case, the cost for obtaining the sorted points can be further reduced by reusing the sorting result obtained for the prior collision detection. Figure 9(a) illustrates the sorted points obtained for a knee-form at grid point $\boldsymbol{q}_0$. After collecting the knee-form colliding points by sliding the knee-form from this point, the initial position of the knee-form is shifted to the left side neighbor point of $\boldsymbol{q}_0$ (which is $\boldsymbol{p}_{10}$ as shown in Figure 9(b)). In this case, the sorted points computed for the knee-form at $\boldsymbol{q}_0$ can be reused. Because of the shifting operation from $\boldsymbol{q}_0$ to $\boldsymbol{p}_{10}$, first 3 points $\boldsymbol{p}_{10}$, $\boldsymbol{p}_{20}$, $\boldsymbol{p}_{00}$ in the sorted list are now useless, therefore they are eliminated from the list. Other points in the sorted list is applicable for detecting colliding points with the knee-form in the sliding motion from $\boldsymbol{p}_{10}$.

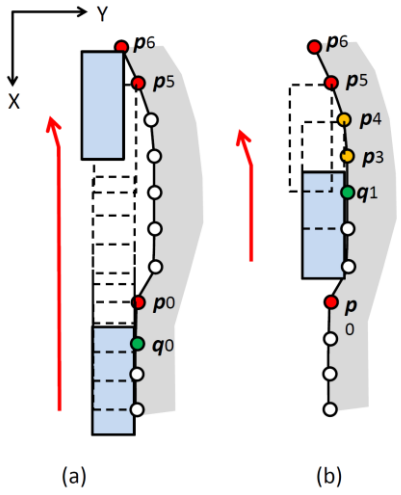


Figure 10. Cancellation of sliding operations.

In our algorithm, some sliding operation can be canceled. Figure 10(a) illustrates a detection process of knee-form colliding points using the sorted points. The knee-form is firstly placed at point $\boldsymbol{q}_0$, then red points are detected as the knee-form colliding points by tracing the sorted points. In the following operation, the initial position of the knee-form is shifted to $\boldsymbol{q}_1$ as shown in Figure 10(b). From this position, the knee-from starts the sliding operation and new colliding points which are yellow points in the figure are detected. The knee-form then reaches point $\boldsymbol{p}_5$ which is a colliding point already detected in the prior sliding motion as shown in Figure 10(a). At this point, the following sliding operations can be canceled because no new colliding points can be found in the operation.

5. COMPUTATIONAL EXPERIMENTS

The proposed knee-form colliding area computation system was implemented using Visual C++ and OpenGL, and a series of computational experiments were performed using an Intel Core i7 Processor (2.6 GHz) with 16 GB memory and an NVIDIA GeForce GTX-960M GPU. We applied the system to several console part models provided by a software company and an automobile company, and the proposed system was found to successfully compute the knee-form colliding area for all parts in 40 to 50 minutes, with the time varying according to the resolution of the grid used in the sampling of the surface points.

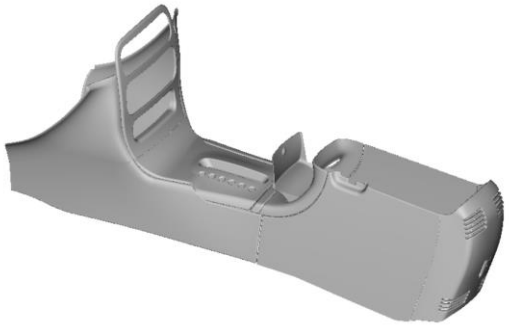


Figure 11. Sample part.

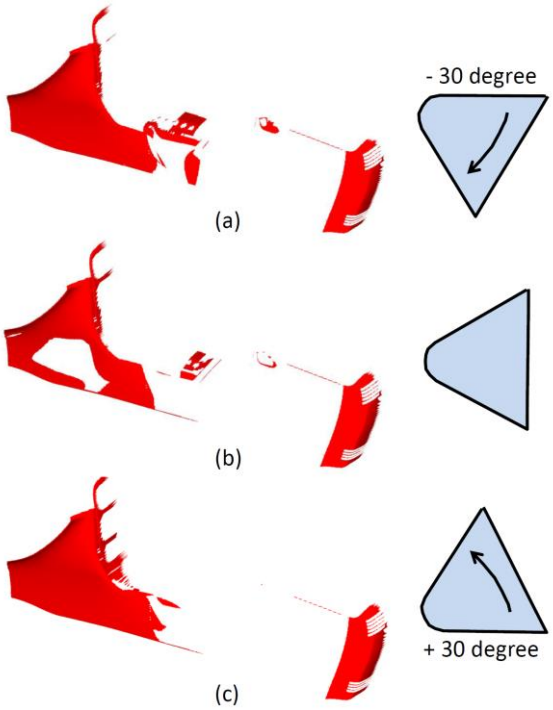


Figure 12. Extraction result using a knee-form in different orientation.

For the purpose of maintaining confidentiality, the computation result for only one sample is shown here. Figure 11 illustrates the sample console model with 369,408 polygons. The left side surface of the part is converted to a grid-like points in Step 1. In this computation, a depth buffer with a grid resolution of 1538 x 801 was used. The grid size, which corresponds to the computation accuracy of our method, was 1.24 mm. In Step 2, the knee-form colliding area is extracted by repeating the push-and-slide operation with the knee-form model with different rotation angle between -30 degree to 30 degree for every one degree interval. 61 extraction results with different knee-form orientations are obtained. Figure 12(a) – (c) show three results of the knee-form colliding area on the part for knee-forms with rotation angle -30 degree, 0 degree, and 30 degree, respectively. Figure 13 illustrates the composition of the 61 results.

Our proposed system was found to require 2519.92 seconds for obtaining the result given in Figure 13. The system can output the polygons of the detected area in STL format, and the automobile designer can verify compliance with the UNECE regulation by superimposing the knee-form colliding area on the CAD model of the console part.

Figure 13. Composition of 61 extraction results.

6. CONCLUSIONS AND FUTURE WORKS

In this paper, we propose a novel algorithm named "push-and-slide" for detecting the knee-form colliding area in the surface of the interior part which is vertical and near parallel to the knee-form approaching direction. This algorithm is developed for automating the UNECE safety regulation inspections of the automobile interior parts.

In the first step of the algorithm, the target surface area of the part is transformed to grid-like points in a high resolution. The knee-form models in various positions and orientations are prepared in the second step. Each knee-form model is pushed to the grid-like points. The model is then moved in the negative x-axis direction to detect possible collisions with the grid-like points on the part surface. In the sliding operation, sorting of the grid-like points locating in front of the knee-form is necessary. By using the grid structure of the points, the costly sorting operation is replaced to a simple merging operation of points.

Our algorithm relies on the grid-based representation of the surface shape of the interior part. We are planning to realize much accurate computation by using a higher resolution grid, which inevitably demand more computation cost in the collision detection. To reduce the cost, use of the parallel processing capability of GPU is considered. We are preparing a field test of the system in an actual automobile design process, and further improvements based on comments and requests from the designers will be reflected in our future work.

7. REFERENCES

[CAVA] CAVA Systems, http://www.edstechnologies.com/index.php/component/content/article/86-solution/cad-quality-check/135-cava.html, EDS Technologies.

[CUDA] CUDA Compute Unified Device Architecture Programming Guide, http://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_ Guide.pdf, NVIDIA.

[Fau08] Faure F.; Barbier S.; Allard J.; Falipou F.: Image-based collision detection and response between arbitrary volume objects, Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2008, 155-162. dx.doi.org/10.2312/SCA/SCA08/155-162

[GAR] Global Auto Regulations, https://globalautoregs.com/unece, United Nations Economic Commission for Europe.

[Inu15] Inui M.; Umezu N.; Kitamura Y.: Visualizing sphere-contacting areas on automobile parts for ECE inspection, Journal of Computational Design and Engineering, 2(1), 2015, 55-66. dx.doi.org/10.1016/j.jcde.2014.11.006

[Inu17] Inui M.; Nakano S.; Umezu N.; Asano T.: GPU-based visualization of knee-form contact area for safety inspections, Computer-Aided Design and Applications, 14(3), 2017, 356-365, dx.doi.org/10.1080/16864360.2016.1240456

[JP06] Japanese Patent [P2006-277304A], http://www.j-tokkyo.com/2006/G06F/JP2006-277304.shtml, Toyota Motor Corporation, in Japanese.

[Kes16] Kessenich J.; Sellers G.; Shreiner D.: OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.5, Addison-Wesley Professional, USA, 2016.

[Li11] Li W.; McMains S.: Voxelized Minkowski sum computation on the GPU with robust culling, Computer-Aided Design, 43(10), 2011, 1270-1283. dx.doi.org/10.1016/j.cad.2011.06.022

[Mol99] Moller T.; Haines E.: Real-Time Rendering. A. K. Peters Ltd., Natick MA, USA, 1999.

[Yam11] Yamazaki S.; Baba T.; Umezu N.; Inui M.: Fast safety verification of interior parts of automobiles, Proceedings of IEEE International Conference on Mechatronics and Automation (ICMA), 2011, 1957-1962. dx.doi.org/10.1109/icma.2011.5986280

CoUIM: Crossover User Interface Model for Inclusive Computing

Abdullah Almurayh
University of Colorado
Colorado Springs
aalmuray@uccs.edu

Sudhanshu Kumar Semwal
University of Colorado
Colorado Springs
ssemwal@uccs.edu

ABSTRACT

Persons with disabilities can face considerable challenges accessing many computing systems, such as cloud computing. We created six low-cost user interfaces using: keyboard-based, touchable, speech-based, touch-less gesture, tactile, and then combined them all in one user interface termed Crossover User Interface Model (CoUIM). We measured inclusiveness, error occurrence, user performance, and user satisfaction though an IRB approved study of twenty-nine participants. We chose Xen cloud platform to evaluate our research. We focused on three groups of users: persons with no disability, persons with blind and visually impairment (B/VI), and persons with motor-impairment. When we combined several interactions in one user interface, results improved for persons with disability. Using CoUIM improved inclusiveness, error rate, user performance and even user satisfaction. Persons with motor disability needed a little more time to complete the same tasks in our study. In particular, we show that persons with blind and visually impairment (B/VI) can compete on equal footing with their sighted peers based on error rate and time to complete the tasks using CoUIM.

Keywords: User Interfaces, Disability Applications, Usability Study.

1 INTRODUCTION

A disability is defined as a physical or mental impairment that significantly confines a person to minor life activities and limits them from major desirable activities [2, 6, 7, 11, 24]. Human computer interaction (HCI) can play a pivotal role in enabling users with various disabilities to manage computer systems. Our goal is to enhance this interaction with a computing system through new interactive interfaces that are geared towards users with disabilities, yet can be used by anyone. This research investigates the effectiveness of creating several crossover [26] user interfaces for users with different (dis)abilities to manage clouds, our chosen application. Our idea is that a user interface can have different interaction methods, and method chosen could be selected by the user based on their (dis)abilities. Additionally, our ideas could be extended to manage other applications, other than clouds, thus creating user interfaces on demand [3] by using the same familiar set of interactions. The cloud management supported by our interfaces are the following commands: start, reboot, shutdown, suspend, resume, reset.

Our research seeks to provide the same level of accessibility and usability for everyone. Our hypothesis is that anyone can manage the cloud if we provide variety of suitable user interfaces for both a person with disability or not. We were statistically able to show that our hypothesis is true. We evaluate our interface based on inclusiveness, accuracy (error occurrence), user performance, and user satisfaction. Our interface was tested by people with visual and motor impairment.

2 PREVIOUS WORK

Any impairment does not have to restrict the person with disablity from performing a major life activity [10, 26]. The U.S. Department of Labor (DOL) states, "*Having a disability can also affect your employment status–such as a job loss or reduction in hours–in a way that could affect your group health benefits, depending on the plan rules*" [11]. People with impairments can perform well in jobs. For example, in Winston-Salem, NC, a factory allows persons who are blind or with visual impairment to make clothing for colleges and universities [16].

Cowan [8] and Mann define assistive technology as a general term that comprises technologies, tools, products, facilities, systems, and mechanisms used by people who have impairments such as the persons with disablity, persons with blindness and visual impairment, and other disabilities [18]. Assistive technology can also be considered as tools to accommodate functional limitations [5]. IDEA describes assistive technology as

any equipment or customized product that is meant to improve people's capabilities [9]. Assistive technology varies from simple inexpensive tools to expensive advanced electronic devices, which are classified as back-end tools (i.e. optical character recognition tools), input (i.e. speech recognition), and output (e.g. screen readers) [20]. We define assistive technology as any shape and form of interactivity between a user with a disability, and a computer application.

Assistive technology can be categorized into access and spatial controls, localization, moving, listening, communication, computer-based instruction, computer applications, vision, recreation, and special care [2, 4, 22]. Assistive technology may be stand-alone such as Mobility Smart Better-Grip Reacher, or embedded hardware such as haptic feedback glove for the blind Another form of assistive technology is computer software, which could be embedded (e.g. Speech recognition) or stand-alone like Braille Translator [14]. Braille Plus 18 is an example of Assistive technology product invented by APH[1]. The Braille Plus 18 is essentially aimed to assist persons with visual impairment. It allows a person with visual impairment read books, write assignments, scan documents, search the web, keep track of appointments, find directions, record lectures, listen to podcasts, use GPS Navigation, run Android apps, use camera, support SD card slot and USB port, connect through wireless, send text messages, make phone calls, and have many other features. However, it can be extremely expensive, approximately $3,599.00. Everyone agrees that assistive tools should not be pricy and should be affordable [20].

Unfortunately, there are different concerns and issues when using assistive technology in real world. Users face social, cultural, design, privacy, security, compatibility, and other limitations. Incorporating social acceptance into the design of assistive technology is necessary; assistive technology should have appealing design [23]. Similarly, the proliferation and wide usage of assistive technology can be affected by cultural differences including life style, language, economy, and diversity [2]. Cultural aspects must be considered in the design of assistive technology for crossover applications. Moreover, as cultural changes may occur in some communities, it is important to occasionally update the existing assistive products and tools to convey these changes. Some assistive tools depend on the structure and the design of existing sources. Having poor design affects the functionally of assistive technology as changes are not easily incorporated. Screen readers, for example, do not read by looking at web pages; however, they indicate text phrases through the HTML code and announce whatever is found. If the screen readers misinterpreted the HTML code, a meaningless sound is played. Another related issue is privacy as the users concern about their private information being read [19]. Assistive technology may disclose personal and confidential information without permission or detection; putting users at tremendous risk.

In term of security, there are two opposite perspectives of security challenges against assistive technologies: vulnerability and those related to settings. Vulnerability affects the underlying system if the assistive tool does not comply with high security level. Setting related issues (i.e. rules, restrictions, permissions, privileges, policies, or firewalls) are set to the underlying system or resource. Such settings can restrict assistive tools to not perform efficiently or prevent them from accessing the resources permanently. Adobe Acrobat software, for example, allows the user to forbid some parts of a PDF file from being copied, printed, extracted, commented on, or edited. Screen readers, on the other hand, will not be able to extract the documents text in order to transform it into a spoken format [1]. Hence, the user who creates the PDF file must be aware of this situation as well as users who depend on screen readers for accessibility.

The compatibility challenge is demonstrated by whether content works well with different assistive tools (e.g. various screen readers) on different platforms (i.e. software and operating systems). Google, for instance, lists supported assistive technology with Chrome web browsers [13]. The considerations mentioned above should be taken into account as we design assistive technology.

JAWS (Job Access with Speech) is one of the most widely used screen readers which is developed for persons with visual impairment and blindness. JAWS is a Windows based screen-reading application that reads text-based content on programs and the Internet (www.freedomscientific.com). Many schools and universities provide screen readers, such as JAWS, to accommodate the standards of accessibility. When we checkes, JAWS is priced at $1,095 for the professional edition and $895 for the standard edition. It could be considered expensive and lacks multilingualism [2]. Screen reader software such as JAWS still lacks some other features (i.e. a screen zoom function, only windows-based version, and reading when mouse is over an item) because it is aimed for those who have no vision at all [25].

Duxbury Systems has produced a Windows-based Braille translation application called Duxbury Braille Translator [12]. This is an example of Braille translators that translate some text or entire file into Braille cells, and sends it to a Braille embosser or printer to produce a hard copy in Braille script of the original text or any display. A drawback that may limit the

[1] Website: www.aph.org.

distribution and the wide usage of Duxbury Braille Translator is the high-priced license that starts from $595 [2]. On the other hand, there are free Braille translators such as Braille Translator service provided by BrailleTranslator.org. Although it is easy to access, free to use, and simple to convert text to braille, much work remains to be done to support more functions, languages, and applications. Still more work can be done as screen readers are not completely fulfilling the users' expectations, specially they do not provide facilities for Xen-management, which is our application.

3 VALIDATION STUDY AND FRAMERWORK

Our focus is on application programming interfaces (API) for building cloud-based applications [21]. A computing system cannot be effectively utilized without a proper user interface (UI) because some of these system are not usable [15] by everyone. Meeting the needs and preferences of all users is difficult. Our interface provides several modalities at the same time. Our API allows developers, or even users, to manipulate the interactive user interface so that various inputs in different forms and shapes can be transformed into a unified form that the API accepts. The Crossover User Interface Model (CoUIM) is a two-part framework that implements collaborative user interface development models for inclusive computing. First part is to use multiple interactions to map user intention to a command, and second part is to execute that command. The beneft of our model is that we can easily replace the application and use the *same framewrok* of commands to map a new application. This helps to avoid redesigning existing computing systems.

Our validation study is based on following characteristics of our interface:

- Mouse, is used for selecting and clicking on entities. This works for sighted-users and blind users since the type of interactions allows visual and audible feedback.
- Remote Controller, is used as a mobile mouse for selecting and clicking on entities from distance. It works remotely just like the method (A) where any clicked or hovered entity is seen and heard.
- Leap Motion, is used for pointing at entities and taping on them from distance. So far, this method requires vision as audible feedback is not available.
- Mouse Pad. is used for moving mouse cursor around the screen and hover entities and clicking on them. It supports users' preferences. It also, allows audible feedback and can be seen.
- Tangible dots, are stuck on a specific key that has a function. It is a helpful tangible tool that blind and visually impaired users can utilizes especially those who are not spatial learners and are unfamiliar with the keyboard layouts.
- Keyboard, is used as an alternative emitter for the mouse. Users can use Tab and Enter buttons to interact with the interface.
- Speaker, produces audible feedback.
- Large Obvious Cursor, helps visually impaired users and those who interact with the interface from distance to locate the cursor easily.
- Screen, displays the components of the user interface visually to the user to receive information. In addition, it can be touchable screen; the user can use it as emitter (or display) device.

A total of 29 participants participated in our University IRB (Internal Review Board) approved study. There were eleven participants with no disability. Nine participants were either with some form of visual impairment or were blind. Nine had some form of physical impairment. Xen Cloud Platform was used for all testing.

4 MULTIPLE INTERACTIONS

We implemented a total of six approaches and they were all tested. Five of these approaches are: (a) Typing based, (b) Touchable Mobile interface, (c) Speech Conversation, (d) Touchless interaction, and (e) Straight forward input board. The sixth approach, which includes all the previous five types of interactions in one single interface, is an example of the Crossover User Interface model (CoUIM) which allows people with visual and motor impairment to complete the tasks asked by people with no impairment.

4.1 Typing-based Cloud Management

This contribution is conventionally used in many computing systems where keyboard is used. It is also known as the command line interface. The user needs eyesight to find the keyboard keys and needs fingers to type, making it highly inconvenient for people with visual impairment. The Computer requires a monitor to display output and speakers to voice results (Figure 1).

4.2 Touchable Mobile Interface for managing the Cloud

This contribution is unique in that it uses a smart phone for input and visual and auditory feedback for output. The user touches the screen to scroll the information. Touch includes mouth stick, head stick and other similar devices for people with motor disability. The user needs vision and/or hearing and the ability to touch the screen of the device. The user interface is the touch screen for input and voice for input and output.

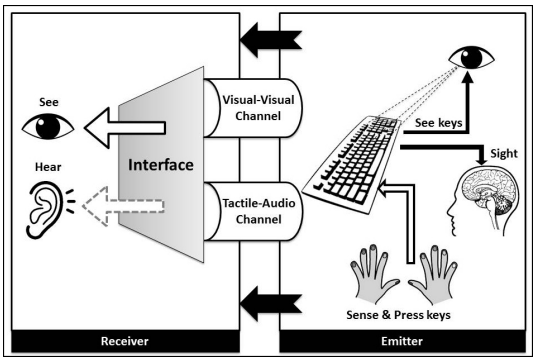


Figure 1: High level Design of typing-based cloud management.

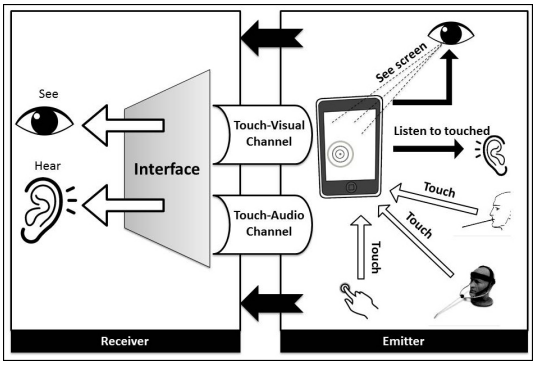


Figure 2: High level Design of Touchable Mobile Interface for managing the Cloud.

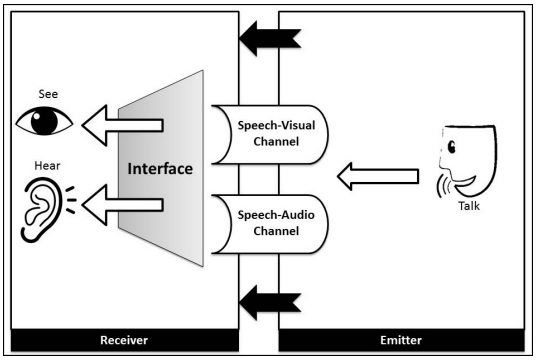


Figure 3: High level Design of Managing the Cloud via Human-computer Conservation..

4.3 Managing the Cloud via Human-computer Speech-conversation

This work depends completely on speech-based interaction to the point that human and machine can converse. The user interface is a computer equipped with the ability to receive and output speech. The computer requires a microphone to interface with the speech recognition. The user receives spoken and/or visual information. The user interface is a screen with text display and a microphone or a headset.

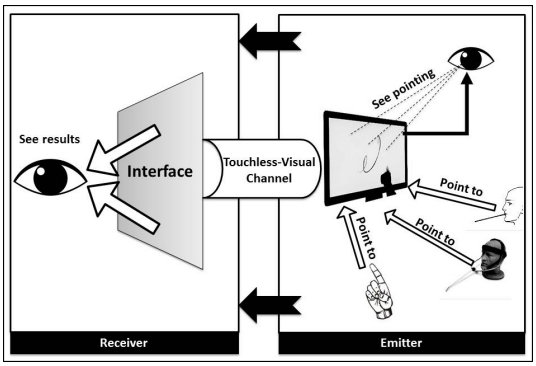


Figure 4: High level Design of Touchless Interaction for Managing the Cloud.

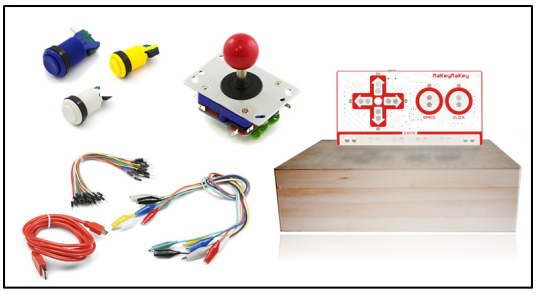


Figure 5: The tools and materials needed for building the MaKey MaKey-based Custom Broad.

4.4 Touchless Interaction for Managing the Cloud

The User Interface is a USB device attached to the machine to track user's movement. The user points to areas on the screen with a finger or with a mouth stick, head stick or similar device. The Computer requires a monitor and a USB port. The user needs vision to see where he/she is pointing and to view the results. The User Interface is a colored circle on the screen that will change colors as the user points and moves the pointer. The Computer requires a monitor to display input and output.

4.5 Managing the Cloud via straight Forward Input Board

MaKey MaKey[2] is an invention kit which refers to the combination of the words Make and Key. The machine assumes inputs come from MaKey MaKey is a regular keyboard and mouse. Figure 5 indicates the collection of objects and tools that we needed to implement to custmize for our application. Collection of objects include: a wooden box, a Joystick, leads, tactile buttons, a USB cable, and a MaKey MaKey tool kit underneath.

Figure 6 shows the architectural design of the MaKey MaKey-based custom board. An earth lead is connected to all buttons and the joystick device. As the user

[2] Available on: http://www.makeymakey.com.

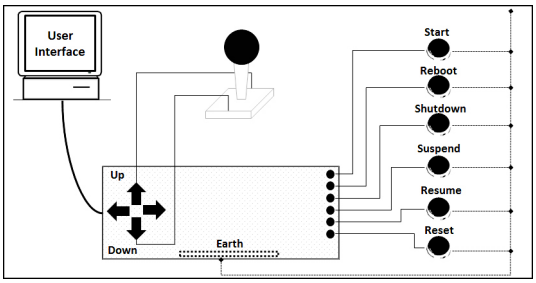


Figure 6: The architecture of the MaKey MaKey-based custom broad.

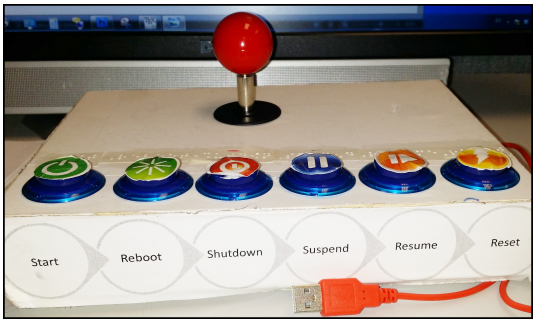


Figure 7: The final look of the built MaKey MaKey-based custom board.

presses a button, it sends a unique signal that the user interface illustrates as a function. Therefore, when the user pushes the joystick up/down the user interface navigates through the virtual machines. This custom board is connected to the computer via a USB cable. Figure 7 exhibits the final look of the built MaKey MaKey-based custom board. It has labels, signs, and Braille labels so that the users can determine the functions based on their (dis)abilities.

4.6 CoUIM: Crossover User Interface Model for Managing the Cloud

This contribution combines different interaction methods to improve the inclusiveness (Figure 8). The user can now interact with the machine using either vision and/or hearing, and tactile sense. The user can touch, gestures, tact, press keys, and click mouse to interact with the computing system based on what they feel comfortable with. The computer requires a display to reveal the interface and audio device to inform the user verbally about the occurrence. Monitor, speakers or headset provide feedback to the user. Results can be simultaneously presented using visual and audio cues.

5 RESULTS BASED ON QUANTITATIVE DATA ANALYSIS

The CoUIM approach that allows multiple interaction channels has excellent inclusiveness as our results show (see Figure 9). Table 1 indicates that all participants were 100% able to utilize this approach using different interaction methods to accomplish the required tasks.

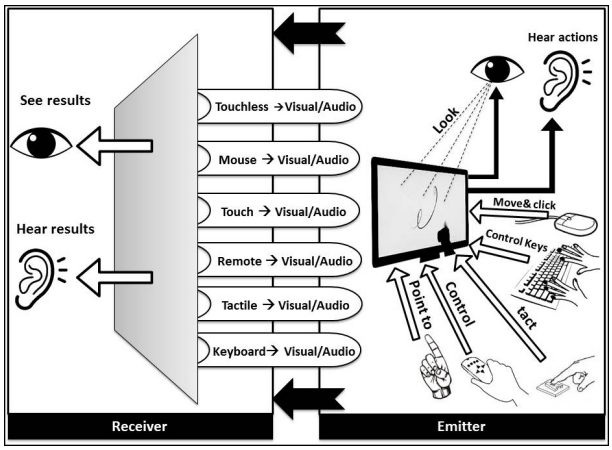


Figure 8: High level Design of Crossover User Interface for Managing the Cloud.

Feasibility: Figure 10 shows how obviously all groups had the same opportunity of utilizing this approach – each group reached 100% effectiveness. While participants with no disability, and participants with motor disabilities were able to use touchless interaction and vision to receive information, most of the participants with blindness and visual impairment (B/VI) used tangibles, mouse, and keyboard to emit and speakers to receive the verbalized results.

Error Occurrence: In terms of error rates, excellent results were obtained as indicated in Figure 11. Users with B/VI were more accurate in completing the tasks precisely– this group of users had a *zero* error rate. Dependence on verbalized feedback helped the B/VI users to be more accurate compared to the other two groups. Hand and finger gestures provides natural interactions with user interface [17]. Nevertheless, finger identification and hand gesture recognition accuracy still needs more improvements[17] in touchless interface. This problem influenced users' gesture-based interaction accuracy so that non-disabled users made 0.4 error per users whereas users with motor disabilities made 0.9 error per users on the average. Although the interactions of both groups were affected by the unperfected gesture recognition accuracy, users with motor disabilities struggled because of their motor impairments.

User performance: Although the users with B/VI were more accurate, they had some latency where each user with B/VI could take as an average of 176.7 seconds to accomplish the required tasks. While non-disabled users used gesture-based interaction, they needed at an average of 153.5 seconds; so they performed more efficiently compared to the B/VI users. The main reason of this difference is that a B/VI participant needed additional time to navigate through the interface depending on tactile and audible interaction. Users with motor disabilities needed an average of 234 seconds to accomplish the tasks because of their motor difficulties. Fig-

ID	*Disability*	*Inclusiveness*	*Error*	*Time*
I	B/VI	1	0	229
K	B/VI	1	0	172
L	B/VI	1	0	339
M	B/VI	1	0	128
O	B/VI	1	0	131
P	B/VI	1	0	158
Q	B/VI	1	0	139
R	B/VI	1	0	137
T	B/VI	1	0	157
A	M	1	2	289
AA	M	1	1	155
C	M	1	3	190
S	M	1	0	180
V	M	1	0	321
W	M	1	0	279
X	M	1	0	126
U	M/VI	1	1	329
Y	M/VI	1	0	153
AB	Non	1	0	115
D	Non	1	0	163
E	Non	1	1	150
F	Non	1	0	132
J	Non	1	0	174
B	Non	1	1	113
G	Non	1	1	160
N	Non	1	0	142
AC	Non	1	1	127
H	Non	1	0	223
Z	Non	1	0	190

Table 1: The overall results of testing the crossover user interface for cloud management. Disabilities: M, motor disabilities; B/VI, blind and visually impaired; Non, non-disabled.

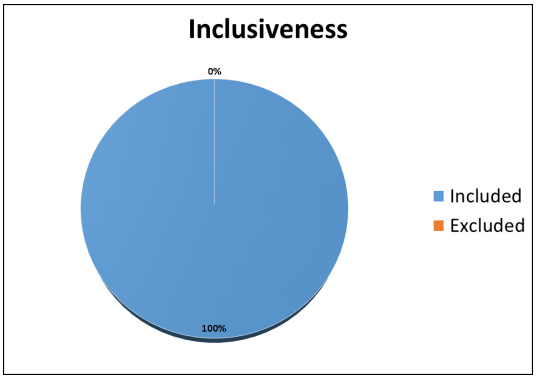


Figure 9: The percentage of included users in the crossover user interface for cloud management.

ure 12 indicates the user performance averages of each group.

5.1 Results based on Qualitative Data Analysis

The results were as shown in Table 3, each criterion has different feedback. However, since not all groups utilized the same interaction method, the qualitative results were categorized based on the groups. Most of the persons with blindness and with Visual Impairment (B/VI) utilized tactile interaction methods for requests. Some participants with visual impairment, who were

Group Of users	*Inclusiveness*	*Error*	*Time (sec)*
B/VI	100%	0.0	176.7
M	100%	0.8	224.7
Non	100%	0.4	153.5

Table 2: The averages of the results of testing the crossover user interface for cloud management. Disabilities: M, motor disabilities; B/VI, blind and visually impaired; Non, non-disabled.

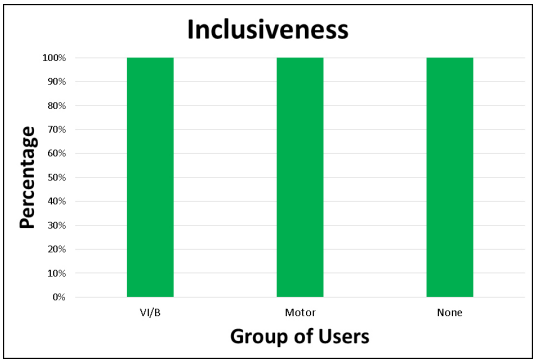


Figure 10: The percentage of included per group users in the crossover user interface for cloud management.

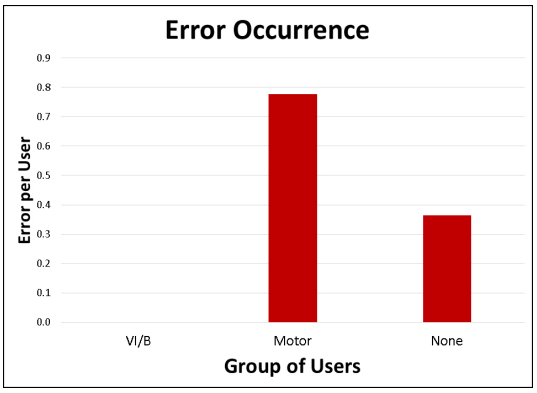


Figure 11: The error occurrence per user for each group using the crossover user interface for cloud management.

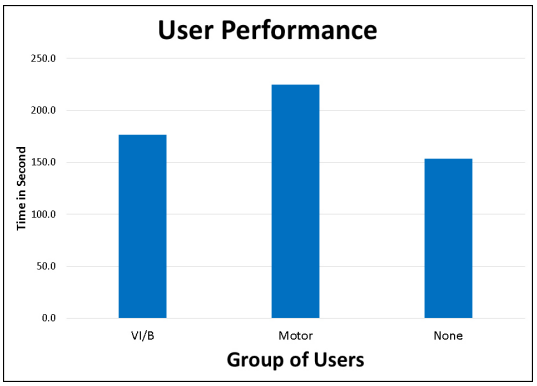


Figure 12: The averages of time per group needed to accomplish the tasks using the crossover user interface for cloud management.

not completely blind, used the mouse with audible feedback for navigating and emitting requests and receiving spoken information. Therefore, not only did the crossover user interface enable them in computing, but also it gave them various options to utilize in this group of users based on differences in degree of blindness. This means that participants with blindness and visual impairment conveyed their preferences. For example, a person with visual impairment who could utilize tangibles could also use mouse as an input. As shown in Table 3, participant have different opinions and experiences based on what they prefer to use: Strongly Disagree (SD), Disagree (D), Neutral (N) , Agree (A), Strongly Disagree (SD).

5.2 Statistical Significance

In terms of ***error occurrence***, the inclusion of group of participants with blindness and visual impairment boosted the statistical significant difference from ($p = 0.19, P > 0.1$) to ($p = 0.07, 0.05 < P < 0.1$). Two reasons influenced the results, they were:

- The sample size increased from $18 \rightarrow 29$ participants.
- The error rate was obviously *decremented* from $0.6 \rightarrow 0.38$.

The t-Test results showed that including a set of users who performed the experimental test with extreme preciseness enhanced the statistical significance of error occurrence. We were pleased that a previously excluded motor-impaired participant was included in the present approach (see Participant "*Y*" in Table 3). The inclusion of this user enlarged the size of the sample by one person. The main point here is that this user was able to perfrom the tasks with zero error occurrence. Inclusion of this participant enhanced the level of accuracy differences between two groups.

The t-Test results showed that B/VI group accuracy is significantly better than the other groups because:

- The statistical significance difference between the non-disabled group vs the B/VI group is ($p = 0.04, P < 0.5$) with the error occurrence means $M_{non} = 0.36$ and $M_{bvi} = 0.0$.
- The statistical significance difference between the motor-impaired group vs B/VI the group is ($p = 0.06, P \leq 0.5$) with the error occurrence means $M_m = 0.77$ and $M_{bvi} = 0.0$.

The variety of interactions allows each participant to select the best suitable interaction that overcomes the participant's difficulty and enables the participant to perform tasks effectively and precisely.

Similarly, the CoUIM approach surprisingly demonstrated a vast improvement in terms of ***user performance*** significant difference. Accordingly, the p-value was increased–compared to the user performance *p-value* from ($p = 0.007, P < 0.01$) to ($p = 0.05, 0.05 < P < 0.1$). The reasons that influenced the results were:

- The size of the sample was enlarged from $18 \rightarrow 29$ participants.
- The means of user performance were obviously decreased from the average of $193.5 \rightarrow 184.9$.

The t-Test results showed that the statistically significant difference occurred when we compared the non-disabled and motor-impaired groups so that ($p = 0.03$, $0.01 < P < 0.05$). The reason of such difference is that the participants with motor-impairment performed much slower compared to users with no disability, which is naturally expected as most of participants with motor-impairment used touchless motion based interaction. Allowing such crossover user interface showed that there was no statistically significant difference between group with no disability and the group with blindness and visual impairment, which was ($p = 0.37$, $P > 0.1$) with means of $M_{non} = 154$, $M_{vib} = 177$. This means that the group with blindness and visual impairment was similar in performance to the group with no disability. Likewise, there was a weak statistical difference between groupd with motor-impairment and the group with blindness and visual impairment, which was ($p = 0.18$, $P > 0.1$) with means of $M_m = 225$, $Mean_{bvi} = 177$. The similarity of the two groups' sizes and the reasonable difference of their user performance means decreased statistical significant differences. The variety of interactions in CoUIM allows every participant to select the best suitable interaction and increase efficiency. Our statistical and qualitative analysis shows that increasing interaction options for users would allow more accommodation, precision, efficient completion of tasks.

6 CONCLUSIONS AND FUTURE WORK

The Crossover User Interface (CoUIM) model makes the task of managing a Xen-cloud more inclusive so that persons with no disability, persons with blindness and visual impairment, and persons with motor disability can share the same user interface by utilizing a variety of interactions. We were able to deliver a very effective solution for a minimal cost ($200). Our results show that people with disability, especially those with blindness or visual impairment, can compete on equal footing with the participants with no disability. Specifically the research showed that user inclusiveness was improved by using our CoUIM that contain a variety of

Participant	*Disability*	*Easy*	*Engaging*	*Quick*	*Interesting*	*Precise*	*Useful*
A	Motor	SA	SA	SA	A	SA	SA
B	—	SA	N	SA	SA	A	A
C	Motor	SA	SA	D	D	SA	D
D	—	SD	SD	SD	SD	SD	SD
E	Visual	SA	A	A	A	A	A
F	Motor , Hearing	N	A	A	A	A	A
G	—	N	A	N	N	N	N
H	Visual	N	N	A	N	A	N
I	—	A	A	A	SA		SA
J	—	SA	SA	A	SA	A	SA
K	Motor	D	A	N	N	D	SD
L	Motor	SA	SA	A	SA	A	SA
M	—	SD	SD	SD	SD	SD	SD
N	—	SA	SA	SA	SA	SA	SA
O	—	SA	SA	SA	SA	A	SA
P	—	SA	SA	A	A	A	A
Q	Visual Motor	A	A	A	SA	A	SA
R	Motor , Visual, Brain injury	SD	D	D	SD	SD	SD
S	Motor	A	A	A	A	A	A
T	Motor ,Brain injury	SA	A	A	A	SA	A
U	Visual, Motor	SA	SA	SA	SA	A	SA
V	Visual	SA	SA	SA	SA	SA	SA
W	Visual	N	D			D	D
X	Visual Motor	SA		D	SA	SA	SA
Y	Visual	A	A	A	A	N	N
Z	Visual	SD	SD	SA	SD	SD	SD
AA	Motor	A	SA	N	N	D	A
AB	—	SA	SA	A	SA	SA	SA
AC	—	A	A	SA	N	SA	N

Table 3: The qualitative data of experiencing the crossover user interface.

Groups	*Count*	*Sum*	*Average*	*Variance*
B/VI	9	0	0	0
Motor	9	7	0.78	1.19
None	11	4	0.36	0.25

Table 4: Statistical summary of error occurrence using Anova single factor method.

Groups	*Count*	*Sum*	*Average*	*Variance*
B/VI	9	1590	176.67	4661.75
Motor	9	2022	224.67	6272.25
None	11	1689	153.55	1114.67

Table 5: Statistical summary of user performance using Anova single factor method.

interactions. The results also showed that user performance can be imrproved by using a more inclusive and better design. CoUIM provides improved user satisfaction as each user utilized the method that best fit their abilities.

Based on our experiences and observations, we would like to provide the following in future: (a) Our inclusive design framework could be extended to provide novel solutions towards other types of disabilities which we were not able to include in our study; (b) Complex disabilities or multiple disabilities (e.g. a person with blindness, hearing, speaking, and motor impairments) pose new forms of challenges and need to be considered in future; and (c) Disabilities often change dynamically over time, and the changes may require different or additional accommodations.

7 ACKNOWLEDGMENTS

Ms. Bonnie Snyder who provided inspiration for our work. We are thankful to our friends at the Independence Center in Colorado Springs. We are also grateful to all the participants who participated in our study. This paper is based on Dr. Abdullah Almurayh's Ph.D. Thesis, entitled Corssover User Interface for Inclusive Computing, Dissertation Advisor Dr. S.K. Semwal at the University of Colorado, Colorado Springs, pp. 1-341 (2014).

8 REFERENCES

[1] Adobe. Prevent security settings from interfering with screen readers.

[2] Abdullah Almurayh and Sudhanshu Semwal. Cultural considerations for designing crossover applications for the visually impaired. In *Information*

Reuse and Integration (IRI), 2013 IEEE 14th International Conference on, pages 668–675. IEEE, IEEE, Aug 2013.

[3] Pradipta Biswas, Peter Robinson, and Patrick Langdon. Designing inclusive interfaces through user modeling and simulation. *International Journal of Human-Computer Interaction*, 28(1):1–33, 2012.

[4] P. Blenkhorn and D.G. Evans. Using speech and touch to enable blind people to access schematic diagrams. *Journal of Network and Computer Applications*, 21:17–29, 1998.

[5] Michael W Boyce, Drea K Fekety, and Janan A Smither. Resource consumption and simulator driving performance using adaptive controls. *Assistive technology*, 25(3):158–165, 2013.

[6] Dirk Burkhardt, Kawa Nazemi, Nadeem Bhatti, and Christoph Hornung. Technology support for analyzing user interactions to create user-centered interactions. In *Universal Access in Human-Computer Interaction. Addressing Diversity*, volume 5614 of *Lecture Notes in Computer Science*, pages 3–12. Springer Berlin Heidelberg, 2009.

[7] Christian BÃ¼hler. Design for all-from idea to practise. In *Computers Helping People with Special Needs*, volume 5105 of *Lecture Notes in Computer Science*, pages 106–113. Springer Berlin Heidelberg, 2008.

[8] Donna M Cowan and Yasmin Khan. Assistive technology for children with complex disabilities. *Current Paediatrics*, 15(3):207–212, 2005.

[9] D.E. Individuals with disabilities education improvement act of 2004, 2004.

[10] D.O.L. What are "mitigating measures"?, 2008.

[11] DOL. Disability, health benefits advisor, 2014.

[12] Duxbury. Braille translation software from duxbury systems, 2012.

[13] Google. Accessibility: Assistive technology support, 2012.

[14] Zhang Hua and Wei Lieh Ng. Speech recognition interface design for in-vehicle system. In *Proceedings of the 2Nd International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, AutomotiveUI '10, pages 29–33, New York, NY, USA, 2010. ACM.

[15] Arthur M. Langer. *The User Interface*, pages 21–35. Springer London, 2008.

[16] Nicole Laporte. Hiring the blind, while making a green statement, 2012.

[17] Unseok Lee and Jiro Tanaka. Finger identification and hand gesture recognition techniques for natural user interface. In *Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction*, APCHI '13, pages 274–279, New York, NY, USA, 2013. ACM.

[18] W.C. Mann and J.P. Lane. *Assistive technology for persons with disabilities*. American Occupational Therapy Association, 1995.

[19] Suzanne Martin, Johan E Bengtsson, and Rose-Marie Dröes. Assistive technologies and issues relating to privacy, ethics and security. In *Supporting People with Dementia Using Pervasive Health Technologies*, pages 63–76. Springer, 2010.

[20] Joyojeet Pal, Manas Pradhan, Mihir Shah, and Rakesh Babu. Assistive technology for vision-impairments: Anagenda for the ictd community. In *Proceedings of the 20th International Conference Companion on World Wide Web*, pages 513–522, New York, NY, USA, 2011. ACM.

[21] Dana Petcu, Ciprian Craciun, Marian Neagul, Iñigo Lazcanotegui, and Massimiliano Rak. Building an interoperability api for sky computing. In *High Performance Computing and Simulation (HPCS), 2011 International Conference on*, pages 405–411. IEEE, 2011.

[22] Joanne Pompano. *Electronics in the 20th Century: Nature, Technology, People, Companies, and the Marketplace*, volume VII, chapter Designing Accessible Websites for Blind and Visually Impaired. Yale-New Haven Teachers Institute, 1999.

[23] Kristen Shinohara. A new approach for the design of assistive technologies: Design for social acceptance. *SIGACCESS Access. Comput.*, (102):45–48, January 2012.

[24] B. Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison Wesley, Menlo Park, CA, 3rd edition, 1998.

[25] TopTenREVIEWS. Jaws 12, 2012.

[26] Brian Wilke, Jonathan Metzgar, Keith Johnson, Sudhanshu Semwal, Bonnie Snyder, KaChun Yu, and Dan Neafus. Crossover applications. In *Proceedings of the 2009 IEEE Virtual Reality Conference*, pages 305–306, Washington, DC, USA, 2009. IEEE Computer Society.

OpenLensFlare: an Open-Source, Lens Flare Designing and Rendering Framework

István Csoba
University of Debrecen
Faculty of Informatics
Debrecen, Hungary
csistvan94@gmail.com

ABSTRACT

Lens flare rendering in computer games has always been lacking. While physically motivated solutions based on real camera systems exist, their inherent complexity makes them inappropriate for widespread use. Developers are not trained to work with these complicated imaging systems, and even if they were, artists and designers prefer intuitive parameters over these complicated optical system descriptions. More support is needed to start adopting to these advanced models, and to show that with an appropriate tool the situation can be vastly improved. This paper describes OpenLensFlare, an open-source C++ framework for rendering convincing physically-based lens flare effects. Additionally, a supporting optical system editor is also provided, which is capable of producing and visualizing optical systems and showing an example usage of the run-time library. Together, these two systems can be used to replace the existing naive algorithms, with small effort and low integration complexity.

Keywords

Lens Flare Rendering, Optical Systems, Lens Editor, Open-Source Software

1 INTRODUCTION

Rendering realistic and plausible images has been a hot topic of research ever since the birth of computer graphics. A subset that is of special interest, which has been gaining popularity in recent years, is proper simulation of the aberrations in a camera system. Working with real optical systems is a challenging task, since it is hard to see what happens to a ray once it enters the lens housing. Supporting tools are needed to aid working with these systems.

This work focuses on one particular aberration, which is often referred to as lens flare. Lens flare is a phenomenon in photography that is due to the incoming light taking undesired paths while passing through the lens system of the objective. The two most prominent effects that lead to the birth of these flares are the internal reflections between the lens elements which causes ghost patches to appear, and the diffraction of the incoming light that is most visible on the starburst patterns at the locations of bright light sources on the image.

Although lens flare is a very subtle effect, its presence is just as important, since it plays a very important role in keeping up the illusion of a non-simulated world. Rendering these effects in real-time, at interactive frame rates is a problem that has existing solutions, but they tend to be very complex and do not enjoy widespread adoption. Softening their limitations is another open problem today, and one that demands more research work to be done.

In this paper an OpenGL-based C++ lens flare rendering library is introduced, named OpenLensFlare. Designed to be modular and extensible, it can serve as both a starting point into learning more about these recent advancements, but may also be used as the rendering solution of real-time applications. The library provides all the components needed to render plausible lens flare effects, while at the same time also granting the required customization and freedom.

A multipurpose supporting tool is also described, capable of not only managing the accompanying data of the run-time library, but also visualizing the various properties of the lens systems and their generated lens flare. The editor also serves as an integration example and can be used to further simplify the process of using lens flare for enhancing the visual quality of our renderings.

The main goal of this work is to help with the adoption of advanced lens flare rendering methods, while at the same time also providing an environment that is useful for designing and testing future, improved versions of the algorithms in question.

The paper is structured as follows: Section 2 gives an overview of similar tools and summarizes the results of the research done in the field of lens flare rendering. In Section 3 the rendering algorithms implemented in the framework are described in detail. Section 4 outlines the main features of the proposed framework. Section 5 describes the design choices of the library and explains some of the potential issues of implementing such a library. Section 6 discusses the various validations and testings performed on the implementations. Lastly, Section 7 sums up this work and talks about possible future improvements for the library.

2 RELATED WORK

2.1 Special Effect Frameworks

Code reusability has always been one of the core principles in computer programming. The advantages of using a well tested system are nearly endless. Real-time rendering is an area where this is especially true, thanks to the poor debugging support and vast amount of graphics APIs that should be supported. This section outlines a few existing frameworks, each similar in spirit to the proposed solutions.

The GameWorks [1] initiative by NVIDIA is one of the largest library collections, aimed at providing game developers with the best material needed for creating their products. The development kit has a whole section devoted to rendering, named VisualFX, covering a wide range of effect rendering needs for most games. These implementations have been used in many games throughout the years, with great success and positive feedback from the developers [6].

What the GameWorks libraries lack though is proper tooling support. The components themselves are well documented, but managing their settings can be unintuitive at times, making it hard to use them correctly. More focused frameworks come with easy-to-use editors, debuggers and visualizers, allowing for a better user experience. Examples of these systems include PopcornFX [11] and the Houdini tools [8]. Although the complexity of these specialized frameworks is much higher, developers still have an easier time creating the desired effects, all thanks to the superior editing capabilities of their accompanying toolsets.

2.2 Other Related Software

Tooling support for working with real camera systems is lacking. The only existing software that is relevant to this field is OpticStudio [16], which is a professional optical system editor for optical engineers, developed by Zemax. Despite its good camera visualization capabilities, usage of an external tool that is so loosely coupled with our system is also a tedious process. Furthermore, since OpticStudio was designed with a different set of features in mind, it lacks any rendering support, making it an inappropriate choice for simulating lens flare.

2.3 Lens Flare Algorithms

Lens flare can be divided into two main components: a starburst pattern, that is due to the diffraction that occurs when the light passes through the aperture of the imaging system, and a varying number of ghosts, each of which corresponds to a unique sequence of inter-reflections between the optical elements. This section gives a broad overview of the most recent advancements in rendering these effects properly.

Texture Composition The most basic approach to the problem is additive composition with pre-rendered ghost and starburst textures. The starburst texture can be positioned at the center of the light source on the screen, while the ghosts are usually placed on a line that passes through it. Although not physically motivated, the algorithm is still often favored, all thanks to its low rendering and integration costs.

Starburst Texture The look of these starburst textures can be improved greatly by taking into consideration the diffraction that occurs in the optical system. Algorithms describing how to do this for the human eye exist [12], and an analogue technique can be implemented for camera systems as well. In fact, the most advanced starburst rendering method invented by Hullin et al. [4] is still based on texture rendering, but instead of artificially generating something that looks similar to a real-world effect, they attempt to mimic the looks of the phenomena by manual texture composition, as dictated by the underlying diffraction theorem.

Ghost Patches A more sophisticated ghost rendering solution can be given by involving a real camera system. Efficiently rendering with such a system has been a topic of hot research lately [5, 2, 13, 17], and has been the main idea of the work done by Hullin et al. [4] and Lee et al. [10] in the field of lens flare ghost image rendering. The premise of their invented algorithms is that today's graphics units are very well suited for tracing a vast amount of rays in parallel, under heavy time constraints. Therefore, images that closely match these ghost effects can be rendered by constructing an environment appropriate for imitating a real imaging system.

3 MATHEMATICAL BACKGROUND

In the last section we have already seen the core ideas behind the related lens flare rendering algorithms. This section gives the reader the required definitions and principles to understand design choices and inner workings of the proposed framework.

3.1 Rendering Starburst Patterns

One of the two parts that make up the lens flare effect is the starburst pattern. The main source for this interesting flare shape is from the diffraction that occurs when the incoming light passes through the iris aperture in the optical system, causing the light to bleed into otherwise shadowed areas.

To render a plausible starburst effect, a pregenerated texture is drawn at the location of each bright light source on the image, using the f-number of the camera to control the size and intensity. As shown by Ritschel et al. [12] and Hullin et al. [4], this texture can be generated by using the far field Fraunhofer diffraction formula [3]. For this first a mask image $T(x,y)$ needs to be provided, which represents the iris shape. From this mask the corresponding Fourier power spectrum $F(u,v)$ is computed by taking its squared Fourier transform. The starburst texture $S(x,y)$ is then generated by iterating through the visible color spectrum and blending together the corresponding scaled copies of the power spectrum as:

$$S(x,y) = \sum_{i=0}^{n-1} \lambda_i F(u_i, v_i)$$
$$\lambda_i = \lambda_s + i\frac{\lambda_e - \lambda_s}{n}$$
$$(u_i, v_i) = (u,v)\frac{\lambda_s}{\lambda_i}$$

where λ_s, λ_e and n are user defined parameters for the starting, ending wavelengths (in nanometers) and the number of wavelength steps, respectively. Note that the way (u_i, v_i) is computed slightly derives from what Ritschel et al. suggested, but that is because they used a mixture of Fraunhofer and Fresnel diffraction, however as they have also shown, the quality gains are marginal and thus very similar outputs can be rendered by only relying on the Fraunhofer formula, as presented above.

3.2 Rendering Ghosts

To render convincing ghost flares, a set of rays have to be traced through the optical system. As the time needed to perform this would be immensely long, some simplifications and optimizations are needed to make it a suitable method for real-time applications. Hullin et al. showed that this is indeed possible to do [4] with some carefully made considerations. Their method relies on the fact that rays originating from the front element reach the sensor in a continuous manner, so instead of tracing a dense set of rays through the system, a sparse set of rays suffices just as well, with some interpolations.

The algorithm starts by enumerating the various light paths that each correspond to a single ghost patch on the image. Knowing that the ray is propagating forward, this can be done by listing the optical system indices from which the light is reflected, and assuming that transmission on every other surface touched by the ray. With the unique paths computed, the ghosts are rendered independently, one after the other.

To perform the ray-tracing, the lenses are virtually treated as analytic surfaces (spheres and planes) in 3D. The interfaces are then iterated over one by one, taking the intersection of the traced ray and the optical element, computing the normal and angle of incidence, and continuing with the reflected and refracted ray, as needed. Once all the rays are traced through the system and their positions on the sensor is computed, the sparse grid is triangulated and filled by interpolating the various attributes that were generated while following the ray, such as the ratio of reflected energy.

To get the intensity of the ghost, the Fresnel reflectivity equation for unpolarized light can be used [3]:

$$R = \frac{1}{2}\left(\frac{n_1 cos\theta_1 - n_2 cos\theta_2}{n_1 cos\theta_1 + n_2 cos\theta_2}\right)^2 + \frac{1}{2}\left(\frac{n_1 cos\theta_2 - n_2 cos\theta_1}{n_1 cos\theta_2 + n_2 cos\theta_1}\right)^2$$
$$T = 1 - R$$

where n_1 and n_2 corresponds to the refractive indices of the two participating media, and θ_1 and θ_2 are the angles of light in the two media relative to the normal. As mentioned above, this equation is evaluated per ray at each interface where the light undergoes reflection, and multiplied together to compute the ratio of the reflected light amount.

To consider dispersion, a wavelength-dependent refractive index has to be computed and used during the ray-tracing process. The lens patents typically only contain the index of refraction at the Fraunhofer D-line (n_d) and the corresponding Abbe-number (V_d). Considering the definition of the Abbe-number:

$$V_d = \frac{n_d - 1}{n_F - n_C}$$

and knowing that λ_D, λ_F, and λ_C are fixed, substituting these into the two-term form of Cauchy's equation we get the following:

$$n(\lambda) = 1.50459 + \frac{\lambda}{0.004215}$$

which can be used to provide the wavelength-dependent refraction indices (note that the wavelengths are measured in micrometers). Using these indices, the final ghost images are rendered by performing the ray-tracing process for multiple wavelengths, as outlined above, and blending the results together.

4 PROPOSED SOLUTION

The main issue of integrating a physically-based lens flare rendering algorithm into our application is the requirement of a proper optical system prescription. Such prescriptions are hard to both acquire and edit, which is why even the largest companies have stayed away from adopting them. Artists and game designers cannot be expected to work with these complex systems, and so it is a reasonable move on the developers' side. But the situation is not hopeless, as with proper support even the largest systems become manageable.

This work focuses on a framework that is capable of rendering proper lens flare and managing all the data needed to achieve that goal. The framework is made up of two components: a run-time library implementing the rendering algorithms and holding the corresponding data, and an editor tool to design custom optical systems and lens flare effects. An example image of this editor in action can be seen on Figure 1. This section explains the various parts and features of the proposed framework and gives an overview of how it attempts to ease the process of using these large optical systems for creating custom lens flare.

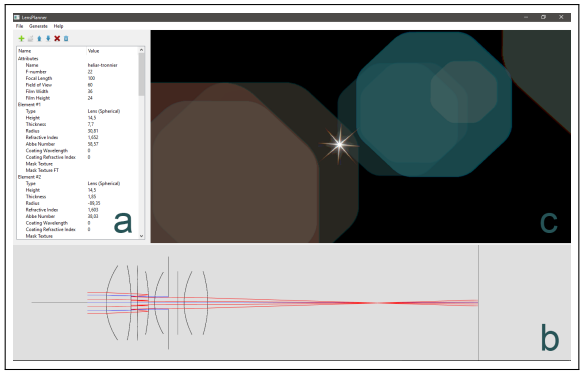


Figure 1: *Interface of the editor component. (a) optical system editor; (b) optical system visualizer; (c) preview lens flare rendering.*

4.1 Optical System Design

To render physically-based lens flare effects, first the description of a real imaging system has to be supplied. This description should include the various parameters of each of the lens interfaces present in the system, such as the material properties, physical height and radius of curvature of the interface, but it should also contain attributes global to the entire system, such as the focal length of the system, or the dimensions of its sensor.

These descriptions may be acquired from patent databases or special books [9, 14], however, designing a custom lens system is often desired, to improve the looks of the generated imagery. Since tweaking values in a text file or directly in the source code to achieve this would be a painful process, the proposed framework contains a component designated solely for editing these systems and exploring the characteristics of their generated flare effects by visualizing the optical system, the flare and various other aspects that contribute to the birth of the phenomenon. Figure 1 shows what this component looks like in action, using a real camera system to show the proposed editor.

To manage the aforementioned parameters, the run-time library contains a special object designed to hold the various optical system attributes. The lens editor component exposes these to the user in a tree structure, which contains both the global imaging parameters and all the lenses with their accompanying descriptions. Figure 2 shows a closeup image of the tree widget as it looks in the editor. The user is then able to assign custom values to these parameters, and immediately see the effects of the changes on the visualization widgets. The process of creating an optical system that has the desired imaging properties is vastly simplified this way. Once finished, the optical system can be exported and later reloaded by both the designer component and the rendering library.

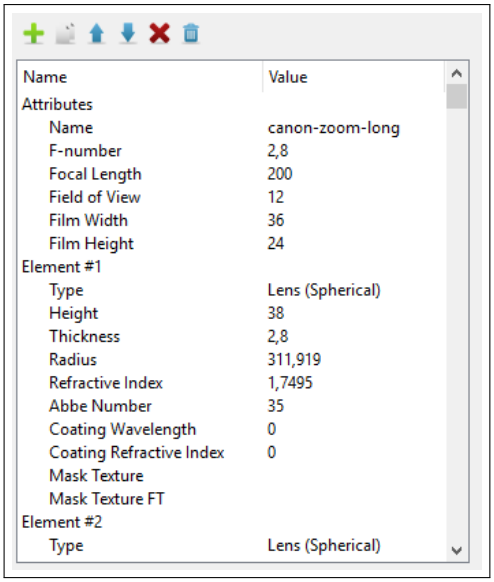


Figure 2: *The optical system editor widget of the lens designer tool. This is used to create and edit the lens system used for rendering lens flares.*

4.2 Data Compilation

To achieve acceptable rendering performance, some preliminary computations need to be performed. For most algorithms, these do not take a significant amount of time and can be carried out in the initialization step of the application. However, computing everything that is needed for rendering lens flare can easily take as long as an hour, even for a camera of average complexity.

Preparing all this data with an external tool is thus desired, so that they can be just loaded in at run-time, when they are needed.

OpenLensFlare was designed with pre-computation in mind. All the algorithm implementations support extracting the data they have computed and loading them back later. Thus, a full optical system coupled with ghost and starburst rendering parameters can be prepared and loaded in a matter of milliseconds. Everything is provided in native C++ and OpenGL objects, meaning the user is not locked to a single serialization strategy, but may choose the used methods as they will.

The editor contains a reference implementation for saving and loading the desired objects. The provided solution can be used to serialize the whole optical system and the rendering algorithm states - along with all of their generated data - into a set of XML and image files. This way the lens attributes can be calibrated once, and either used by us, in our product, or shared with others. Listing 1 shows the structure of one of these XML files - the one describing the optical system. Using the code provided with the framework, the user has out of the box support for generating run-time data and injecting it into their software.

```
<?xml version="1.0" encoding="UTF-8"?>
<opticalSystem>
  <name>ExampleOpticalSystem</name>
  <fnumber>11</fnumber>
  <!-- Other camera attributes... -->
  <elements>
    <element>
      <type>lensSpherical</type>
      <height>24.0</height>
      <thickness>9.6</thickness>
      <radius>65.220001</radius>
      <!-- Other lens attributes... -->
    </element>
    <!-- More elements...-->
  </elements>
</opticalSystem>
```

Listing 1: *Structure of an XML file that describes an optical system, as generated and understood by the provided reference serialization implementation.*

4.3 Rendering

Understanding an optical system is only half of the problem. We still need to find a way to simulate how the ghosts would be formulated on the captured image, and render one that looks similar to that of the original. As it turns out the problem is so complex that existing algorithms are very hard to grasp and providing a fully functional implementation is troublesome.

The proposed framework tries to ease the process of generating eye-catching lens flare imagery by giving the developers the necessary tools to render ghosts and starburst images. All the rendering algorithms reside in the lower, run-time layer. Currently the framework implements the algorithms presented by Hullin et al. [4], as described earlier in Section 2. The user is free to select, configure and combine these algorithms in any way that fits their requirements, allowing to adopt to the running environment.

The rendering aims to be fully customizable. It is possible to specify intensity scaling factors, list of ghosts to render, ray grid resolutions, anti-reflection coating parameters, ray culling factors and iris shape smoothing. Furthermore, the library expects the rendering context to be pre-configured, thus enabling the use of custom render targets and blend modes.

Large optical systems can imply high rendering times, which might be unacceptable for certain applications. The library has options for resolving this issue. Acceleration can be achieved by using the output of the precomputation process discussed in Section 4.2. The generated ghost data can be edited to reduce the number of rays used for rendering, decreasing the overall pressure on the graphical unit. Additionally, intensity and bounding information can be used to cull ghosts with a low impact on the render, saving additional time for other tasks.

4.4 Visualization

In computer science, visualization of the data is a key step of understanding a problem. Data that might take the developer days to fully learn, can easily be perceived in a matter of hours with the use of images. That being said, data visualization has been one of the most important motivating factors for making this framework.

As it has been outlined above, lens prescriptions are hard to understand. For the untrained eye, their parameters are totally meaningless and difficult to fully grasp. To help us better understand these raw camera parameters, the editor provides a view of the lenses and their relations to one another. First the optical axis is rendered in the center of the viewport. The elements are then drawn on top of it one by one, as either a simple section (flat surfaces and sensor), an arc (spherical surfaces) or a section with a hole in the middle (iris aperture). The elements themselves are not connected together, since the optical system descriptions do not contain the information needed to reassemble the physical lenses from the provided surface parameters.

The usefulness of this optical system sketch is further improved by showing how a set of example rays travel through the system, with or without involving interreflection paths between the lens elements. Each ray is traced through the whole optical system independently, computing and storing their intersections with each interface and updating the ray directions by following the

reflected or refracted ray, as needed. The intersection points that correspond to the same ray are then connected, forming a set of lines that either reach the sensor or show the points where their rays are fully blocked or miss an interface. This kind of exploration of the optical systems is a common practice among optical engineers as well, since it allows to better examine the performance and imaging characteristics of the system. Figure 3 shows an example output generated with the proposed lens editor, visualizing rays that correspond to normal, and reflected ray paths, respectively.

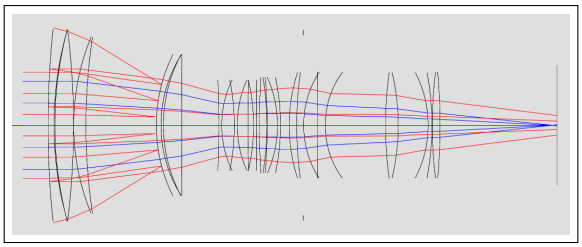


Figure 3: *Example visualization of rays taking the normal path, converging in a single point on the sensor (blue) and rays corresponding to ghost, spreading out on the camera sensor (red).*

Another important aspect of data visualization is previewing the output of rendering lens flare with our optical system. Immediate feedback of the changes done to the lenses is very important, as the whole purpose of the work done on designing these lens systems is to use them for lens flare rendering. To provide the user this wisdom, the editor component has an embedded flare previewer widget that is updated on the fly as the optical system is being modified, to display what the flares generated by the current configuration would look like. This preview rendering is performed by the run-time library component, to make sure the looks of the sample render fully matches the final image in the embedding application.

Finally, there are some other characteristics of both the camera system and the rendering algorithms that are worth exploring. Examples of these include the bounds on the front element that correspond to rays reaching the sensor, or the UV coordinates of the rays when passing through the aperture iris. An example output of such a simulation can be seen on Figure 4, which visualizes the aforementioned front element bounds. While these may not always have a visual impact on the rendering, they usually affect the average render times of the system. The run-time library can render this data directly on its own, by only setting a few flags on the rendering objects. This way these visualizations can not only be shown in the editor, but are also available in the embedding product. Such hidden information not only allows the designer improve the run-time performance of the rendering, but also helps developers to perform debugging tasks and better understand how a specific ghost image is rendered.

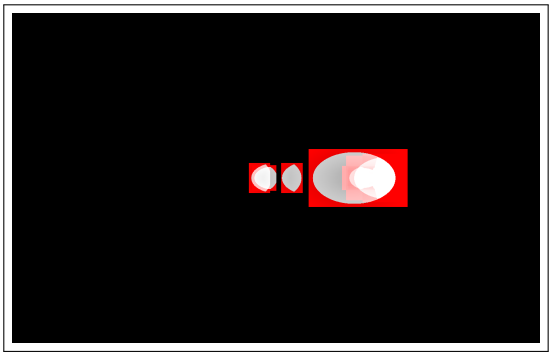


Figure 4: *Visualization of rectangles on the front element bounding the rays that can reach the sensor. The black section corresponds to the front element surface, on which each colored rectangle denotes a single ghost (note that they may overlap). It is clearly apparent that limiting the ray-tracing to these areas can vastly improve the quality and performance of the ghost renderings.*

5 IMPLEMENTATION

The main goals of the framework have been discussed in the previous section. The following subsections provide some insight into the design process that the library went through and some of the issues with alternative solutions that have been tested.

5.1 API Design

The library was designed with extensibility in mind and aims to provide an environment suitable for implementing any physically-based lens flare algorithm. The fundamental terms are all fully represented with separate classes describing all of their relevant attributes. Examples of these core abstraction classes include optical systems with their elements, ghosts, ghost enumerators and abstract light sources.

The rendering implementations are all built on top of these foundation classes. They fall into two main categories, which correspond to the division of the phenomena described in Section 2.3. The user is free to choose any number of these as they will. It is possible to distribute the rendering of the same set of ghosts between multiple instances, by using a heuristic, for example.

The library also provides a tight abstract interface for the algorithms. These can be used where starburst or ghost rendering is required, without having to refer to concrete algorithm classes after initialization. This feature proved useful when implementing the visualizer components, and should also simplify the customization process for end-users as well. An usage example can be seen on Listing 2.

```
// Lens system for rendering, assumed
// to be loaded by the application
OpticalSystem* pSystem =
    loadOpticalSystem();
// Initialize a diffraction starburst
// algorithm, with a size of 0.1
// and intensity of 1.0
DiffractionStarburstAlgorithm* sb = new
    DiffractionStarburstAlgorithm(pSystem
    , 0.1f, 1.0f);
// Create a 1024x1024 starburst texture
// that merges scaled copies from 380
// to 780 nm, using a step size of 5 nm
sb->generateTexture(1024, 1024, 380.0f,
    780.0f, 5.0f);
// Create a ray traced ghost renderer
// that renders all ghosts and traces
// the parameter wavelengths
RayTraceGhostAlgorithm* ghost = new
    RayTraceGhostAlgorithm(pSystem,
    GhostList(pSystem), { 650.0f, 510.0f,
     475.0f }, 1.0f);
// Compute optimal rendering parameters
ghost->computeGhostAttributes();
```

Listing 2: *An example C++ code, showcasing the initialization process of the rendering objects.*

5.2 Graphics API

The choice of graphics API is mostly irrelevant for a tool, but the supported APIs play a crucial role when selecting a rendering library. Since OpenGL is so widely supported, it seemed natural to use it for both the flare algorithms and the editor interface. All objects – including the algorithms and the optical system descriptions – expect OpenGL handles when working with objects in their interfaces. The rendering implementations also make heavy use of OpenGL and require a valid context to be active when calling any of their rendering functions.

An alternative solution is the usage of an API abstraction layer. While it has the benefit that in theory it could work with every API in existence, it is not true in practice. There are holes in the feature matrix, and certain object types are not available everywhere. Another possibility is a third party library, of which the most commonly chosen one is bgfx [7]. While it could solve some the aforementioned issues, it has another set of deficiencies, such as the lacking capability of generating shaders at run-time, which is required to function correctly by some of the planned algorithms.

5.3 Shaders

Almost every external resource is configurable in the interface, but the use of shaders needed special attention. Since a large section of the rendering logic resides in shaders, they are long enough to require separation into their own files. Accessing them is not as trivial as it may first seem, since their presence should ideally be hidden in the interface. Additionally, since a rendering library should function properly with limited file system privileges, direct file system access is also ruled out, further complicating the situation.

A different approach has been taken to solve the problem. Instead of accessing files at run-time, the shader sources are compiled into the binary files upon compilation, as raw string literals. When using these shaders, the implementations can directly refer to their source codes in a special namespace, with the same name as they were on the disk before compilation. This way the application can run under limited file system access rights and the library remains functional, saving the user from the trouble of distributing and managing external file dependencies.

5.4 Rendering

Rendering of the starbursts and ghosts is done on-demand, when initiated by the user. The embedding application is expected to configure the rendering objects, as described in 5.1, and call their appropriate methods to perform the rendering. These objects rely on embedded shaders as described in the previous section, and assembles them under the hood into the required OpenGL shader programs.

Before rendering with the optical system, the lens parameters are locally transformed into an optimal format that is faster to access during rendering. These values are then uploaded to the GPU into a set of uniform arrays, consumed by the corresponding shader programs. The rendering is then carried out by issuing the appropriate draw calls for the selected ghosts and starbursts. The ray-tracing needed for ghost rendering is performed in the vertex shader and finalized in a geomery shader, outputting triangles to the hardware rasterizer for automatic interpolation.

5.5 Lens Editor

The lens editor tool is based on Qt 5 [15]. An alternative and natural solution would be using an immediate mode GUI library, but it proved to be not suitable for the task. The pre-computations discussed in Section 4.2 allow for important optimizations, but do not work very well with rapid changes to the optical system. To display a meaningful image takes a significant amount of time without examining the data, which needs to be repeated every frame with immediate mode interfaces. The interface thus becomes unusable, creating a horrible user experience.

6 RESULTS

Validation A test scene has been created to verify the library's applicability and its appropriate performance.

Many different camera systems and render parameters were experimented with, to check that the library is capable of working with imaging systems of varying complexity. Figure 5 shows an example output of a starburst and many ghosts rendered with the proposed library and composited onto the test scene, all in real-time, allowing for user interaction.

Figure 5: *Lens flare generated by a complex Nikon lens system, rendered with OpenLensFlare and composited onto the test scene.*

The rendering algorithms are all based on already existing methods, well tested by their own authors. Thus, only the correctness of the implementation had to be verified. This was done by reproducing the environments used to render the sample images provided in the original papers, and manually comparing my results to the reference renderings. The generated starburst texture, also the shape and intensity of the rendered ghosts seemed to closely match the reference images, proving that the implementations are correct. The only difference is in the colouring of the ghosts, which is due to the anti-reflective coating parameters used by the authors that cannot easily be reverse engineered. However, OpenLensFlare gives the user the option to assign these values to each lens, making it possible to produce images that also match the colors of the reference ghost rendering, and also create custom setups as desired.

Performance Performance of the library was tested with the lens system mentioned above. Table 1 shows the average render times for a Heliar (US2645156, 8 reflective interfaces, 13 ghosts rendered) and a Nikon (JPS53131852A, 21 reflective interfaces, 142 ghosts rendered) lens system (both if which can be found in Smith's book [14]), with and without involving a precomputation step. The tests concluded that both the algorithms and the implementations are suitable for real-time applications, and even a complicated optical system with many lenses can be simulated at interactive frame rates.

Finally, performance of the precomputation process has been also measured. Usage of pregenerated rendering parameters - such as optimal number of rays or ray bounding on the front element - is crucial for achieving both fast render times and convincing renderings. OpenLensFlare provides a custom algorithm for approximating them in real time, and also implements a slower, full-blown ray-traced parametrization process, which - once editing the imaging system is finished - can be used to compute the final values, and export them as outlined in Section 4.2. The parameters used during this process can also be customized, which directly affects not only the quality of the generated data, but more importantly, the run time of precomputation step. Table 2 summarizes the running times of both the approximate and precise methods with the lens systems used in the previous tests, with varying computation parameters. As it can be seen, a detailed ray-tracing solution is inappropriate for real-time lens modifications, however it is perfectly capable of preparing the final rendering parameters in a short amount of time, while the approximated values can be used throughout the process of editing the optical system.

	No Precomputation	Precomputation
Heliar	70,0 ms	4,8 ms
Nikon	458,4 ms	215,3 ms

Table 1: *Performance comparison of ghost rendering with a Heliar and a Nikon lens system, with and without pre-computated ghost bounding information.*

	Approximate	2 passes	3 passes
Heliar	0,006 s	8 s	9 s
Nikon	0,011 s	338 s	365 s

Table 2: *Run time comparison of the approximate and precise pre-computation processes, computing parameters for 181 different incident angles in total. The number of refinement passes denotes how many iterations were taken per angle per ghost when bounding the ghosts on the front element.*

Limitations As for limitations, the algorithms are currently implemented with the assumption that the optical system contains exactly one iris, which also has an accompanying mask texture. Additionally, while the shaders are well generalized for any number of reflections, currently they only consider the first two reflections of any ghost path. These limitations are easy to come around and are expected to be removed in the future, but to ease the implementation and management process, the decision was made to use these simplifications in the initial implementation, based to the observations of the authors of the rendering algorithms.

Furthermore, to keep the rendering times low, the algorithms currently treat each interface of the optical system as spherical. Also, only objectives with a field of view smaller than 90° were tested, since such a field

of view is used most commonly in computer games. While in theory the algorithms could handle all types of optical systems, the ray-tracing is performed with the assumption that rays reach each interface exactly once, in the order that they are given, which is often not true for other types of lens systems (e.g. fisheye lenses).

7 CONCLUSION

Lens flare rendering is often overlooked in rendering applications of the present. Developers tend to favor basic algorithms, due to their implementation and maintenance simplicity. However, the more advanced algorithms can be made to work in these time-constrained environments, and the efforts are well worth it. This paper described an open-source, physically-based lens flare rendering library and an accompanying editor capable of providing all the data needed for the runtime to function correctly. These together can be used to render physically-based lens flare in games and similar applications, but may also be useful for optical experts to explore the lens flares generated by their lens system designs. Hopefully, these efforts will make developers realize the potential lying in these algorithms and I am looking forward to seeing these effects incorporated into future applications.

As for areas for future work, the user interface of the editor tool needs more customization options for its visualization components. Namely, it is currently impossible to select in the software which ghosts should be previewed, and the looks of the sketch also cannot be altered. The implementation is capable of doing all of these tasks, and thus only the corresponding widgets need to be added, after a slight interface redesign.

Additionally, to make this framework a much more generic solution, implementations need to be added for other graphics APIs too. Furthermore, creating bindings to popular game engines could improve the library's accessibility. Lastly, implementing custom algorithms based on the solutions mentioned in Section 2.3 could further improve the visual quality of the renderings and give the user more options to choose from.

8 SUPPLEMENTARY MATERIAL

The source code of both the run-time library and the lens editor components is available for download, at https://github.com/luorax/OpenLensFlare.

9 REFERENCES

[1] COOMBES, DAVID. Cutting Edge Tools and Techniques for Real-Time Rendering with NVIDIA GameWorks. ACM SIGGRAPH, 2016.

[2] HANIKA, J., AND DACHSBACHER, C. Efficient monte carlo rendering with realistic lenses. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 323–332.

[3] HECHT, E. *Optics*, 4 ed. Addison-Wesley, 2002.

[4] HULLIN, M., EISEMANN, E., SEIDEL, H.-P., AND LEE, S. Physically-based real-time lens flare rendering. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, pp. 108:1–108:9.

[5] HULLIN, M. B., HANIKA, J., AND HEIDRICH, W. Polynomial optics: A construction kit for efficient ray-tracing of lens systems. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 1375–1383.

[6] JOHNSTON, D., MAHER, M., AND TOROK, B. The Witcher 3: Enabling Next-Gen Effects through NVIDIA GameWorks. Game Developers Conference, 2014.

[7] KARADZIC, B. bgfx [Programming library]. `https://bkaradzic.github.io/bgfx/`, 2017. accessed 13-03-2017.

[8] KEATING, SCOTT. Houdini 16 for Games. Game Developers Conference, 2017.

[9] LAIKIN, M. *Lens Design*, 2 ed. Marcel Dekker Inc, 1995.

[10] LEE, S., AND EISEMANN, E. Practical real-time lens-flare rendering. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 1–6.

[11] PERSISTANT STUDIOS. PopcornFX [Computer software]. `http://www.popcornfx.com/`, 2017. accessed 13-03-2017.

[12] RITSCHEL, T., IHRKE, M., FRISVAD, J. R., COPPENS, J., MYSZKOWSKI, K., AND SEIDEL, H.-P. Temporal glare: Real-time dynamic simulation of the scattering in the human eye. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 183–192.

[13] SCHRADE, E., HANIKA, J., AND DACHSBACHER, C. Sparse high-degree polynomials for wide-angle lenses. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 89–97.

[14] SMITH, W. *Modern Lens Design*, 2 ed. McGraw Hill Professional, 2004.

[15] THE QT COMPANY. Qt 5 [Programming library]. `https://www.qt.io/`, 2017. accessed 13-03-2017.

[16] ZEMAX LLC. OpticStudio [Computer software]. `http://www.zemax.com/os/opticstudio`, 2017. accessed 13-03-2017.

[17] ZHENG, Q., AND ZHENG, C. Neurolens: Data-driven camera lens simulation using neural networks. In *Computer Graphics Forum* (2017), Wiley Online Library. DOI: `10.1111/cgf.13087`.