The 13-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2005

in co-operation with

EUROGRAPHICS

W S C G ' 2005

Short Papers

University of West Bohemia
Plzen
Czech Republic

Honourary Chair

M.L.V.Pitteway, Brunel University, Uxbridge, United Kingdom

Co-Chairs

Tosiyasu L. Kunii: Kanazawa Institute of Technology, Tokyo, Japan
Vaclav Skala, Univ. of West Bohemia, Plzen, Czech Republic

Edited by

Vaclav Skala

WSCG'2005 Short Papers Proceedings

Editor-in-Chief: Vaclav Skala
University of West Bohemia, Univerzitni 8, Box 314
306 14 Plzen
Czech Republic
skala@kiv.zcu.cz
Managing Editor: Vaclav Skala

Author Service Department & Distribution:
Vaclav Skala - UNION Agency
Na Mazinách 9
322 00 Plzen
Czech Republic

Printed at the University of West Bohemia

Hardcopy: *ISBN 80-903100-9-5*

WSCG 2005

International Programme Committee

Alexa, Marc (Germany)
Bajaj, Chandrajit (United States)
Bartz, Dirk (Germany)
Bekaert, Philippe (Belgium)
Benes, Bedrich (Mexico)
Bengtsson, Ewert (Sweden)
Bouatouch, Kadi (France)
Brodlie, Ken (United Kingdom)
Brunet, Pere (Spain)
Brunnet, Guido (Germany)
Clapworthy, Gordon (United Kingdom)
Coquillart, Sabine (France)
Debelov, Victor (Russia)
Deussen, Oliver (Germany)
du Buf, Hans (Portugal)
Ertl, Thomas (Germany)
Ferguson, Stuart (United Kingdom)
Floriani, Leila De (Italy)
Flusser, Jan (Czech Republic)
Goebel, Martin (Germany)
Haber, Jörg (Germany)
Harris, Mark (United Kingdom)
Hauser, Helwig (Austria)
Hege, Hans-Christian (Germany)
Chen, Min (United Kingdom)
Chrysanthou, Yiorgos (Cyprus)
Jansen, Frederik,W. (The Netherlands)
Jorge, Joaquim (Portugal)
Kakadiaris, Ioannis (United States)
Kalra, Prem (India)
Kjelldahl, Lars (Sweden)
Klein, Reinhard (Germany)
Klosowski, James T. (United States)
Kobbelt, Leif (Germany)
Kruijff, Ernst (Germany)
Magnor, Marcus (Germany)
Margala, Martin (United States)
Moccozet, Laurent (Switzerland)
Mudur, Sudhir,P. (Canada)
Mueller, Klaus (United States)
Muller, Heinrich (Germany)
Myszkowski, Karol (Germany)
O'Sullivan, Carol (Ireland)
Pasko, Alexander (Japan)
Peroche, Bernard (France)
Post, Frits H. (Netherlands)
Puech, Claude (France)
Puppo, Enrico (Italy)
Purgathofer, Werner (Austria)
Rauterberg, Matthias (Netherlands)
Rheingans, Penny (United States)
Rokita, Przemyslaw (Poland)
Rossignac, Jarek (United States)
Rudomin, Isaac (Mexico)
Sbert, Mateu (Spain)
Shamir, Ariel (Israel)
Schaller, Nan,C. (United States)
Schneider, Bengt-Olaf (United States)
Schumann, Heidrun (Germany)
Skala, Vaclav (Czech Republic)
Slusallek, Philipp (Germany)
Sochor, Jiri (Czech Republic)
Stuerzlinger, Wolfgang (Canada)
Sumanta, Pattanaik (United States)
Szirmay-Kalos, Laszlo (Hungary)
Taubin, Gabriel (United States)
Teschner, Matthias (Switzerland)
Theoharis, Theoharis (Greece)
Trahanias, Panos (Greece)
Velho, Luiz (Brazil)
Veltkamp, Remco (Netherlands)
Weiskopf, Daniel (Germany)
Westermann, Ruediger (Germany)
Wuethrich, Charles Albert (Germany)
Zara, Jiri (Czech Republic)
Zemcik, Pavel (Czech Republic)

WSCG 2005 Board of Reviewers

Adzhiev,V. (United Kingdom)
Alexa,M. (Germany)
Ammann,C. (Switzerland)
Anan,H. (United States)
Andreadis,I. (Greece)
Artusi,A. (Italy)
Aspragathos,N. (Greece)
Aveneau,L. (France)
Bajaj,C. (United States)
Bartz,D. (Germany)
Bekaert,P. (Belgium)
Benes,B. (Mexico)
Bengtsson,E. (Sweden)
Bieri,H. (Switzerland)
Bilbao,J. (Spain)
Bischoff,S. (Germany)
Bottino,A. (Italy)
Bouatouch,K. (France)
Bourdin,J. (France)
Brodlie,K. (United Kingdom)
Brunet,P. (Spain)
Brunnet,G. (Germany)
Buehler,K. (Austria)
Callieri,M. (Italy)
Clapworthy,G. (United Kingdom)
Coleman,S. (United Kingdom)
Coombe,G. (USA)
Coquillart,S. (France)
Daniel,M. (France)
de Aquiar,E. (Germany)
De Decker,B. (Belgium)
de Geus,K. (Brazil)
Debelov,V. (Russia)
del Rio,A. (Germany)
Deussen,O. (Germany)
Diehl,S. (Germany)
Dingliana,J. (Ireland)
Dmitriev,K. (Germany)
Doleisch,H. (Austria)
Dong,F. (United Kingdom)
Drakopoulos,V. (Greece)
du Buf,H. (Portugal)
Duce,D. (United Kingdom)
Durupina,F. (Turkey)
Egges,A. (Switzerland)
Eibl,M. (Germany)
Erbacher,R. (United States)
Ertl,T. (Germany)
FariaLopes,P. (Portugal)
Faudot,D. (France)
Feito,F. (Spain)
Ferguson,S. (United Kingdom)
Fernandes,A. (Portugal)
Fischer,J. (Germany)
Flaquer,J. (Spain)
Floriani,L. (Italy)
Flusser,J. (Czech Republic)
Gagalowicz,A. (France)
Galo,M. (Brazil)
Geraud,T. (France)
Giannini,F. (Italy)
Gudukbay,U. (Turkey)
Gutierrez,D. (Spain)
Haber,J. (Germany)
Hadwiger,M. (Austria)
Haro,A. (United States)
Harris,M. (United Kingdom)
Hast,A. (Sweden)
Hauser,H. (Austria)
Havran,V. (Germany)
Hege,H. (Germany)
Hladuvka,J. (Slovakia)
Horain,P. (France)
Hornung,A. (Germany)
Chen,M. (United Kingdom)
Chin,S. (Korea)
Chover,M. (Spain)
Chrysanthou,Y. (Cyprus)
Iwanowski,M. (Poland)
Jaillet,F. (France)
Jansen,F. (Netherlands)
Jeschke,S. (Germany)
JoanArinyo,R. (Spain)
Kalra,P. (India)
Kjelldahl,K. (Sweden)
Klosowski,J. (United States)
Kobbelt,L. (Germany)
Kolcun,A. (Czech Republic)
Koutek,M. (Netherlands)
Krolupper,F. (Czech Republic)
Kruijff,E. (Germany)
Larsen,B. (Denmark)

Leopoldseder,S. (Austria)
Lewis,J. (United States)
Lintu,A. (Germany)
Loizides,A. (Cyprus)
Loizides,A. (Cyprus)
Magnor,M. (Germany)
Maierhofer,S. (Austria)
Mandl,T. (Germany)
Mantler,S. (Austria)
Margala,M. (United States)
Marinov,M. (Germany)
Maughan,C. (USA)
McAllister,D. (USA)
McMenemy,K. (United Kingdom)
Mertens,T. (Belgium)
Moccozet,L. (Switzerland)
Mokhtari,M. (Canada)
Moltedo,L. (Italy)
Montrucchio,B. (Italy)
Moreton,H. (USA)
Mudur,S. (Canada)
Mueller,K. (United States)
Muller,H. (Germany)
Myszkowski,K. (Germany)
Neubauer,A. (Austria)
Nielsen,F. (Japan)
O'Sullivan,C. (Ireland)
Ozguc,B. (Turkey)
Pan,Z. (China)
Pandzic,I. (Croatia)
Pasko,A. (Japan)
Pedrini,H. (Brazil)
Perez,M. (Spain)
Peroche,B. (France)
Plemenos,D. (France)
Post,F. (Netherlands)
Prakash,E. (Singapore)
Pratikakis,I. (Greece)
Prikryl,J. (Czech Republic)
Puppo,E. (Italy)
Purgathofer,W. (Austria)
Rauterberg,M. (Netherlands)
Renaud,c. (France)
Revelles,J. (Spain)
Rheingans,P. (United States)
Rodrigues,M. (United Kingdom)
Rokita,P. (Poland)
Rossignac,J. (United States)
Rudomin,I. (Mexico)
Sainz,M. (USA)
Sbert,M. (Spain)
Segura,R. (Spain)
Shamir,A. (Israel)
Schaller,N. (United States)
Schneider,B. (United States)
Scholz,V. (Germany)
Schumann,H. (Germany)
Sijbers,J. (Belgium)
Sips,M. (Germany)
Sirakov,N. (United States)
Sitte,R. (Australia)
Slusallek,P. (Germany)
Snoeyink,J. (United States)
Sochor,J. (Czech Republic)
Sorel,M. (Czech Republic)
Sroubek,F. (Czech Republic)
Stuerzlinger,W. (Canada)
Stylianou,G. (Cyprus)
Suarez Rivero,J. (Spain)
Sumanta,P. (United States)
Szekely,G. (Switzerland)
Szirmay-Kalos,L. (Hungary)
Tang,W. (United Kingdom)
Taubin,G. (United States)
Teschner,M. (Germany)
Theobald,C. (Germany)
Theoharis,T. (Greece)
Theußl,T. (Austria)
Tobler,R. (Austria)
Torres,J. (Spain)
Trahanias,P. (Greece)
Traxler,A. (Austria)
Van Laerhoven,T. (Belgium)
Velho,L. (Brazil)
Veltkamp,R. (Netherlands)
Vergeest,J. (Netherlands)
Vuorimaa,P. (Finland)
Weiskopf,D. (Germany)
Weiss,G. (Germany)
Westermann,R. (Germany)
Wu,S. (Brazil)
Wuethrich,C. (Germany)
Yilmaz,T. (Turkey)
Zach,C. (Austria)
Zachmann,G. (Germany)
Zara,J. (Czech Republic)
Zemcik,P. (Czech Republic)
Zhu,Y. (United States)
Zitova,B. (Czech Republic)

WSCG 2005

Short papers proceedings

Contents

Session SH4 - Chair: Christopher Zach (Austria)

Session SH5 - Chair: Josef Pelikan (Czech Republic)

Algebraic Representation of CSG Solids Built from Free-Form Primitives

Á.L. García, J. Ruiz de Miras, F.R. Feito
Departamento de Informática. Universidad de Jaén
Paraje Las Lagunillas s/n
23071, Jaén. Spain
{algarcia, demiras, ffeito}@ujaen.es

ABSTRACT

A mathematical model for free-form solid modelling was presented in previous published works. The key aspects of this model are the decomposition of the volume occupied by the solid into non-disjoint cells, and the representation of the solid as an algebraic sum of these cells. Here we apply this scheme to represent CSG solids built by combining free-form solids in boolean operations. As a proof of the validity of this scheme, we present an algorithm that allows us to visualize the non-evaluated result of the operations. We have worked with free-form solids whose surfaces are bounded by a set of low degree triangular parametric patches.

Keywords

Cell decomposition, boolean operations representation, free-form solid modelling.

1. INTRODUCTION

Solid modelling can be defined as a consistent set of principles for mathematical and computer modelling of three-dimensional solids [Sha02]. One open issue in solid modelling is the representation in an exact, useful and efficient way of solids bounded by free-form elements (curves in 2D, curved surfaces in 3D, etc.). These are usually named free-form solids.

Operations like visualization, boolean combination, surface determination, collision detection and so on, depend on the representation scheme used for solids; some schemes enhance the performance of some operations, while others are intended to be as versatile as possible.

The Extended Simplicial Chain (ESC) model was presented as a mathematical model to represent free-form solids in a simple and robust way by means of an algebraic decomposition [RF99b, GRF03]. Based on this scheme, algorithms have been developed to test the inclusion of a point [RF99a, GRF04], obtain a B-Rep or a voxel representation [RF02] of free-form solids.

Here we apply the ESC model to develop a method to obtain an algebraic decomposition of CSG solids built by applying boolean operations to a set of free-form primitives bounded by triangular Bézier patches [Far86]. Notice that the ESC model is not only a boundary representation, but also a volume representation scheme.

WSCG'2005 Short papers proceedings. ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. PREVIOUS WORK

Constructive Solid Geometry (CSG) is one of the most popular representation schemes for solid modelling [RV77]. CSG solids are built as boolean combinations of simpler solids (primitives). The classic data structure associated with this scheme is a binary tree, where the non-terminal nodes represent boolean operations, and the terminal nodes store either primitives or transformations.

There appear problems when applying the CSG approach to free-form solid modelling, mainly related to boundary evaluation and visualization. Using a reduced set of free-form primitives and exact computation [Key00] is one way to solve this; other solutions get approximate results with variable cost and precision [BKZ01].

The visualization of CSG solids is another important field of research. Many different approaches have been studied: using polyhedral approximations (lossy precision) [RS97], complex mathematical developments (numerical stability problems) [KGMM97], ray shooting techniques (computationally expensive) [HG96], or modifying the rendering pipeline (memory expensive and complexity) [SLJ02].

Feito et al. proposed the algebraic decomposition of polyhedral solids in previous works [FR98]; this was the Simplicial Chain (SC) model. Ruiz et al. extended the SC model to free-form solids, this is the Extended Simplicial Chain (ESC) model [RF99b]. The application of this scheme to free-form solids bounded by triangular Bézier patches has been recently developed [GRF03].

3. THE ESC MODEL

The Extended Simplicial Chain (ESC) model use a *divide and conquer* approach to model free-form solids. The solids are decomposed into a set of non-disjoint *extended cells*

of two types: simplices and free-form cells. Each cell has an associated sign; the volume of positive cells is added to the solid, and the volume of negative cells is subtracted from the solid. This decomposition scheme allows us to develop algorithms just by combining in a sum the results of computing the test locally in the individual cells.

The ESC model is intended for representing general free-form solids. Up to now, the model has been successfully applied to 2D solids bounded by conics and Bézier curves [RF97], and 3D solids bounded by triangular algebraic patches [RF99b, Rui01] and triangular Bézier patches [GRF03].

In this work, the starting point to construct the ESC associated to a 3D free-form solid is a triangle mesh that approximates the shape of the solid. The surface of the solid is built from these triangles [VPBM01, HB03].

Simplices

A d-dimensional simplex is defined as the convex hull of *d+1* affinely independent points. We use 2D simplices for 2D solids, 3D simplices for 3D solids and so on.

In our model, each 3D simplex is built by joining one arbitrary point (e.g. the origin of coordinates) with the vertices of each triangle. These are named *original* simplices.

We compute the associated sign of a 3D simplex as the sign of its signed volume, which can be computed as:

$$\frac{1}{6}\cdot\begin{vmatrix} T_1.x & T_2.x & T_3.x & T_4.x \\ T_1.y & T_2.y & T_3.y & T_4.y \\ T_1.z & T_2.z & T_3.z & T_4.z \\ 1 & 1 & 1 & 1 \end{vmatrix}$$

T_1, T_2, T_3, T_4 being the vertices of the simplex.

Figure 1 shows two original simplices (tetrahedra).

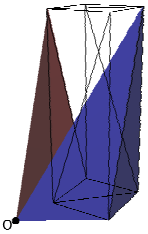


Figure 1. Wireframe model, positive (blue) and negative (red) original simplices

Free-form Cells

A general d-dimensional free-form cell (ffc) is defined as a set of points in $\mathbf{R}^d$, obtained as the intersection of the half-spaces defined by a free-form element of dimension *d*, and one or more planar elements of dimension *(d-1)* that verifies a number of properties ensuring closure and conectivity (see [RF99b, GRF03] for details).

The ffc as defined in the bibliography needs to be specified in more detail, as the number and computation of the planar elements depend on the type of free-form elements chosen to bound the solids. This time, we are working with triangular Bézier patches [Far86], and the planar elements necessary to completely close the ffc are the plane containing the base triangle of the main patch of the cell and three additional planes (called *associated planes*); each plane contains one of the edges of the base triangle of the main patch, and is parallel to the weighted average vector of the normal vectors from the two triangles that share that edge. An ffc is bounded by the mentioned planar elements, the main patch, and all the patches that share a vertex with the main patch and intersect the planar elements [GRF04].

Figure 2 left shows the bounding elements of an ffc. The associated planes are drawn as quadrilaterals; the main triangular Bézier patch of the ffc is drawn in shaded mode, and one neighbour patch that also bounds the ffc is drawn in wireframe mode. Figure 2 right shows the ffc as a solid.

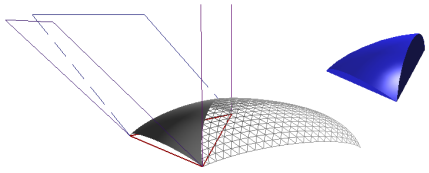


Figure 2. Left: bounding elements of an ffc. Right: solid view of the same ffc

Considering the triangle vertices disposed counterclockwise, an ffc is positive if it is in the positive half-space defined by the plane that contains the base triangle, and negative if it is in the negative one. If the patches pass through the plane that contains the base triangle, each connected component is considered a different ffc.

Extended Simplicial Chains

A (3-dimensional) Extended Simplicial Chain, δ, is defined as $\delta = \sum_i \alpha_i \cdot E_i$. E_i being extended cells, each of them multiplied by an integer α_i (its coefficient). There is an integer-valued function *associated* to each ESC, which is defined in the following way:

$$f_\delta : \mathbf{R}^3 \rightarrow \mathbf{Z}; \qquad f_\delta(\mathbf{Q}) = \sum_{\mathbf{Q}\in E_i} \alpha_i$$

For a point **Q** of the space, f_δ is the sum of the coefficients associated with the extended cells that contain **Q**.

The associated function of an ESC is different from zero only for those points that belong to the free-form solid represented by δ (noted FF_δ). There is a simple point in solid test based on this property [RF99a, GRF04].

The same solid can be represented by several ESC's. We define a *normal extended simplicial chain* as the one that verifies $f_\delta(\mathbf{Q}) = 1 \qquad \forall \mathbf{Q} \in FF_\delta$.

From now on, we will consider only normal ESC's.

3.3.1 *Operating with ESC's*

Let $\delta = \sum_{i=1}^{n} \alpha_i \cdot E_i$ and $\delta' = \sum_{j=1}^{m} \alpha_j \cdot E_j$ be two ESC's, and λ a scalar value. The sum of two ESC's, and the product of an ESC by a scalar value are defined as:

$$\delta + \delta' = \sum_{i=1}^{n} \alpha_i \cdot E_i + \sum_{j=1}^{m} \alpha_j \cdot E_j;$$

$$\lambda \cdot \delta = \sum_{i=1}^{n} (\lambda \cdot \alpha_i) \cdot E_i;$$

The regularization of these operations is described in [RF99b, Rui01, GRF03]. We consider always regularized operations.

4. REPRESENTATION OF BOOLEAN OPERATIONS

Using the ESC model, boolean operations can be represented applying the *divide and conquer* approach to reduce them to a combination of operations with extended cells, which are much simpler than the original solids.

Let us note the intersection of extended cells E and E' as *ExCell(E∩E')*. Computing *ExCell(E∩E')* involves computing the intersection of the bounding elements of E and E'. It would be also necessary to define a new type of extended cell to represent the sets of points bounded by fragments of the bounding elments of E and E'. At the moment, we are working on this topic. An alternative option for this process consists of decomposing the intersection in an approximative way, using smaller simplices and ffc's to represent the resulting sets.

Using ESC's to represent solids and boolean operations allows us to delay the computation of *ExCell(E∩E')* depending on how the result will be applied. As the volume of positive and negative cells often compensates, the evaluation of *ExCell(E∩E')* can be avoided for cells whose intersection do not add volume to the final solid.

Now we present the expressions for the representation of boolean operations. Theorems establishing these formulas and their demonstrations can be found in [Rui01, GRF03].

Let FF_1 and FF_2 be two free-form solids, and let $\delta_1 = \sum_{i=1}^{n} \alpha_i \cdot E_i$, and $\delta_2 = \sum_{j=1}^{m} \alpha_j \cdot E_j$ be their associated normal ESC's. The associated normal ESC for the solid obtained as the intersection $FF_\delta = FF_1 \cap FF_2$ is:

$$\delta = \sum_{i=1}^{n} \sum_{j=1}^{m} (\alpha_i \cdot \alpha_j) \cdot ExCell(E_i \cap E_j);$$

In practice, if there is no intersection between one E_i and one E_j, *ExCell*($E_i \cap E_j$) is an empty set, and therefore it can be deleted from the sum.

The union and difference operations can be expressed using the intersection:

Let $\delta_{FF1 \cap FF2}$ be the ESC of the intersection of FF_1 and FF_2. The normal ESC for the solid obtained as the union $FF_\delta = FF_1 \cup FF_2$, is $\delta = \delta_1 + \delta_2 - \delta_{FF_1 \cap FF_2}$, the normal ESC of the difference of FF_1 and FF_2, noted $FF_\delta = FF_1 - FF_2$ is $\delta = \delta_1 - \delta_{FF_1 \cap FF_2}$, and finally, the extended simplicial chain for the solid obtained by applying the complement operation on a free-form solid, noted ${FF_1}^C$ is $\delta_C = \delta_R - \delta_1$, where δ_R verify $f_{\delta_R} = 1 \ \forall \mathbf{Q} \in \mathbf{R}^3$. In practice, δ_R can be implemented as a cube that contains the whole scene.

5. REPRESENTATION OF CSG SOLIDS

The expressions from section 4 allow us to build an ESC for the result of a boolean operation between two free-form solids represented by ESC's. Since the result of this operation is another ESC, it can be used again as an operand in a new boolean operation. If we repeat this process, we can obtain a chain for the final result of a whole tree of boolean operations to design a complex CSG solid.

Figure 3 shows an example of a CSG tree where the leaf nodes are free-form solids represented by ESC's.

By applying the ESC model, we can obtain an homogeneous representation for both free-form primitives and CSG solids, and algorithms designed for free-form solids with ESC's can be easily applied to CSG solids [RF02].

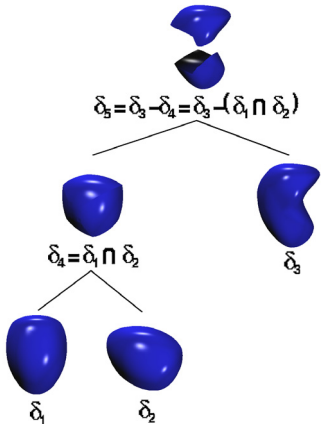


Figure 3. Example of CSG tree

As it was mentioned in section 4, the intersection of extended cells is a very important issue to be considered when creating the ESC that represents the result of a boolean operation. This can be handled in two ways:

- Compute the intersection of the cells and construct a new ESC that represents the resulting set of points. This can be done by defining a new type of extended cell to represent the sets of points, or by decomposing the result in simplices and ffc's. The second choice can result in a huge number of cells that can make the resulting ESC useless, while the first one seems more efficient.
- Do not evaluate the intersection, but keep a pointer to each extended cell instead. In this case, when applying an operation on an intersection of cells it is necessary to operate on both cells, and then combine the results.

Proceeding as explained in the first option will produce a completely evaluated CSG solid. The second option corresponds to a classic non-evaluated CSG solid.

6. VISUALIZATION OF NON-EVALUATED CSG SOLIDS

Based on the point in solid test mentioned in section 3.3, we have developed a visualization algorithm for solids based on ray tracing techniques [Gla93]. The algorithm consists basically of the following steps:

- Cast a ray through the centre of each pixel.
- Compute all the intersections of the ray with each cell.
- Use the point in solid test to find the first intersection point that belongs to the solid.
- Use the colour properties of that intersection point to draw the corresponding pixel as usual.

To render images of a non-evaluated CSG solid, we just have to consider the second option from section 5 to represent the intersection of extended cells. To compute the intersection of a ray with *ExCell(E∩E')*, simply compute the intersections of the ray with *E* and *E'*, and then study the position of the intersection points of *E* with regard to *E'* and vice versa. A point will be considered as an intersection with *ExCell(E∩E')* only if it belongs to *E* and *E'*.

Figure 4 shows a pair of free-form solids and the visualization of the boolean operations applied to them.

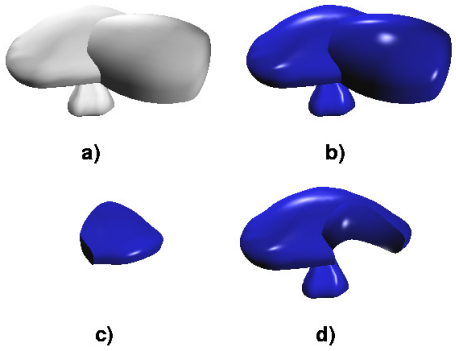


Figure 4. a) Two free-form solids. b) Union. c) Intersection. d) Difference

The visualization algorithm also makes it possible to get partial images of a given solid. This is possible by passing portions of the ESC as arguments for the visualization algorithm. Doing this allows us to observe how the positive and negative cells compensate to obtain the final result.

7. CONCLUSIONS

The application of a mathematical model to the representation of CSG solids built from free-form primitives has been presented. This model makes it possible to decompose the solids in an algebraic expression, which allows a simple and uniform handling of the primitives and the final solids just by working with the individual cells and then combining the results. This decomposition considers the solid not only as its boundary, but also as a volume, and the cells are allowed to intersect with other cells. This feature makes the decomposition process very simple.

The utility of the model to develop robust and simple algorithms has been proved in previous published works [RF99a, RF02, GRF04], and also in this paper, where an algorithm to render images of non-evaluated CSG solids based on ray shooting techniques has been presented.

8. ACKNOWLEDGEMENTS

This work has been partially supported by the Ministry of Science and Technology of Spain and the European Union by means of the ERDF funds, under the research projects TIC2001-2099-C03-03 and TIN2004-06326-C03-03.

The work of Á.L. García was supported by a Junta de Andalucía (Spain) fellowship.

9. REFERENCES

[BKZ01] Biermann, H. Kristjansson, D. Zorin, D. Approximate boolean operations on free-form solids. In ACM Siggraph conf. proc., Los Angeles, ACM Press, pp. 185-194. 2001

[Far86] Farin, G. Triangular Bernstein-Bézier patches. *Computer Aided Geometric Design*, Vol. 3, 83-127. 1986.

[FR98] Feito, F.R. Rivero, M. Geometric modelling based on simplicial chains. *Computers & Graphics*, Vol. 22 (5), 611-619. 1998.

[Gla93] Glassner, A. (Ed.). An Introduction to Ray Tracing. Academic Press, 1993.

[GRF03] García, Á.L. Ruiz de Miras, J. Feito, F.R. Free-form solid modelling based on extended simplicial chains using triangular Bézier patches. *Computers & Graphics*, Vol. 27 (1), 27-39. 2003.

[GRF04] García, Á.L. Ruiz de Miras, J. Feito, F.R. Point in solid test for free-form solids defined with triangular Bézier patches. *The Visual Computer*, Vol. 20 (5), 298-313. 2004.

[HB03] Hahmann, S. Bonneau, G.-P. Polynomial surfaces interpolating arbitrary triangulations. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 9 (1), 99-109. 2003.

[HG96] Hasenfratz, J.-M. Ghazanfarpour, D. Rendering CSG scenes with general antialiasing. In CSG 96 Set-theoretic Solid Modelling: Techniques and Applications. pp. 275-289. Informations Geometers Ltd. 1996.

[Key00] Keyser, J. Exact Boundary Evaluation for Curved Solids. PhD Thesis, University of North Carolina at Chapel Hill. 2000.

[KGMM97] Krishnan, S. Gopi, M. Manocha, D. Mine, M. Interactive boundary computation of boolean combinations of sculptured solids. *Computer Graphics Forum*, Vol. 16 (3), 67-78. 1997.

[RF97] Ruiz de Miras, J. Feito, F.R. Inclusion test for curved edge polygons. *Computers & Graphics*, Vol. 21 (6), 815-824. 1997.

[RF99a] Ruiz de Miras, J. Feito, F.R. Inclusion test for free-form solids. *Computers & Graphics*, Vol. 23 (2), 255-268. 1999.

[RF99b] Ruiz de Miras, J. Feito, F.R. Mathematical free-form solid modeling based on extended simplicial chains. *Journal of WSCG*, Vol. 7, 241-248. 1999.

[RF02] Ruiz de Miras, J. Feito, F.R. Direct and robust voxelization and poligonalization of free-form CSG solids. 3DVPT 2002: 1st International Symposium on 3D Data Processing, Visualization and Transmission. 2002.

[RS97] Rappoport, A. Spitz, S. Interactive boolean operations for conceptual design of 3-d solids. ACM 24th Conference on Computer Graphics & Interactive Techniques. 269-278. 1997.

[Rui01] Ruiz de Miras, J. Free-Form Solid Modeling. PhD Thesis, Universidad de Granada. 2001.

[RV77] Requicha, A. Voelcker, H. Constructive Solid Geometry. Tech. Memo 25, Production Automation Project, University of Rochester. 1977.

[Sha02] Shapiro, V. Solid modeling. In Farin, G. Hoschek, J. Kim, M.-S. (eds.) Handbook of Computer Aided Geometric Design. North-Holland, 2002.

[SLJ02] Stewart, N. Leach, G. John, S. Linear-time CSG rendering of intersected convex objects. *Journal of WSCG*, Vol. 10, 437-444. 2002.

[VPBM01] Vlachos, A. Peters, J. Boyd, C. Mitchell, J. Curved PN triangles. ACM Symposium on Interactive 3D Graphics, 159-166. 2001.

Spacetime Catmull Recursive Subdivision Facilitated with Occlusion Culling

Andreas Genz
FB3, Grafische Datenverarbeitung, Universität Bremen
Linzerstr. 9a, 28359 Bremen, Germany
genz@acm.org

ABSTRACT

We describe an extension and a generalization of the Catmull recursive subdivision algorithm: first, an image-based occlusion culling stage is added; second, all rendering stages, that is, view-frustum and occlusion culling, subdivision of geometric primitives into micropolygons, and rasterization, are performed in spacetime.
Operating in spacetime allows to exploit temporal coherence in animated scenes.

Keywords

image generation, rendering, spacetime rendering, occlusion culling, hidden surface removal, performance

1. INTRODUCTION

The following rendering algorithm is optimized for very complex dynamic scenes preferring a procedural description of geometry. Main objective is to produce and process geometry only if it affects the synthesized animation. We choose the Catmull recursive subdivision algorithm as a foundation, because it was designed with procedurally described geometry in mind and it is the ancestor and back-bone of the REYES architecture, which is in use for most movie productions these days. We insert an occlusion culling stage with a summed-area table as its occlusion representation. Also, we generalize the complete rendering pipeline to operate in spacetime. The occlusion representation becomes a summed-volume table then.

2. PREVIOUS WORK

In 1988, Andrew Glassner published implementation results of accomplishing ray tracing in spacetime [Gla88]. In 1999, Damez and Sillion have shown how to perform radiosity calculations in spacetime [DS99]. And in 2003 Havran et al. picked up the task of ray tracing in spacetime again [HDMS03]. But to our knowledge, no one ever tried to realize a spacetime z-buffer rendering pipeline.

SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

An extensive overview on general occlusion culling techniques is given by Cohen-Or et al. [COCSD03]. Image-based occlusion culling was pioneered by Ned Greene [GKM93]. Variants of his hierarchical z-buffer are used in today's hardware [AMN03] and in the prman renderer—Pixar's implementation of the REYES architecture—to reduce the number of shading calculations [AGB99].
Another candidate for an image occlusion representation is the summed-area table, which is in common use for texture mapping today [Cro84]. It was used for volume rendering by Huang et. al. [HCSM00], for occlusion culling by Ho and Wang [HW99], and the author [Gen01].
Catmull introduced in his dissertation a rendering algorithm [Cat74] fetched up by Lucasfilm's rendering group to realize the REYES architecture [CCC87, AGB99]. This architecture may be a good candidate for future realtime rendering pipelines implemented in hardware [OKTD02].

3. THE CATMULL RECURSIVE SUBDIVISION ALGORITHM

Z-buffer algorithms hold the advantage to address surfaces independently of each other.
Subdivision splits an object into more than one smaller objects, the sum of the smaller objects represents the original one, and the smaller objects can be handled independently. Subdivision of the scene space (for example by octrees), of the raster image space into buckets or a hierarchy, of the camera space into layers along

the z-axis as will be shown later, and of geometric primitives in different spaces are plausible.
The REYES architecture takes this two concepts—z-buffer and subdivision—devised by Catmull [Cat74] and extends them by incremental methods to dice each surface into micropolygons, by shading micropolygons with texture filtering, and by subpixel rasterization with jittered samples for antialiasing [CCC87].

Algorithm C (*Catmull's recursive subdivision algorithm*). Given: camera, surfaces. Find a raster image of the scene

For each surface:
C1. Compute camera-space axis-aligned bounding box.
C2. If bounding box is entirely outside viewing volume:
drop surface.
C3. Else: project bounding box into image space.
C4. If area of bounding rect larger than one pixel:
C5. subdivide surface.
C6. Else: rasterize surface, z-buffer visibility test.

Catmull's recursive subdivision algorithm can be seen as the first stage (splitting) of the REYES pipeline plus a z-buffer. Surfaces are subdivided in parameter space (**C5**) until the resulting face becomes small enough, such that the area of its bounding rect is smaller than one pixel (**C4**). Then, one point inside the surface is tested against the z-buffer for visibility, and its color is written to the color buffer if it passes the test (**C6**). During subdivision, if an surface is outside the viewing volume, it is dropped. For purpose of efficiency and code minimalism, tests are not done on the surface directly, but on its bounding box. Therefore, every face must have the following properties to be used: it must have a well defined bounding box and subdivision procedure.
The Catmull recursive subdivision algorithm accomplishes hierarchical view-frustum culling and level-of-detail implicitly. Ideally, the whole scene can be described as one surface that is subdivided during the rendering process, so that unnecessary geometry is not stored and processed at all.
Here we make use of this algorithm, because it is well suited for a simple implementation to measure the number of required recursive subdivisions. Our goal is to reduce this number by occlusion culling and by exploiting temporal coherence, so that the following pipeline stages of a fictional REYES architecture would not be fed with too many geometric primitives.

4. SUMMED-AREA TABLE OCCLUSION CULLING

For scenes with high depth-complexity processed by traditional z-buffer visibility algorithms, a huge amount of scene geometry entirely hidden by other geometry is rendered. Although, hidden geometry does not contribute to the requested image if we restrain light models to be local.
To overcome this, we add a simple occlusion culling method to the Catmull recursive subdivision algorithm. The algorithm is slightly variated by exchanging **C3** with **C2**; the bounding rect given by the projected bounding box is used for view-frustum culling, occlusion region estimation, and area measurement (subdivision criteria).

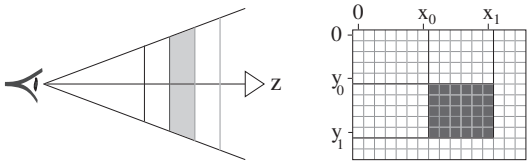


Figure 1: Depth layers and summed-area table in raster space

Our image based occlusion culling algorithm using a summed-area table is defined as follows:
a. Split scene into layers along the z-axis, see the left side of figure 1. Necessary depth sorting for occlusion culling is done implicitly via hierarchical view frustum culling.
b. Render the front most layer with the Catmull recursive subdivision algorithm.
c. Build a summed-area table $T(x,y)$ of the function $c(x,y)$, where $c(x,y) = 1$, if pixel (x,y) is covered by some surfaces, otherwise $c(x,y) = 0$. The summed-area table is the occlusion representation.
d. Render the next layer with a modified Catmull recursive subdivision algorithm:

Algorithm C´ (*Catmull's recursive subdivision algorithm facilitated with occlusion culling*). Given: camera including layerdepth, surfaces. Find a raster image of the scene

For each surface:
C´1. Compute camera-space axis-aligned bounding box.
C´2. Project bounding box into image space.
C´3. If bounding rect is entirely outside image borders:
drop surface.
C´4. Else: if area of bounding rect is larger than one pixel:
C´O. If bounding rect is totally occluded:
drop surface.
C´5. Else: subdivide surface.
C´6. Else: rasterize surface, z-buffer visibility test.

e. Return to step **c**, unless the image has been totally covered or the last layer is reached.
In step **C´O**, let us assume the bounding rect has two corners (x_0,y_0) and (x_1,y_1), see also the right side of figure 1. Denote the number of pixels covered by the bounding rect by B and compute $B = (x_1 - x_0)(y_1 - y_0)$. Denote the number of pixels covered by some surfaces in this bounding rect by A and compute

$A = T(x_1,y_1) - T(x_1,y_0) - T(x_0,y_1) + T(x_0,y_0)$. Also, denote an error tolerance $\varepsilon \geq 0$.
It can be seen that the surface is totally occluded and can be dropped, if $A \geq B - \varepsilon$.

Figure 2: Scene of trees.

Figure 2 shows a case study: the rendering of a scene containing approximately one hundred thousand trees with geometrically different features. The scene has been described procedurally, we make use of data amplification [EWM+98]. Our software implementation of the outlined algorithm is three times faster than the native Catmull subdivision algorithm for the rendition of this image. But runtime is highly dependent on the geometric model and the view-point.

Figure 3: Depth-of-field and tollerance culling.

Properties of using a summed-area table as the occlusion representation are as follows: (1) An easy to understand and straight forward to implement method. (Think about the z-buffer compared to more elaborate hidden surface algorithms.) (2) A relatively long update time compared to the z-pyramid used for the hierarchical z-buffer. However, computing the summed-area table can be accomplished in parallel to rendering the next layer. (3) A fast occlusion test and efficient culling of small and thin surfaces. (4) A simple way to achieve occlusion culling with tolerances, for that surfaces do not have to be totally occluded to be dropped, see figure 3, left for an excessive appliance for illustrative purposes. Furthermore, using layers gives the potential to generate computationally cheap depth-of-field effects, as shown in figure 3, right. Different layers can be rendered even in different resolutions resulting in reduced rendering time.

5. SUBDIVISION IN SPACETIME

Traditionally, rendering is handled by a mapping from a 3D scene description to a 2D raster image: $R^3 \rightarrow N^2$. However, rendering an animation is a 4-dimensional problem. We generalize this mapping process from a 4D scene description including motion descriptions to a 3D raster image block—a sequence of 2D images: $R^4 \rightarrow N^3$.

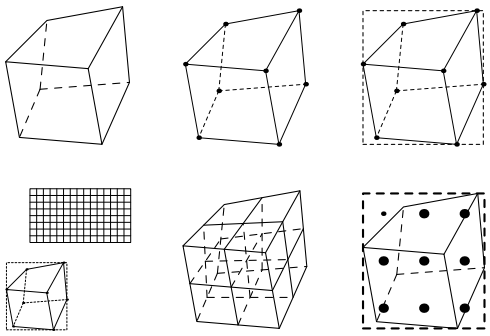


Figure 4: Illustrations for spacetime subdivision denoted in left-to-right order: trilinear body; steps **S1**, **S2**, **S3**, **S4**, and **S6** of algorithm S.

In computer graphics bodies are usually reduced to their surfaces, the things that are directly visible. In a minimalistic case study, we address linear motions of the control points of a bilinear patch: 3D control points are linearly interpolated in time. We define a trilinear body in spacetime, given by 8 control points: $P_0, P_1, \ldots, P_7; P_i \in R^4$, see figure 4.

Algorithm S (*Recursive subdivision algorithm operating in spacetime*). Given: camera, bodies. Find a raster image block.

For each body:
S1. Project control points to image spacetime.
S2. Compute bounding block in image spacetime.
S3. If bounding block is entirely outside view spacetime:
drop body.
S4. Else: if area of bounding rect in image space larger
than area of one pixel:
S5. Subdivide body.
S6. Else: rasterize body, z-buffer visibility test.

We add an occlusion culling stage analogous to that of the last chapter: Corners of the bounding block in raster image spacetime become then (x_0,y_0,t_0) and (x_1,y_1,t_1). The Number of pixels covered by this block is $\#B_S = (x_1-x_0)(y_1-y_0)(t_1-t_0)$, and the number of pixels covered by some body in this block computed with a summed-volume table T_S becomes:

$$\begin{aligned} A_S \quad = \quad & T_S(x_1,y_1,t_1) - T_S(x_0,y_1,t_1) - T_S(x_1,y_0,t_1) - \\ & T_S(x_1,y_1,t_0) + T_S(x_1,y_0,t_0) + T_S(x_0,y_1,t_0) + \\ & T_S(x_0,y_0,t_1) - T_S(x_0,y_0,t_0) \end{aligned}$$

An body is totally occluded for all frames in $[t_0, t_1]$ and can be dropped, if $A_S \geq B_S - \varepsilon$.

Figure 5: Trilinear bodies moving in spacetime.

In figure 5, the light patch is not moving and so the number of its subdivisions is independent of the number of frames. On the right side, it is culled by only one occlusion test for the last two frames (micropolygons are randomly graded to show their structure).

6. DISCUSSION AND FUTURE WORK

We made some observations that may influence future work for spacetime recursive subdivision:
Bounding boxes are too large in higher dimensions. Our implementation shows an explosion in the number of subdivisions for complex bodies. One way to address these problems is to perform anisotropic subdivisions: to subdivide not in all parameter spaces at once. This is also beneficial for texture filtering.
The layer depth should be changed adaptively in dependance of the number of newly covered raster elements of the last rendered layer. To adapt our algorithm to interactive real-time rendering, a prediction function would be required measuring the probability of future events.
An A-buffer rasterization [Car84] with volume samples represented by bit masks would be beneficial.
Geometric primitives and motion descriptions that are well suited for subdivision in space time have to be found. Furthermore, a known scene description language has to be adapted or a new one has to be developed.
Our approach consumes a huge amount of memory for z-buffer, raster image, and summed-volume table. Subdivision of the image into buckets [CCC87] can help, although some spacial coherence at the border of buckets will be lost.

7. ACKNOWLEDGEMENT

Warm thanks go to Daehyun Kim, Frieder Nake, Caroline von Totth, Eric Allen Engle, anonymous WSCG reviewers, and Eva.

8. REFERENCES

[AGB99] Apodaca A. A., Gritz L., and Barsky B. A. *Advanced RenderMan: Creating CGI for Motion Picture.* Morgan Kaufmann Publishers Inc., 1999.

[AMN03] Aila T., Miettinen V., and Nordlund P. Delay streams for graphics hardware. *ACM Trans. Graph.*, 22(3):792–800, 2003.

[Car84] Carpenter L. The a-buffer, an antialiased hidden surface method. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 103–108. ACM Press, 1984.

[Cat74] Catmull E. E. *A subdivision algorithm for computer display of curved surfaces.* PhD thesis, University of Utah, 1974.

[CCC87] Cook R. L., Carpenter L., and Catmull E. The reyes image rendering architecture. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 95–102. ACM Press, 1987.

[COCSD03] Cohen-Or D., Chrysanthou Y. L., Silva C. T., and Durand F. A survey of visibility for walkthrough applications. In *Transactions on visualization and computer gaphics, vol. 9, no. 3*, pages 412–431. IEEE Computer Society, 2003.

[Cro84] Crow F. C. Summed-area tables for texture mapping. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 207–212. ACM Press, 1984.

[DS99] Damez C. and Sillion F. Space-time hierarchical radiosity. In *Rendering Techniques '99*, pages 235–246, New York, NY, 1999. Springer Wien.

[EWM+98] Ebert D. S., Worley S., Musgrave F. K., Peachey D., Perlin K., and Musgrave K. F. *Texturing and Modeling*. Academic Press, Inc., 1998.

[Gen01] Genz A. Occlusion culling and summed-area tables. In *SIGGRAPH'2001: Conference Abstracts and Applications*, page 238. ACM SIGGRAPH, August 2001.

[GKM93] Greene N., Kass M., and Miller G. Hierarchical z-buffer visibility. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 231–238. ACM Press, 1993.

[Gla88] Glassner A. S. Spacetime ray tracing for animation. *IEEE Computer Graphics and Applications*, 8:60–70, March 1988.

[HCSM00] Huang J., Crawfis R., Shareef N., and Mueller K. Fastsplats: optimized splatting on rectilinear grids. In *Proceedings of the conference on Visualization '00*, pages 219–226. IEEE Computer Society Press, 2000.

[HDMS03] Havran V., Damez C., Myszkowski K., and Seidel H.-P. An efficient spatio-temporal architecture for animation rendering. In *Proceedings of the Eurographics Symposium on Rendering, vol. 14*, pages 106–117, June 2003.

[HW99] Ho P. C. and Wang W. Occlusion culling using minimum occluder set and opacity map. In *IEEE International Conference on Information Visualization*, pages 292–300, 1999.

[OKTD02] Owens J. D., Khailany B., Towles B., and Dally W. J. Comparing reyes and opengl on a stream architecture. In *Proceedings of the conference on Graphics hardware 2002*, pages 47–56. Eurographics Association, 2002.

Automatic Generation of Urban Zones

Mathieu Larive
OktalSE
2, impasse Boudeville
31100 Toulouse
France
mathieu.larive@oktal.fr

Yann Dupuy
OktalSE
2, impasse Boudeville
31100 Toulouse
France
yann.dupuy@oktal.fr

Véronique Gaildrat
IRIT-UPS
118 route de Narbonne
31062 Toulouse
France
gaildrat@irit.fr

ABSTRACT

We present a state of the art of techniques that can be implemented to automatically generate virtual cities (non existing ones). An urban zone constitutes a too vast volume of data to be directly understood, modeled or visualized. The multiresolution approach based on imbricated logical structures allows us to divide this volume of data into successive levels of detail.

Keywords

Urban modeling, architecture, procedural methods, declarative modeling

1. INTRODUCTION

Recent applications in virtual reality, video games and simulation of city expansion as well as the emergence of problems due to the urbanization, such as the influence of electromagnetic radiations and the forecast of the urban transportation network, creates increasing needs in term of digital mock-ups studies and forecasting. The capacity to quickly generate credible digital city models helps the user to fulfill these needs.

A real city satisfies construction rules and depends on multiple influences throughout the time. However the detailed modeling of realistic towns is very challenging for computer graphics. Modeling a virtual city which is detailed enough to be credible for a visiting user is a huge task that requires thousands hours of work. In this context, we think that automatic approaches are well suited for this problem. Hence, they represent a promising research topic which has to be developped because the current results do not satisfy the previously defined needs. For these reasons, we propose a state of the art of the current techniques to distinguish what exists and what remains to be studied.

Conference proceedings ISBN 80-903100-7-9
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. AUTOMATIC TECHNIQUES

Figure 1 presents a hierarchical division in seven stages of city generation. This division is used as the guideline of this paper to present the existing methods. According to the complexity of every stage, we propose a brief presentation of the formalism and the handled data as well as a discussion about the presented generation methods. A stage can be seen as a level of detail during a multi-scale representation of the city. Techniques described in this paper come from various domains such as artificial intelligence, operational research, town planning or declarative modeling.

2.1 Urban zone

We need to know at least the limits of a city to be able to generate it. We consider the definition of the surface that the city covers as the minimum level of information. This information is generally defined within a GIS (Geographic Information System). It is also possible to constraint the generation in order to respect data map: geographical (altitude, hydrography and vegetation) or socio statistical (population density, street patterns and elevation).

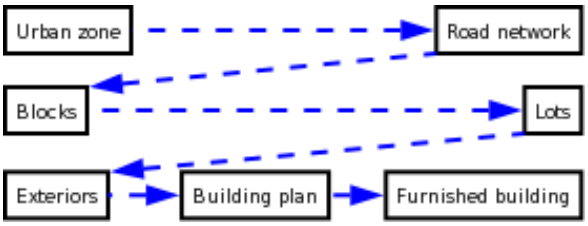


Figure 1: Hierarchical division of city generation

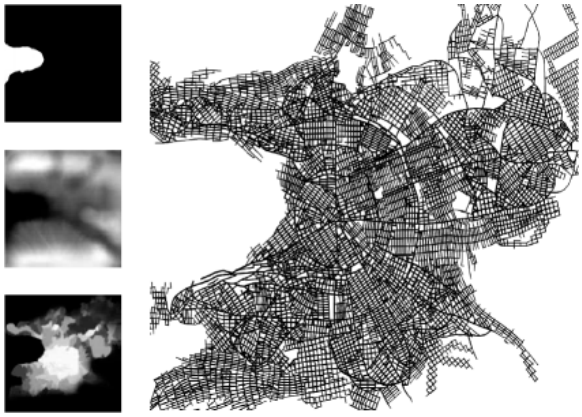


Figure 2: One possible roadmap generated from a set of maps [PM01]

2.2 Road network

Even if it seems illusory to try to characterize all existing cities according to predefined patterns of streets, some classic patterns can be used during the generation. For example, most of the cities in USA are designed according to a checkerboard pattern, whereas European cities tend to follow a radial/concentric pattern. Nevertheless, these observations are to be balanced by the fact that the structure of a city moves during time, according to geographic and socio-economic constraints. In practice, we tend to observe a composition of patterns within the same city.

L-System CityEngine[PM01] uses an extension of the L-System for the creation of the road network. Creating realistic road networks needs a lot of rules with a large number of conditions and parameters. Writing a new rule means rewriting numerous rules, so the extension of a system is actually a difficult task. In order to avoid the growth of the set of rules, Parish and Müller defined an extension of the L-Systems that creates generic successors at every generation stage: they named them *ideal successors*. The parameters of these successors are not instantiated, the call to the global goal function instantiates them in order to satisfy the global goals. Then, the system checks if these parameters satisfy the local goals and modifies them if not.

Declarative modeling of line segments In the field of declarative modeling for urban layout, [LH97] introduces the notions of urban elements that can be combined to obtain more complex urban patterns. The scene is described in a hierarchical and incremental way. The user begins with an approximate sketch, giving only the main features such as the main roads and crossings. This description is used to propose one or several solutions that the designer can refine. The generation process is based on the CSP approach.

Discussion The procedural method based on the L-Systems creates city-scale results, while the one based on declarative modeling suffers from the chosen generation method that does not allow it to generate results from a too vast search space. Nevertheless, the approach stemming from the declarative modeling describes urban patterns not yet managed within procedural approaches. To guarantee a bigger variety of generated road networks, it would be a good idea to integrate the not yet available urban patterns (as the diamond shaped one) into the L-System formalism.

2.3 Blocks

A block is a connected surface delimited by roads or streets. The partition of the city into blocks defines a first stage of hierarchical organization: this decreases the number of constraints to manage, for example in the case of the disjunction constraint. The used model for a block is the polygon, commonly only the convex polygons are used to take advantage of the lesser complex geometric algorithms for this kind of polygon.
The generation process of this stage is not at all complex compared to the previous stage. It can be seen as a simple intersection computation between the segments of the road network as in [PM01].

2.4 Lots

A *lot* is a surface of the same mailing address. Lots are the areas on which the buildings will be placed. It is possible to generate all the lots of a block then to generate again those that do not satisfy the user. There are several similarities with the blocks stage, they use the same model: the convex polygons.
Like for the road networks, the partition of a block into lots is generally done according to a predefined pattern. This resemblance of description does not extend into the generation methods: the small dimensions of a block do not require the use of the L-System. The used methods tend to be based on a geometric partition or on an instantiation of a pattern.

Geometric partition In [PM01], this stage is handled by a recursive process that divides the bigger lot in the middle of its bigger side. This process ends when the surface of the biggest lot is lower than a predefined threshold. Then a post-filtering on the lots

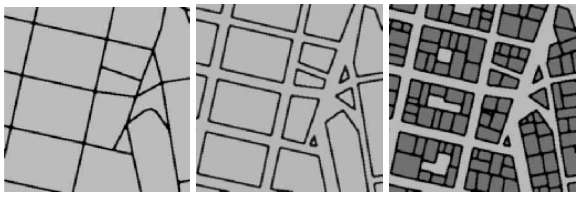


Figure 3: Roads, blocks and lots [PM01]

delete the too small lots as well as those that do not have a direct access to a road.

Pattern guided partition In [LH97] the generation begins from a discretized surface corresponding to the block. A point of this grid is randomly chosen, then the process searches for another point in order to create a new segment that will be compatible with the chosen pattern. This guided search for the extremity of the segment reduces efficently the search space.

Discussion Both methods presented here are extremely different. The first one is effective but creates little varied results. The second offers a big variety of results but uses a generation based on an enumerative process, that leads to restricting the domain of application. According to the degree of realism and efficiency wanted either can be used.

2.5 Buildings

This stage defines the exterior of the buildings. The variables to instantiate during this stage are the shape, height and orientation of the building as well as the distance to the lot's contour.

Shape grammar The LaHave House project studies the industrial architectural design in [RMS96]. This tool is intended to help the designer to conceive aesthetic and functional houses that will be buildable.
From shape grammars, a library of architectural elements is generated then used to visualize working/assembly drawings and 3D scenes. A model is made up of the geometric description and the configuration of the house levels.

Declarative modeling BatiMan ([Cha98]) is a declarative modeller of a small number of buildings. It is based on a learning process stem from the Artificial Intelligence. It classifies the solutions to propose to the user a restricted number of characteristic solutions. Batiman relies on a constraint solver based on the CSP approach on finite domains for the generation phase. The number of parameters defining a building can go to about twenty with a small number of values for each variable.

Split grammars Work described in [WWSR03] derived buildings using split grammars. Theses grammars are parametric set grammars based on the concept of shape. The system uses a second kind of grammar, the *control grammar*, in order to control the propagation of the split grammar's attributes. Using theses two types of grammars, the system can create realistic buildings from high-level or precise descriptions. Nevertheless, if the user wants to modify the rules of the grammar, he must be familiar with the split grammar approach.

Geometric modeling AGETIM [LLGC+05] is an integrated software tool suite that enables 3D environment generation with infrared, electromagnetic and acoustic characterization. The GenVillage module allows the system to create buildings from their base (created in an automatic or manual way) by using predefined templates. Buildings which are created in this way have variable heights and roof forms, and are automatically adapted to their environment (style, orientation and textures).

Discussion The methods described for this stage answer different needs. The construction of the library required for the use of the shape grammar takes a lot of time but allows the user to generate realistic results. The declarative modelling is less realistic but quicker than the first one, whereas the method based on geometrical modelling allows an easy integration according to the building's environment.

2.6 Building plan

This stage was studied as a 2D architectural problem that corresponds to a partition process. Work presented in [Mac91] introduces the building design as a process of constraint satisfaction. This work relies on the framework of AI and Operational Research. The representation of the spatial knowledge in architecture was more particularly studied to be able to define the spatial constraints. More precisely, he studied the link between the symbolical (topological) and the numerical spatial knowledges that had not been properly studied before.
Theses knowledge were used on objects which were modeled using 3D boxes called the *Manhattan boxes*. This representation was selected because of its adequacy with the architectural problems: a building can be seen as a compound of simple shapes. These boxes and the associated operations form the *Manhattan algebra*. A constraint network represents the problem, that is solved by a static propagation mechanism of constraints using heuristics for the instantiation order of the objects.
An extension was developed [Cha93] in order to take into account the geometric specificity of a layout problem, using a new filtering method *semi-geometrical arc-coherency*. Work presented in [MY01] proposes an improvement of previous work by the use of dynamic space ordering heuristics and a topological characterization of the solution space.

2.7 Furnished buildings

It is possible to completely define the instantiation of an object within a 3D model by its geometrical description and its variables of orientation and position.

Figure 4: Scene generated by a tabu search based declarative modeler [LLRG04]

The number of constraints to solve grows exponentially with the number of objects.

CSP approach During the last decade, several constraint solvers based on the CSP approach have been developed to resolve the interior space layout problem. We can distinguish two classes: those limited to an isothetic layout [BP99, LRG03] and those that allow the user to place objects with any orientation [RP02, LR03].

Metaheuristics Stochastic methods (or heuristics) were first studied within the framework of combinatorial optimization in operational research and AI. During the last fifteen years, the stochastic methods have notably progressed with the emergence of a new generation of powerful and general tools called *metaheuristic methods* [Glo86].
Two different classes of metaheuristics were studied for the space layout problem: first the genetic algorithms were studied in [SRGL03, VMC$^+$02], then [LLRG04] studied the interest of a local search metaheuristic, the tabu search.

3. CONCLUSION

The work presented in this state of the art shows the variety of the existing approaches for the city generation. CityEngine, allows a city to be modeled from scratch to exteriors (stages 1 to 5), but no current tool takes into account the seven described stages. Moreover, it is difficult to compare the methods because they result from various research domains.
Some of the seven stages presented in this article have not been extensively studied. The block generation (as well as the lot generation) have been less studied. We have begun to work on this stage in order to evaluate the interest of geometric methods such as Voronoï diagrams and the straight skeleton approach.
It seems possible to create a unified tool that should be able to process all the stages. But this will probably be a difficult task.

4. REFERENCES

[BP99] Bonnefoi P.-F. and Plemenos D. Object oriented constraint satisfaction for hierarchical declarative scene modeling. WSCG'99, 1999.

[Cha93] Charman P. Solving space planning problems using constraint technology. Technical report, Institute of Cybernetics, Estonian academy of sciences, 1993.

[Cha98] Champciaux L. Gestion des contraintes géométriques pour l'aide à l'aménagement urbain. PhD thesis, Ecole des Mines de Nantes, 1998.

[Glo86] Glover F. Future paths for integer programming and links to artificial intelligence. Computers and Operations Research, 13:533-549, 1986.

[LH97] Liège S. and Hégron G. An incremental declarative modelling applied to urban layout and design. WSCG'97, 1997.

[LLGC$^+$05] Latger J., Le Goff A., Champseix N., Cathala T., and Larive M. Automatic 3d virtual scenes modeling for multi sensors simulation. to appear. In Proceedings of SPIE Defense and Security Symposium, 2005.

[LLRG04] Larive M., Le Roux O., and Gaildrat V. Using meta-heuristics for constraint-based 3d objects layout. 3IA'04, 2004.

[LR03] Le Roux O. Modélisation déclarative d'environnements virtuels : contribution à l'étude des techniques de génération par contraintes. PhD thesis, Université Paul Sabatier, 2003.

[LRG03] Le Roux O. and Gaildrat V. Constraint-based 3d isothetic object layout for declarative scene modeling. CISST'03, 2003.

[Mac91] Maculet R. Archipel : intelligence artificielle et conception assistée par ordinateur en architecture. PhD thesis, Université Paris 6, 1991.

[MY01] Medjoub B. and Yannou B. Dynamic space ordering at a topological level in space planning. Artificial Intelligence in engineering, 15:47-60, 2001.

[PM01] Parish Y. and Müller P. Procedural modeling of cities. ACM SIGGRAPH, 2001.

[RMS96] RauChaplin A., MacKayLyons B., and Spierenburg P. The lahave house project: Towards an automated architectural design service. Cadex'96, pp. 24-31, 1996.

[RP02] Ruchaud W. and Plemenos D. Multiformes: a declarative modeller as a 3d scene sketching tool. ICCVG'2002, 2002.

[SRGL03] Sanchez S., Roux O. L., Gaildrat V., and Luga H. Constraint-based 3d-object layout using a genetic algorithm. 3IA'03, 2003.

[VMC$^+$02] Vassilas N., Miaoulis G., Chronopoulos D., Konstantinidis E., Ravani I., and Plemenos D. Multicad-ga : A system for the design of 3d forms based on genetic algorithms and human evaluation. SETN, pp. 203-214, 2002.

[WWSR03] Wonka P., Wimmer M., Sillion F., and Ribarsky W. Instant architecture. ACM Transactions on Graphics, 4(22):669–677, july 2003. Proceedings of ACM SIGGRAPH 2003.

A Subdivision Scheme Based on Vertex Normals for Triangular Patches

Zhihong Mao
Dept. of Computer Science and Engineering
Shanghai Jiao Tong University
200030, Shanghai, PR China
mzh_yu@sjtu.edu.cn

Lizhuang Ma
Dept. of Computer Science and Engineering,
Shanghai Jiao Tong University
200030, Shanghai, PR China
ma-lz@cs.sjtu.edu.cn

Mingxi Zhao
Dept. of Computer Science and Engineering,
Shanghai Jiao Tong University
200030, Shanghai, PR China
zhaomx@cs.sjtu.edu.cn

ABSTRACT

In order to achieve a smooth surface while rendering a triangle-based mesh, we need to eliminate the mismatch between the smoothness of the shading and the non-smoothness of the geometry that is particularly visible at silhouettes. To eliminate these artifacts, we substitute the geometry of a cubic triangular Bézier curved patch for a triangular flat geometry. Meanwhile a subdivision algorithm is proposed by using the degree elevation to approximate the triangular cubic Bézier patch only with little cost. The proposed algorithm can be processed without further knowledge of neighboring triangles, and can be operated as a process prior to a traditional rendering pipeline and required little change to existing authoring tools.

Keywords

Smoothing, geometric modeling, subdivision, surface, triangular Bézier patch.

Figure 1. (a) shading of initial model, (b) shading of our scheme

1. INTRODUCTION

Interpolated shading, such as Gouraud shading and Phong shading, has been used in computer graphics widely because of its effectiveness and simplicity. But in these two methods there is an inherent mismatch between the smoothness of the shading and the non-smoothness of the geometry that is particularly visible at silhouettes (see Figure 1a). In this paper, we study the problem for removing straight silhouette edges and develop a subdivision algorithm based on vertex normals for triangular patches (see Figure 2), which can be taken as an inexpensive means to improve the visual quality (see Figure 1b).

The paper is organized as follows. In section 2 we review the related work on this topic. Section 3 discusses how to construct a triangular cubic Bézier control net. Section 4 analyzes the degree elevation algorithm and adapts it to our subdivision scheme. Section 5 summarizes our work.

WSCG SHORT papers ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. RELATED WORK

Since Doo and Sabin [Doo78a] gave a corner-cutting algorithm to generate bi-quadratic uniform B-spline patches, Subdivision schemes defining smooth surfaces have been widely studied in the field of computer graphics [Cat78a, Loo87a, Aym99a, Jör98a, Den00a]. The basic idea behind subdivision is to create a smooth limit function by infinite refinement of an initial piecewise linear function. However, they rely on the availability of vertices of adjacent patches rather than vertices and normal vectors of the patch.

Recently local subdivision over a patch to eliminate the artifacts at silhouettes has received much attention. Alex Vlachos and Overveld [Vla01a, van97a] introduced curved point-normal (PN) triangle that is re-triangulated into a number of sub-triangles to replace the flat triangle. Stefan Karbacher [Ste00a, Ste98a, Ste98b] present a non-linear subdivision scheme for the refinement of triangle meshes that generates smooth surfaces with minimum curvature variations. Different to these schemes our scheme takes the information of each subdivision step into account. Our method is more simple and efficient. The algorithm is described in details in the following sections.

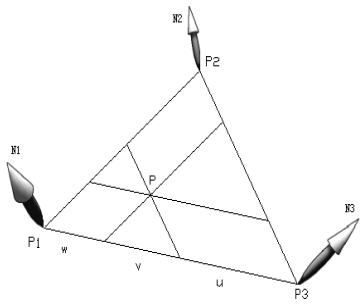


Figure 2. Input Triangle

3. CONTROL NET OF THE PATCHE

The bi-cubic triangular Bézier patch can be written in terms of Bernstein polynomials:

$$P(u, v, w) = \sum_{i=0}^{3} \sum_{j=0}^{3-i} b_{i,j,k} B^{3}_{i,j,k}(u, v, w);$$
$$0 \le u, v, w \le 1; \; u + v + w = 1;$$
$$i + j + k = 3; \quad i, j, k \ge 0 \qquad (3.1)$$

$$B^{3}_{i,j,k}(u, v, w) = \frac{3!}{i!\, j!\, k!} u^{i} v^{j} w^{k} \qquad (3.2)$$

For the patch, its end tangents must be perpendicular to the given normal vectors that attached to the three end points. We group the $b_{i,j,k}$ together as:

End vertices: b_{300}, b_{030}, $b_{003;}$ Center vertex: $b_{111;}$

Boundary vertices: b_{210}, b_{120}, b_{021}, b_{012}, b_{102}, $b_{201.}$

3.1 Construct the boundary curves

Here we construct a cubic curve leading from P_i to P_j that is normal to N_i at P_i and to N_j at P_j. In Bezier form, we already have $b_{3i} = P_i$ and $b_{3i+3} = P_j$, We still need to find the two inner Bézier points b_{3i+1} and $b_{3i+2.}$ For each inner Bézier point, we have a one-parameter family of solutions: We only have to ensure that each triple b_{3i-1}, b_{3i}, b_{3i+1} is collinear on the tangent directors I_i at b_{3i}:

$$b_{3i+1} = b_{3i} + \alpha_i I_i;$$
$$b_{3i-1} = b_{3i} - \beta_{i-1} I_i \qquad (3.3)$$

Where the tangent directors I_i have been normalized to be of unit length: $\| I_i \| = 1$.

So that the problem become to find reasonable values for α_i and β_i. Values for α_i and β_i that are too small cause the curve to have a corner at b_{3i}, while values that are too large can create loops. An optimal choice must depend on the desired application. Here, we set

$$\alpha_i = \beta_i = 0.33 \| \Delta X_i \|,$$
$$\| \Delta X_i \| = \| b_{3i+3} - b_{3i} \|$$

So using the upper method, we can decide the six boundary vertices: b_{210}, b_{120}, b_{021}, b_{012}, b_{102}, $b_{201.}$ Three end vertices defined as follows:

$b_{300} = P_1$, $b_{030} = P_2$, $b_{003} = P_{3.}$

3.2 Control net of triangular patches

The center vertex b_{111} corresponds to the center of the triangle and is decided by six boundary vertices and three end vertices. We move the center vertex from the position V to the average of the six boundary vertices and continue its motion in the same direction for 0.5 the distance already traveled [Vla01a, Far88a].

$E = (b_{210}+b_{120}+b_{021}+b_{012}+b_{102}+b_{201})/6$

$V = (P_1+P_2+P_3)/3$, $b_{111} = E + (E-V)/2.$

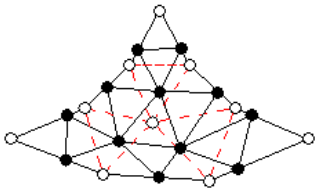


Figure 3. Bézier patch (from cubic to the fourth order)

4. SUBDIVISION SCHEMES

4.1 Degree Elevation

It is possible to represent a degree n triangular Bézier patch in the form of degree n+1:

$$P(u,v,w) = \sum_{i=0}^{n} \sum_{j=0}^{n-i} b_{i,j,k} B^{n}_{i,j,k}(u,v,w)$$

$$= \sum_{i=0}^{n+1} \sum_{j=0}^{n+1-i} b^*_{i,j,k} B^{n+1}_{i,j,k}(u,v,w) \quad (4.1)$$

For a triangular Bézier patch, we have u + v + w = 1 (see 3.1). We multiply the left hand of (4.1) by (u + v + w):

$$\sum_{i=0}^{n} \sum_{j=0}^{n-i} [b_{i,j,k} B^n_{i,j,k}(u,v,w)u + b_{i,j,k}$$

$$B^n_{i,j,k}(u,v,w)v + b_{i,j,k} B^n_{i,j,k}(u,v,w)w]$$

$$= \sum_{i=0}^{n+1} \sum_{j=0}^{n+1-i} b^*_{i,j,k} B^{n+1}_{i,j,k}(u,v,w) \quad (4.2)$$

Use (3.2) to substitute (4.2) and compare coefficients of the terms having the same degree, we can get:

$$b^*_{i,j,k} = \frac{1}{n+1}(ib_{i-1,j,k} + jb_{i,j-1,k} + kb_{i,j,k-1}) \quad (4.3)$$

4.2 Subdivision algorithm

Degree elevation process generates a sequence of control nets that have a cubic triangular Bézier surface as the limit after a sequence of successive refinements (see Figure 3). From (4.3) we can find the following three properties:

1. For three vertices of the input flat triangle, two of indices i, j, k equals 0. So we have $b_{n00} = P_1$, $b_{0n0} = P_2$, $b_{00n} = P_3$.
2. For boundary vertices, one of indices i, j, k equals 0, the boundary vertices are only decided by the two edge end points. Using this method, we can get C^0 continuity at the boundary of two surfaces and no hole occurs between two adjacent triangles.
3. Vertices inside the triangle are decided only by three end points.

Finally we also use the degree elevation equation to calculate the normal vector of each vertex for shading. Figure 4 illustrates the choice of normal. The equation defines as follows.

$$N^*_{i,j,k} = \frac{1}{n+1}(iN_{i-1,j,k} + jN_{i,j-1,k} + kN_{i,j,k-1}) \quad (4.4)$$

Where $N_{1,0,0} = N_1$, $N_{0,1,0} = N_2$, $N_{0,0,1} = N_3$

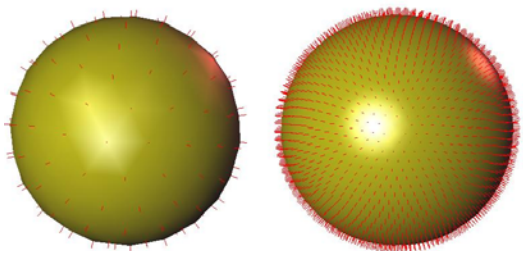


Figure 4: normal vector attached on each vertex (a) initial mesh; (b) after degree elevation

5. SUMMARY AND CONCLUSION

A method has been given for removing the artifacts at silhouettes, based on the vertex normals. Figure 5, 6 and 7 demonstrate that this algorithm can produce a visually smooth effect from an initial coarse mesh model. To some extent, the method eliminates the mismatch between the smoothness of the shading and the non-smoothness of the geometry. The proposed algorithm makes good use of the theory of degree elevation and simplifies the computation. Moreover it can be processed without further knowledge of neighbor triangles. Several questions still remain to be investigated. For example, these surfaces are still not C^1, the theoretical background of computation of the triangular cubic Bézier control net is still not satisfying. Adaptive subdivision is not discussed in this paper.

6. ACKNOWLEDGMENTS

We gratefully thank Jörg Peters for his mesh model data. The work is partially supported by national natural science foundation of China (Grand No. 60373070 and No. 60173035) and 863 High Tech Project of China (Grant No. 2003AA411310).

7. REFERENCES

[Doo78a] D. Doo and M. Sabin. Behavior of recursive division surfaces near extraordinary points. Computer-Aided Design 10: 356-360, Sept, 1978.

[Cat78a] CatmullL, E. and Clark, J. Recursively generated B-spline surfaces on arbitrary topological meshes. Computer-Aided Design 10:350-355, Sept, 1978.

[Loo87a] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, August 1987.

[Aym99a] Ayman Habib and Joe Warren. Edge and vertex insertion for a class of C1 subdivision surfaces. Computer Aided Geometric Design, 16(4): 223-247,1999

[Jör98a] Jörg Peters and Ulrich Reif. Analysis of algorithms generalizing B-spline subdivision. SIAM Journal on Numerical Analysis, 35(2): 728-748(electronic), 1998.

[Den00a] Denis Zorin. Smoothness of subdivision on irregular meshes. Constructive Approximation, 16(3), 2000.

[Vla01a] Vlachos Alex, Jörg Peters, Chas Boyd and Jason L. Mitchell. Curved PN Triangles. ACM Symposium on Interactive 3D Graphics :159-166, 2001.

[Ste00a] Stefan Karbacher, Stefan Seeger and Gerd Häusler. A Non-linear Subdivision scheme for Triangle Meshes. Proc. Of Vision, Modeling and Visualization: 163-170, 2000.

[Ste98a] Stefan Karbacher, Gerd Häusler and H. Schönfeld. Reverse engineering using optical 3D sensors. Three-Dimensional image capture and applications, Proceedings of SPIE(3313): 115-125, 1998.
[Ste98b] Stefan Karbacher, Gerd Häusler. A new approach for modeling and smoothing of SPIE(3313):168-177, 1998.
[van97a] C.W.A.M. van Overveld and b.Wyvill. An algorithm for polygon subdivision based on vertex normals. In Computer Graphics International : 3–12, 1997.
[Far88a] G. Farin. Curves and Surfaces for Computer Aided Geometric Design. Academic Press, 1988.

Figure 5,6,7,. (from left to right) (a) Initial mesh; (b) Mesh with our method; (c) shading model (initial mesh); (d) Shading with our method.

Constructing the layer representation of polygons in parallel

F. Martínez, A.J. Rueda and F.R. Feito
Depto. de Informática (Universidad de Jaén)
Campus Las Lagunillas
23071, Jaén, Spain
{fmartin,ajrueda,ffeito}@ujaen.es

ABSTRACT

For several years we have developed a scheme for representing polygons and 3D polyhedra by means of layers of triangles and tetrahedra, together with effective algorithms for basic geometric operations on the represented objects. In this paper we explore the optimization of some of these algorithms by parallel processing techniques.

Keywords
Graphic object representations, parallel geometric algorithms

1. INTRODUCTION

In previous work [Rue02a, Fei99a] we have developed a method for representing polygons based on layers of triangles, showing several interesting properties: it is valid for any kind of polygon, simple, has little space needs, and makes easy the effective implementation of several operations, as the point-in-polygon inclusion test [Fei95a, Fei97a] or Boolean operations on polygons [Riv00a].

During the development of our work we have had some problems with the algorithms used for constructing the polygon representation. The most efficient, based on a radial line sweeping, needs $O(nlogn)$ time, where *n* is the number of vertices of the polygon. However, this algorithm is not extensible to 3D, so that we are more interested in a simpler, 3D extensible algorithm that requires $O(n^2)$ time and space.

Realistic scenes have a great number of polygons containing a great number of vertices. So, in order to speed up the representation of scenes we must optimize the algorithm used for constructing polygons.

Conference proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

For this reason we have decided to parallelize the $O(n^2)$ constructing algorithm, and this experience is described in the rest of the paper.

The remainder of the paper is structured as follows. Section 2 briefly describes the representation of polygons by layers. Section 3 explains the sequential, $O(n^2)$ algorithm used for constructing the representation. Section 4 shows how to parallelize this algorithm. Section 5 compares execution times of both algorithms. Finally, Section 6 brings conclusions and states the future directions of our research.

2. THE LAYER REPRESENTATION OF POLYGONS

This section briefly describes the representation of polygons by layers, so that the algorithm used for its construction, explained in the next section, can be understood. See [Rue02a, Fei99a] for a detailed description of the contents of this section.

Given a polygon, it is possible to define a set of triangles between an arbitrary point, named *origin*, and each edge of the polygon. These triangles are called *origin triangles* because one of their vertices coincides with the origin. An origin triangle has two *origin edges* whose vertices are determined by the origin and a vertex of the polygon, and a *non-origin edge* which belongs to the polygon. Figure 1 shows a polygon and their corresponding origin triangles.

The layer representation is based on the concepts of: *subordination relation*, *subordination chain*, *index*, and *layer*.

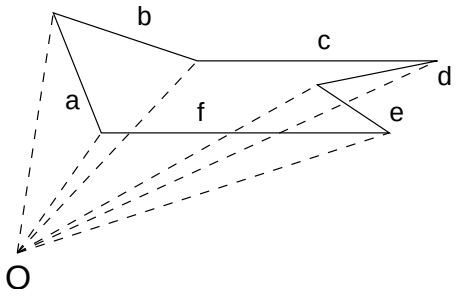


Figure 1: A polygon and its origin triangles.

Definition 1. Let *t* and *s* be two origin triangles of a polygon. We say that *t* is subordinate to *s* ($t \triangleleft s$) if and only if $t = s$ or, in the case that $t \neq s$, exists a point belonging to the unidimensional interior of the non-origin edge of *t* that also belongs to the interior of the triangle *s*. In Figure 1 *a* is subordinate to *b* ($a \triangleleft b$) because the interior of *b* clearly contains a big portion of the non-origin edge of *a*. Triangles *b* and *c* are not subordinate to any other one.

Definition 2. A *subordination chain* is defined as an ordered sequence of triangles $S = \{s_0 s_1 \dots s_n\}$ where $s_i \triangleleft s_{i+1} \forall i : 0 \leq i < n$. In Figure 1 all non-trivial subordination chains are $\{a, b\}$, $\{f, e\}$, $\{f, e, d\}$, $\{f, e, d, c\}$, $\{f, b\}$, $\{f, c\}$, $\{f, d\}$, $\{e, d\}$, $\{e, d, c\}$, $\{e, c\}$ and $\{d, c\}$.

Definition 3. We define the *index* of an origin triangle *s* ($ind(s)$) as the length of the longest subordination chain of the set of ordered triangles determined by a polygon *P*, starting with triangle *s*. Formally, $ind(s) = max\{|S_i|\}$ where $S_i = \{s_0 s_1 \dots s_n\}$ is any valid subordination chain as defined previously verifying $s = s_0$. In Figure 1 $ind(a) = 2$, $ind(b) = 1$, $ind(c) = 1$, $ind(d) = 2$, $ind(e) = 3$ and $ind(f) = 4$

Triangles can be sorted in several *layers* according to their index. This is the *layer representation* or *L-REP* of a polygon, as we see in the next definition.

Definition 4. We define the *layer representation* or *L-REP* of a polygon *P* as the set of layers containing the triangles with the same index, that is $R = \{L_1 L_2 \dots L_l\}$ where $L_i = \{s_j : ind(s_j) = i\}$ is the layer of origin triangles s_j with index *i*, and *l* is the maximum index of the origin triangles generated by the polygon *P*.

Figure 2 illustrates the representation of a polygon by its L-REP.

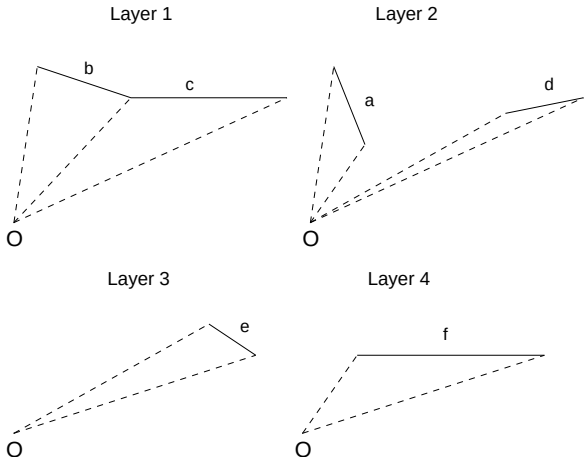


Figure 2: A polygon and its L-REP.

3. SEQUENTIAL CONSTRUCTION OF THE L-REP

In this section we describe the sequential algorithm used for constructing the layer representation of a polygon. Its pseudocode is shown below.

```
int n = p.n_edges(); // # edges
bool M[n][n];      // sub. matrix
int sc[n];         // sub. counters

// Stage 1: Build M and sc
for (int i = 0; i < n; i++) {
  sc[i] = 0;
  for (int j = 0; j < n; j++) {
    M[i][j] = p.edge(i).sub_to
              (p.edge(j));
    sc[i] += M[i][j] ? 1 : 0;
  }
}

// Stage 2: Build layers
int ntp = n;  // # triangles processed
int l = 0;    // current layer
vector<int> sct(sc); // copy of sc
while (ntp > 0) {
  l++;
  p.lrep.add_layer(l);
  for (int i = 0; i < n; i++) {
    if (sc[i] == 0) {
      p.lrep.add_triangle(l, p.edge(i));
      for (int j = 0; j < n; j++) {
        if (M[j][i]) {
          sct[j]--;
        }
      }
      sct[i] = -1;  // inserted
      ntp--;
    }
  }
  sc = sct;
}
```

The L-REP of a polygon *p* with *n* edges is constructed in two stages. In the first stage two data structures are built: the subordination matrix (*M*) and the subordination counters (*sc*). *M* is a nxn boolean matrix that stores at position (i, j) if $s_i \triangleleft s_j$, where s_x refers to the origin triangle associated to the edge *x* of *p*. The *sc* vector stores at position *i* the number of triangles to which s_i is subordinate — i.e., the origin triangle associated to the edge *i* of *p*.

From the data structures built in the first stage, *M* and *sc*, is easy to calculate the layers of a polygon taking into account the following remark: after stage 1, triangles whose subordination counter are 0 belong to layer 1. If, for every layer 1 triangle s_i, the subordination counters of the triangles s_j such that $s_j \triangleleft s_i$ are decremented by 1, the new triangles whose subordination counter are 0 belong to layer 2. The same reasoning can be applied to compute the layer 3 triangles from layer 2 triangles, and so on, all the layers of a polygon.

4. PARALLEL CONSTRUCTION OF THE L-REP

We will apply the *data domain decomposition* scheme to parallelize the sequential algorithm described in the previous section. Following this scheme, a data domain to be processed is divided into several disjoint "chunks", and every "chunk" is allocated to a different process that runs an algorithm on it. If, as in our case, all processes run the same algorithm we have a SPMD (*Single Program Multiple Data*) application.

In our parallelization the data domain consists of the polygon edges, — i.e., their associated origin triangles. A different subset of origin triangles is allocated to every process. A process is responsible for computing the layer associated to every triangle of his subset. In order to do this, processes run the following algorithm.

```
Broadcast(p, root);
int pid = get_rank();  // process id.
int np  = get_size();  // # processes
int n   = p.n_edges(); // # edges
int sr;                // starting row
int ne;   // # edges allocated
calculate_chunk(pid, np, n, &sr, &ne);
bool M[ne][n];         // sub. matrix
int sc[ne];            // sub. counters
// Stage 1: Build M and sc
for (int i = 0; i < ne; i++) {
  sc[i] = 0;
  for (int j = 0; j < n; j++) {
    M[i][j] = p.edge(sr+i).sub_to
              (p.edge(j));
    sc[i] += M[i][j] ? 1 : 0;
  }
}

// Stage 2: Build layers
int ntp = n;  // # triangles processed
int l = 0;      // current layer
vector<int> cl; // current local layer
vector<int> cg; // current global layer
while (ntp > 0) {
  l++; cl.resize(0); cg.resize(0);
  p.lrep.add_layer(l);
  for (int i = 0; i < ne; i++) {
    if (sc[i] == 0) {
      cl.push_back(sr+i);
      sc[i] = -1;  // inserted
    }
  }
  AllGather(cl, cg);
  for (int i = 0; i < cg.size(); i++) {
    p.lrep.add_triangle(l, p.edge(cg[i]));
    for (int j = 0; j < ne; j++) {
      if (M[j][cg[i]]) {
        sc[j]--;
      }
    }
  }
  ntp -= cg.size();
}
```

At first, every process receives a copy of the polygon and calculates the subset of triangles allocated to it (from sr to $sr + ne - 1$). After this, the process computes the subordination matrix and the subordination counters of his subset of triangles. This computation is done without interacting with other processes. Next, the second stage starts. In this stage the layers of the polygon are calculated. Every process computes locally the triangles from his subset belonging to the current layer. Next, processes gather the current layer triangles found in all the processes by an *Allgather* collective operation. So that processes can decrement local subordination counters.

Before ending this section we want to do the following remarks:

- The *Broadcast*, *get_rank*, *get_size*, *Allgather* and *calculate_chunk* functions used in the algorithm are similar to functions defined in the MPI specification [Mpi94a].
- *Broadcast* and *Allgather* are synchronous collective operations. So, the parallel algorithm progresses at the pace of the slowest process — the others must wait for it. However, this is not a problem if the target hardware, as in our case, is a homogeneous cluster exclusively dedicated to the application execution. In fact, taking into account our target hardware, self-scheduling schemes as master-slave perform worse than our proposed solution.

Edges (number of processes)	Time	Speedup
1000 (1)	0,62	–
1000 (2)	0,51	1,22
1000 (4)	0,22	2,82
1000 (8)	0,18	3,44
5000 (1)	21,78	–
5000 (2)	9,12	2,39
5000 (4)	4,46	4,88
5000 (8)	2,43	8,96
10000 (1)	85,82	–
10000 (2)	42,32	2,03
10000 (4)	20,93	4,1
10000 (8)	9,91	8,66

Table 1. Execution results (in seconds).

- At algorithm termination the L-REP of the polygon is available to all the processes. This can be useful for future parallel operations on the polygon.

5. EXPERIMENTATION

In this section we describe the results from implementing the previous section algorithm using MPI. The target hardware consists of 8 AMD dual processors at 2.1 GHz connected by a 1 Gigabit/s Ethernet.

Table 1 shows the results of some sample executions. The left column contains the execution parameters, i.e., the number of edges of the polygon and the number of processes — or processors — used in the parallelization. If the number of processes is 1, the sequential algorithm is executed. The others columns show the execution time and the speedup obtained — calculated as the sequential execution time divided by the parallel execution time.

As can be seen the results are excellent. Even for the polygon with 1000 edges, which sequential execution only takes 0,62 seconds, good performance in terms of speedup has been obtained. For the polygons with 5000 and 10000 edges a super speedup is obtained — greater than n if n processors are used. This fact is normal when parallelizing algorithms with great memory needs, as in our case due to the subordination matrix, because parallel distributed hardware not only offers more CPU cycles, but also more cache and main memory.

Lastly to note some information not shown in Table 1:

- The time needed to broadcast the polygons is negligible compared to the total execution time.
- The speedup obtained in the two stages of the parallel algorithm is similar.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have described the parallelization of the algorithm used for constructing the L-REP of a polygon. As described in the previous section, the execution results are excellent, specially when representing big polygons. Big polygons can be frequently found, as in complex fonts or GIS, so the parallelization is quite useful. Furthermore, this algorithm is extensible to 3D where the size of the objects can be very big.

In the future we would like to apply parallel processing techniques to some applications of the L-REP such as Boolean operations or plain location.

7. ACKNOWLEDGEMENTS

This work has been partially granted by the Ministry of Science and Technology of Spain and the European Union by means of the ERDF funds, under the research projects TIC-2001-2099-C03-03 and TIN-2004-06326-C03-03.

8. REFERENCES

[Fei95a] Feito, F.R.; Torres J.C., “Orientation, simplicity and inclusion test for planar polygons”, Computer & Graphics 19, pp. 595–600, 1995.

[Fei97a] Feito, F.R.; Torres J.C., “Inclusion test for general polyhedra”, Computer & Graphics 21, pp. 64–77, 1997.

[Fei99a] Feito, F.R.; Rivero M. y Rueda, A.J., “Boolean representation for general planar polygons”, Proceedings of the WSCG’99, pp. 87–92, 1999.

[Mpi94a] Message Passing Interface Forum, “MPI: A Message-Passing interface standard”, International Journal of Supercomputer Applications, 8(3/4), pp. 165–414, 1994.

[Riv00a] Rivero, M.L.; Feito, F.R., “Boolean operations on general planar polygons”, Computer & Graphics 24, pp. 881–896, 2000.

[Rue02a] Rueda, A.J.; Feito, F.R. y Rivero, M., “A triangle-based representation for polygons and its applications”, Computer & Graphics 26, pp. 805–814, 2002.

Attributed Collage Grammars: A Rule-Based Modeling Framework

Carolina von Totth
University of Bremen, Department of Computer Science
Linzer Strasse 9A, 28359 Bremen, Germany
caro@tzi.de

ABSTRACT

The strength of procedural modeling methods in general and grammar-based modeling in particular lies in their ability to generate visually complex objects and scenes from a comparatively small and simple description. This paper presents attributed collage grammars, a general and versatile modeling method for the generation of scenes. Tree grammars are employed to generate a syntactical description of objects in the form of trees, which are then assigned an actual geometric meaning by an algebra. Complex scenes and objects with a level of detail that would be tedious or impossible to reproduce manually can be constructed from a number of user-defined components which are replicated or transformed by manipulating their attributes through the iterative application of rules.

Keywords

procedural modeling, grammar-based modeling, Greek architecture, tree grammar, algebra, data amplification

1. INTRODUCTION

The construction of any convincing virtual environment, be it photo-realistic or not, is always a difficult task. The reason lies in the complexity of natural phenomena, and although all manner of modeling methods have been devised in order to handle such visual complexity (e.g. particle systems for simulating dynamic processes [RB85], L-systems with an extensive application in plant generation [PL96, PMKL01] as well as many procedural algorithms with their roots in fractal geometry [EMP+98], to name but a few), there is no general solution that is uniquely suitable to all applications.

So far grammar-based methods, L-systems in particular, have been primarily used in the modeling of plants and more recently modern cities [PM01].

This paper presents a new method that is not bound to a specific modeling purpose. Rather, it provides a flexible grammar-based scene generation framework that can be customized to suit multiple modeling goals, from architecture to jewelry design. Attributed collage grammars build upon collage grammars as presented by Drewes and Kreowski in [DK99], using and extending the tree-based version specified in [Dre00] and [Dre04].

In the diploma thesis[1] by Thomas Meyer and the author a restricted implementation has been used to model languages of buildings based on the principles governing ancient Greek architecture (Figure 1).

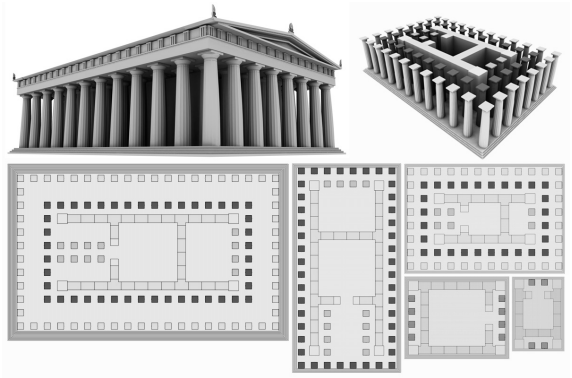
The paper is structured as follows: in Section 2 an overview of attributed collage grammars is given. Section 3 gives solutions to two problems which were previously not solvable using context-free collage grammars; one of these is illustrated by means of an example. Section 4 concludes the paper with an outlook on future work.

SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. ATTRIBUTED COLLAGE GRAMMARS

Most grammar-based methods of image or geometry generation, like chain code grammars, turtle grammars and L-systems, originated from the string-rewriting formalisms of formal languages. Such grammars generate strings that require the additional step of a pictorial interpretation to become images. In their original form [DK99], collage grammars were an attempt to work spatially from the beginning, bypassing the

[1] http://www.tzi.de/~ariel/GreekArchitectureWithCollages.pdf



(a) Sample layouts of temples generated with a single attributed collage grammar

(b) A peripteral and a dipteral temple generated with the same grammar

Figure 1: Models of Greek temples generated with attributed collage grammars

string rewriting mechanism by replacing the nonterminal symbols of formal languages with labelled spatial placeholders, called *hyperedges*, and terminal components with actual point sets in $\mathbb{R}^d$, called *parts*. A clustering of hyperedges and parts is called a *collage* and provides a spatial counterpart to the concatenation operation used to group line drawings.

The rules of a collage grammar specify how hyperedges and parts can be altered through the application of affine transformations, and are applied iteratively starting from an initial collage. For collage grammars in $\mathbb{R}^2$ this already allows generating well-known fractals like the Barnsley fern or the Koch snowflake curve. In fact the image generating power of collage grammars in this form already exceeds that of iterated function systems.

The method presented here builds upon the LEGO™-like idea of collage grammars, using it as a basis for a much more general approach. Attributed collage grammars are described using a tree-based formalism, once again separating the syntax of the generated structures from their semantics; this is mainly done in order to achieve better modularity.

In tree-based scene generation, objects or scenes are represented as trees over scene components (leaf nodes) and operations on these (internal nodes). Such trees are generated as purely syntactic constructs over a signature by a tree grammar, and evaluated by an algebra which provides the nodes of the tree with the proper semantics. This approach is highly modular, since different algebras can be paired with different types of grammars.

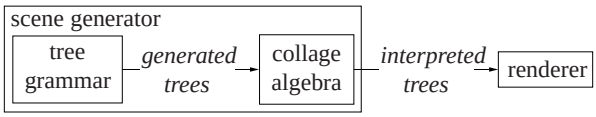


Figure 2: The scene generator principle

Attributed collage grammars are *scene generators* (Figures 2 and 3), consisting of a *tree grammar* and an *attributed collage algebra*. Therefore, in trees produced by an attributed collage grammar, the leaf nodes are the collages, which have retained their meaning as a grouping of parts, while the operations serve as a grouping mechanism for internal transformations.

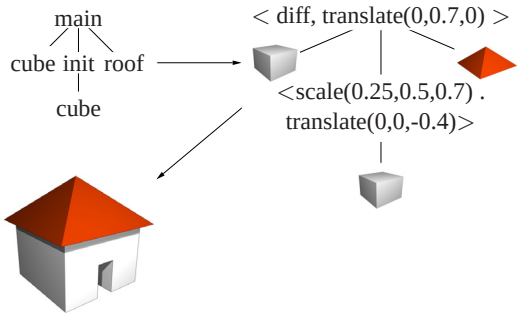


Figure 3: Scene generation: an example

In the small example displayed in Figure 3 a very simple house is built the following way:

- First a syntax tree is generated by some grammar: its root is an operation called `main` which is applied to subtrees `cube`, `init[cube]` and `roof`. At this stage we have no idea what `main` does — or what `init`, `cube` and `roof` are, for that matter.
- In the next step the tree has been interpreted by some attributed collage algebra. The middle cube is distorted by a scale transformation and moved a little, and then it is subtracted from the first cube using `diff`, which is a Boolean difference transformation that takes two arguments. Finally the roof is moved up a little: the result is a small cartoony house.

In attributed collage grammars, the parts are extended to include not only simple point sets, but also any other type of scene component available, among them splines, particle emitters and light sources. All aspects of an object that should be modifiable are encoded into *attributes*, of which each part can have an arbitrary number. Examples for such attributes are

Figure 4: An example of linear growth

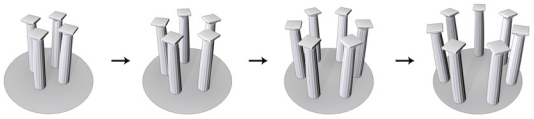


Figure 5: Circular expansion of columns

the object's transformation matrix, its local coordinate system, or its materials. For a spline or a parametric surface the control points are additional attributes; for light sources anything from the light type to parameters like brightness and shadow type are attributes as well.
Transformations operate on attributes only, leaving the underlying part definition unchanged. The transformation concept is no longer restricted to only affine transformations, but has been extended to include every possible form of user-defined action, from random distribution algorithms to Boolean or color-changing operations on components.

3. APPLICATIONS

Attributed collage grammars have a multitude of applications. Since the attribute and transformation system allows practically unlimited extensions to both the set of transformations and the objects that are considered, many modeling problems can be solved by either the addition of new attributes or new operations on existing attributes.

3.1 Linear growth

One problem that needs to be addressed when modeling in a recursive way with affine transformations is the restriction to exponential growth through repeated scaling operations (see [DKL02]). This problem also occurs in the construction of temples, e.g. where the floor and roof have to grow to match the addition of new columns. The occuring linear increase in area size cannot be matched by the recursive application of affine scaling.
We have solved this problem by introducing variables, which are used to alter the transformation parameters as needed during the derivation process.

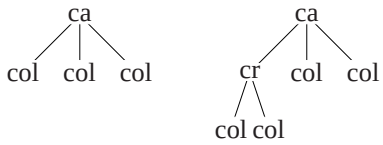
3.2 Recursive circular arrangement

A spatial function with many uses is one that arranges its arguments in a circular fashion around some central point. This can be done with benches in an amphitheatre, columns in a round temple or support beams in a pavillion, the petals of a flower, etc. When such a function is used recursively however, as is the case within the derivation process of a grammar, one must be a little more careful than when the same type of function is used just once on a fixed number of arguments.

Example 1
The following is just the rule section of a *regular tree grammar* [Dre00] where `S` is the start symbol, `C` is another nonterminal, `ca` is an operation that contains exactly one circular arrange transformation, `cr` changes the value of the global radius variable `r`, and `col` is a collage with the shape of a column. A section of the generated language (displayed on a circular base) is shown in Figure 5.

```
S -> ca[C, col, col],
C -> cr[col, C],
C -> col
```

Derivations are very straightforward, with the first two trees generated by the grammar looking like this:



And this is the relevant part of the algebra:

```
r = gl_var(2),

ca           = circularArrange(~r),
changeRadius = assign(~r=~r+1),

ca = < ca >,
cr = < changeRadius, id>
```

The very simple version of a circular arrangement transformation used here has only one parameter: the radius of the desired circle, $r \in \mathbb{R}$. It works the following way:

1. For argument collages $C_1, \ldots, C_k$ and a radius r, *circularArrange* takes the concatenation $\bigcup_{i=1}^{k}(C_i) = p_1 \cdots p_n$ of the parts p_i contained in the given collages C_i. (This way, the sequence of parts is maintained.)

2. Then all parts p_i are translated to the center of the coordinate system, where

3. every part p_i is first translated $-r$ along the z-axis and then rotated $\frac{360}{n} \cdot i$ degrees about the y-axis.

It might be interesting to note that the same circular arrangement of objects can also be achieved using only variables, but with considerably more effort.

△

Figure 6: Model of a Greek tholos generated with attributed collage grammars

4. CONCLUSION

In this paper, attributed collage grammars have been introduced as a framework for the grammatical modeling of scenes that is geared towards extension and customization by the user.

Future work will include specifying and implementing a constraint architecture that allows further influencing the selection and application of rules based on constraints specified by the user, and also automatically postprocessing the generated scene. Examples for such constraints are collision control and handling during the growth of a scene, or synchronization of rule application. Spatial coherence in the generated structures might also be enforced by constraints.

Another topic for future work is a referencing system for scene generators where grammars can access the yield of other scene generators as terminal symbols in their rules. This would allow modularizing the generation of scene components and therefore achieving a greater variation of results. This concept together with a sample implementation and possible applications were tentatively explored in the author's diploma thesis, where a first attempt to model the growth of an ancient settlement has been made using both collision constraints and grammar hierarchies.

Apart from further study of the method itself, attributed collage grammars will be applied in a number of case studies. In this context, the generation of ancient Greek architecture will continue to be studied with an emphasis on creating more detailed buildings as well as complete settlements. Grammar libraries will be developed and made available together with an implementation. Another interesting topic for such a case study is the domestic and religious Japanese architecture starting with the Heian period.

An implementation of tree generators, including a wide range of two-dimensional picture generation methods is provided with the software TREEBAG [DK01]. TREEBAG is written and maintained by Frank Drewes and has already been used as a framework for our first prototypical implementation of attributed collage grammars. In the future it will be extended to include a full implementation of a complex attributed collage algebra; it will also acquire plugin functionality for such commercial modeling packages as *Cinema 4d* and *Maya*.

5. REFERENCES

[DK99] Drewes F. and Kreowski H.-J. Picture generation by collage grammars. In Ehrig H., Engels G., Kreowski H.-J., and Rozenberg G., editors, *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 2, chapter 11, pages 397–457. World Scientific, 1999.

[DK01] Drewes F. and Klempien-Hinrichs R. TREEBAG. In Yu S. and Paun A., editors, *Proc. 5th Intl. Conference on Implementation and Application of Automata (CIAA 2000)*, volume 2088 of *Lecture Notes in Computer Science*, pages 329–330, 2001.

[DKL02] Drewes F., Kreowski H.-J., and Lapoire D. Criteria to disprove context-freeness of collage languages. *Theoretical Computer Science*, pages 1445–1458, 2002.

[Dre00] Drewes F. Tree-based picture generation. *Theoretical Computer Science*, 246:1–51, 2000.

[Dre04] Drewes F. *Grammatical Picture Generation – A Tree-Based Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004. To appear.

[EMP+98] Ebert D. S., Musgrave F. K., Peachey D., Perlin K., and Worley S. *Texturing and Modeling: A Procedural Approach*. AP Professional, 2nd edition, 1998.

[PL96] Prusinkiewicz P. and Lindenmayer A. *The algorithmic beauty of plants*. Springer-Verlag New York, Inc., 1996.

[PM01] Parish Y. I. H. and Müller P. Procedural modeling of cities. In *Proceedings of the 28th annual conference on computer graphics and interactive techniques*, pages 301–308, 2001.

[PMKL01] Prusinkiewicz P., Mündermann L., Karwowski R., and Lane B. The use of positional information in the modeling of plants. In *Proceedings of the 2001 conference on Computer Graphics*, pages 289–300. ACM Press, 2001.

[RB85] Reeves W. T. and Blau R. Approximate and probabilistic algorithms for shading and rendering structured particle systems. In *Proceedings of the 12th annual conference on computer graphics and interactive techniques*, pages 313–322, 1985.

Planning with Smart Objects

Tolga Abaci Jan Ciger Daniel Thalmann
Virtual Reality Laboratory
EPF Lausanne
CH-1015 Lausanne, Switzerland
{tolga.abaci | jan.ciger | daniel.thalmann}@epfl.ch

ABSTRACT

This paper presents a novel method of employing "smart objects" for problem solving in virtual environments. Smart objects were primarily used for behavioral animation in the past. The paper demonstrates how to use them for AI and planning purposes as well. We formally define which operations can be performed on a smart object in terms of their requirements and their effects. A planner uses this information to determine the correct sequence of actions needed to achieve a goal. This approach enables intelligent agents to solve problems requiring a collaboration of several agents and complex interactions with objects.

Keywords

Smart objects, artificial intelligence, virtual reality, animation, planning.

1. INTRODUCTION

A significant number of virtual reality applications require that the virtual characters are able to manipulate the objects in their environment. Such interactions can be arbitrarily complex and their precision requirements vary as well (i.e. ranging from simple, single-shot motions to sequences of numerous motions that require high accuracy). Many existing applications try to tackle this problem in ad-hoc ways; the usual solution is to combine pre-designed or motion-captured key frame animations with simple object animations. Another, more general, approach is to use a concept of smart objects where the responsibility for the animation is shared between the virtual character and the object itself [Kal01].

The smart objects paradigm has been introduced for interactions of virtual humans with virtual objects [Kal01]. It considers objects as agents where for each object interaction features and plans are defined. Even though smart objects are more flexible than other approaches when it comes to animation and behaviors, the fact that interaction plans are fixed imposes a severe limitation from the artificial intelligence point of view, reducing the capability to adapt to new situations and to solve more complex, dynamic problems. This could be addressed by the use of planning techniques.

Conference proceedings ISBN 80-903100-7-9
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

One of the first published works about AI planning is the STRIPS planner from 1971 [Fik71]. It introduced the concept of operators, with preconditions and effects. The state of the world is expressed using predicate calculus. This method of describing the planning problem is still popular and was used in many planners – e.g. UCPOP [Pen92], Prodigy [Vel95]. One of the most popular planners using the STRIPS representation is Graphplan [Blm97] and its many derivatives, such as Blackbox, Sensory Graphplan [Wel98], Temporal Graphplan [Smi99] and many others.

Our implementation employs a modified version of Sensory Graphplan (SGP). It extends the standard Graphplan and adds sensing actions and conditional effects. It builds contingency plans – plans where the initial truth value of some predicate may be uncertain and the planner plans for both eventualities indicating which actions have to be taken in each case (planning worlds). As such, it is more suitable for virtual reality simulations because the input language is much more expressive compared to the standard STRIPS-like planners.

In this paper, we present a method how to extend the smart object concept for use with SGP. Embedding the high-level information together with animation

data in the smart object allows for more efficient planning, because only relevant operations and data are considered. Another advantage is that the embedding allows integrating the creation of the logic data into the design pipeline. This ensures that the smart object animations and the corresponding high-level information are created at the same time and in a consistent way.

2. EXTENDING SMART OBJECTS

Smart objects provide not only the geometric information necessary for displaying them on the screen, but also semantic information useful for animation purposes. We store this information in the form of sets of attributes attached to the scene graph nodes of the object.

The attributes convey various kinds of information – e.g. important places on or around the object (e.g. where and how to position the hands of the virtual character in order to grasp it), animation sequences (e.g. a door opening) and general, non-geometric information associated with the object (e.g. weight or material properties) The semantic information in the smart object is used by the virtual characters to perform actions on/with the object, e.g. grasping, moving it, operating it (e.g. a machine or an elevator).

However, simple uses of semantic information are not sufficient if more complex behavior is desired. For example, moving of the crate (a smart object) is easily animated using a script and few attributes of the object. Moving the same crate by a virtual character through a closed door requires a higher level planning (i.e. "open the door first, if not open already and then push the crate through") and moving a heavier crate may require two virtual characters and careful planning to do it. All such simulation requirements put high demands on the script driving the virtual characters because it has to know about all possible situations which may occur. This is impractical and inefficient. Planning has the potential to solve this problem; however, with many possible actions it can become very cumbersome because the complexity of the search space explodes and certain cases may be simply intractable.

There is another way to address this problem. Just as we move the object-specific "animation intelligence" from the virtual characters to the smart objects, we can add also the planning and AI-related data there. The virtual character does not have to know how to interact with every kind of object; he can acquire the necessary capabilities on the fly from the object.

Smart objects [Kal01] already contain "interaction plans", which are essentially scripts containing the animation of the action itself. These scripts coordinate the human and object animations to create the intended result, which could be a virtual human pushing a crate, opening a door, etc. Our proposal for the extension of smart objects consists of the following items to be associated with each action:

- Preconditions for the action. These conditions have to be satisfied in order to be able to perform the action.
- Effects of the action. These predicates will be added/removed from the representation of the world state when the action is performed.

The action script is usually written in some high level scripting language (in our case Python), the preconditions and effects are expressed in PDDL format used by the STRIPS planners (i.e. Lisp-like).

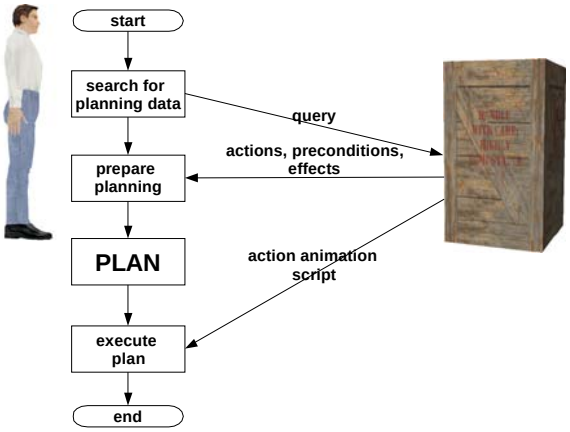


Figure 1 Algorithm outline

The general flow of events when interacting with extended smart objects is shown in Figure 1. The algorithm is as follows:

1. The virtual character collects the relevant information about the current state of the surrounding world. This is taken from the agent's own beliefs about the world and from the smart objects involved in the interaction.
2. Prepare planning step builds the problem representation.
3. Planning is performed. The result is a plan and the set of alternative planning worlds in case that there were some predicates with uncertain value in the initial state. There is always at least one planning world – in this case it corresponds to the initial state directly.
4. The virtual character executes the plan. The actions are taken from the plan, mapped to the corresponding low-level functions and executed.

Sensing actions scheduled by the planner are used to determine the status of the originally uncertain predicates from the initial state. The result determines which planning world the agent is in and therefore decides which branch of the plan has to be executed.

This general algorithm does not guarantee that the agent will be able to solve every solvable problem. It is best-effort heuristics only, because the knowledge of the agent about his surroundings is limited. The agent may not be aware of critical information needed to solve the problem. Another possibility for planning failure comes from the fact that it is very hard to select the relevant objects from which to retrieve the planning information. In fact, this is as hard as the planning problem itself, because the agent will know whether the object is or is not relevant to the task only after the plan is built.

3. RESULTS

The described system was implemented as an extension of our existing virtual reality platform VHD++ [Pon03] and our agent framework described in [Aba04]. The agents driving the virtual characters are implemented as Python scripts; the planner is running as compiled LISP code.

We constructed a simple scenario for evaluation of the proposed approach. In our case, we have a virtual art gallery which received two crates (one large one, one smaller one) with new art. The goal of the two virtual humans is to move the crates inside the lobby of the gallery. Both agents have basic facilities for teamwork (forming a team, disbanding a team), communication and some rudimentary capabilities, like navigation in the virtual environment. However, they do not have any a-priori knowledge about how the crates can be moved.

Our crates are modeled using the extended smart object approach. They contain reference to the file with the geometry of the object (mesh, textures, etc.), position and orientation data for the hands of the agents during the animation, proper position where the agent has to be before the animation script is started and finally the planning data.

The difference between the small and the large crates is that we have defined the large crate as a heavy object and therefore it needs two people to move it.

Planning data have associated animation scripts, in our case a simple animation moving the virtual human and the crate on the screen, using inverse kinematics to keep the hands in position.

a.

```
((((TRANSPORT GINO SMALL_BOX FRONTYARD
    LOBBY))))
```

b.

```
((((RECRUIT-HELP SELF TEAMMATE1)))
 (((MOVE TEAMMATE1 ANYWHERE FRONTYARD))
  (((DECLARE-DOOR-PASSABLE SELF DOOR)))
 (((PREPARE-PUSH SELF BIG_BOX))
  ((PREPARE-PUSH TEAMMATE1 BIG_BOX)))
 (((TEAM-PUSH SELF TEAMMATE1 BIG_BOX
    FRONTYARD LOBBY))))
```

Figure 2 Plans for pushing the small crate (a) and the big crate (b)

We are asking the agent Gino to form a plan to get the big box to the lobby and afterwards to execute it (see Figure 2b). The plan tells the agent to get somebody to help. “SELF” is the agent which submitted the planning request and “TEAMMATE1” is a collaborating agent asked to help. During the execution of the plan the originating agent (team leader) will negotiate the team formation and when successful, it will substitute the real name of a team member for “TEAMMATE1”. The next step consists of moving the teammate from his current position to the front yard, where the crates are. “Anywhere” denotes a special place, from which it is possible to go everywhere. The team leader does not care where the teammate is at the moment, but it needs to establish him into a known state (and move him to the necessary place). The operation “DECLARE-DOOR-PASSABLE” is a helper operation which declares the two places on the sides of the door as connected in case that the door is open. The remaining two steps are self-explanatory. The resulting action after executing the associated animation script from the smart object is depicted in Figure 3.

Figure 3 Two agents pushing the large crate

For the small crate, the resulting plan is simpler since the agent can immediately perform the needed action – Figure 2a. The animation performed by the associated animation script is shown in Figure 4.

Figure 4 Gino pushing the small crate

4. CONCLUSIONS

Our scenario shows the possibility of the virtual characters (agents) to "learn" how to interact with previously not encountered objects by exploiting the information stored in them. Furthermore, such information encapsulation allows us to let the agent work with only the information relevant to his task, simplifying the planning process.

From the design point of view, keeping the animation data and formal representation of the interaction in one place is beneficial for ensuring that all required elements will be created. It is feasible to create an authoring tool for extended smart objects which will help generate the formal representation and animation script template from the specified description.

The process of agent development is simplified as well, because the developer does not have to design an agent capable of performing many specialized actions. It is sufficient to create a simple agent with few basic capabilities (such as navigating the virtual environment) and leave the rest to a generic procedure created on the fly based on the declarations and code defined in the extended smart object. It could be considered as filling in an action template based on the data in the smart object.

We demonstrated how the smart object approach can be extended to handle the formal representation of the interaction. The smart objects can handle not only the animation alone but also the formal description of it. This tight coupling between animation and its formal description enables the virtual characters to perform sophisticated actions which are either very complex to achieve otherwise or just impossible outright. It allows us to apply the known artificial intelligence techniques to improve the realism of the simulation and to provide richer experience to the user.

5. ACKNOWLEDGEMENTS

The project was sponsored by the Federal Office for Science and Education in the Framework of the EU Network of Excellence AIM@SHAPE.

6. REFERENCES

[Kal01] Kallmann, M. Object interaction in real-time virtual environments. Phd Thesis, EPFL, 2001.

[Blm97] Blum, L. A. and Furst, L. M. Fast planning through planning graph analysis. Journal of Artificial Intelligence, Vol.90, pp.281-300, 1997.

[Fik71] Fikes R. and Nilsson, J. N. STRIPS: A new approach to the application of theorem proving to problem solving. Journal of Artificial Intelligence, Vol.2, pp.189-208, 1971.

[Pen92] Penberthy J. S. and Weld D. S. UCPOP: A sound, complete, partial-order planner for ADL. Third International Conference on Knowledge Representation and Reasoning, Cambridge, MA, October 1992.

[Vel95] Veloso M., Carbonell J., Perez A., Borrajo D., Fink, E. and Blythe J. Integrating planning and learning: The PRODIGY Architecture. Journal of Experimental and Theoretical Artificial Intelligence, Vol.7, 1995.

[Wel98] Weld D. S., Anderson C. R. and Smith D. E. Extending Graphplan to handle uncertainty & sensing actions. In proceedings of AAAI 98, 1998.

[Smi99] Smith D. E. and Weld D. S. Temporal planning with mutual exclusion reasoning. In proceedings of IJCAI, pp.326-337, 1999.

[Pon03] Ponder, M., Papagiannakis, G., Molet T., Magnenat-Thalmann N., Thalmann D., VHD++ development framework: Towards extendible, component based VR/AR simulation engine featuring advanced virtual character technologies, Computer Graphics International (CGI), pp. 96-104, 2003.

[Aba04] Abaci, T., Ciger J. and Thalmann D. Speculative Planning With Delegation. In proceedings of CYBERWORLDS 2004, to appear.

Adaptive Tessellation of Bézier Surfaces Based on Displacement Maps

F.J. Espino[1] M. Bóo[1] M. Amor[2] J.D. Bruguera[1]

[1]Dept. of Electronic and Computer Eng., Univ. of Santiago de Compostela,
E-mail: {javi, mboo, bruguera}@dec.usc.es

[2]Dept. of Electronics and Systems, Univ. of A Coruña
E-mail: margamor@udc.es

ABSTRACT

Bézier surfaces are widely used in computer graphics applications. Rendering of such surfaces is commonly performed by tessellation. In order to generate less triangles for high quality surfaces, adaptive tessellation algorithms are better. The geometric tests used by these algorithms perform vector computations of high latency that decreases the performance of the algorithm. We propose an adaptive tessellation algorithm that avoids vector computations for tests, replacing them with scalar computations. This way, latency of tests is reduced and therefore, performance improved.

Keywords: Bézier surfaces, tessellation, subdivision, displacement map

1 INTRODUCTION

Bézier surfaces are used in several computer graphics applications. Due to the excellent performance of hardware based methods for triangle meshes, the strategy mostly employed for rendering is the surface tessellation [Kumar96, Moret01]. There are different methods and hardware proposals [Kumar96, Moret01] for Bézier surfaces tessellation. Adaptive tessellation methods increase detail where required [Chung00, Espin03]. However, redundant computation of vertices is made, because every shared vertex is computed once per triangle. The method proposed in [Espin04] improves performance processing groups of triangles instead of individual triangles. The tests employed for adaptive tessellation are based on the computation of all candidate vertices and normal vectors to be inserted and their posterior analysis. This implies much computation, because vertices that do not need to be inserted are computed.

We propose an adaptive tessellation algorithm based on the *layer strips* method [Espin04]. Edge tests are performed using scalar computations over a displacement map [Lee00]. The displacement map is sent together with a coarse tessellation to the the graphics unit. During subdivision, no vertex is computed before a decision is made, improving performance. As a result, good quality meshes are generated with less computations.

WSCG SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4,2005
Plzen, Czech Republic.

2 ADAPTIVE TESSELLATION BASED ON DISPLACEMENT MAPS

Adaptive tessellation algorithms analyze geometric information of the surface in order to subdivide with high detail the regions of high curvature. This way, the resulting meshes have high quality with less triangles than uniform tes-

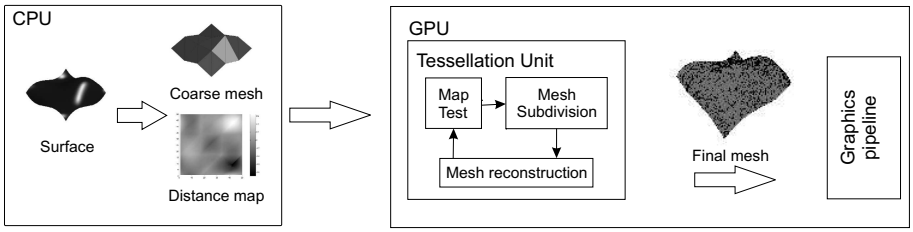


Figure 1: Diagram of the adaptive tessellation algorithm

sellations. The simplest algorithms [Chung00, Espin03] generate an initial coarse tessellation and then processes each triangle iteratively, analyzing each edge for deciding the subdivision pattern of the triangle. Several tests can be used, obtaining different results in number of triangles and quality of the tessellation. The problem of processing triangles individually is that vertices inserted in edges are re-computed. Moreover, tests need the coordinates and normal vector of the vertex that is going to be inserted, so useless computation is performed.

In this section we propose an algorithm that avoids unnecessary vertex computations. The coarse tessellation and its displacement map are computed and sent for rendering. The map is a matrix of distances between the initial coarse tessellation and the surface. This information is employed to generate the adaptive subdivided mesh. The resulting triangle mesh is rendered in the graphics system. This algorithm is a good candidate to be implemented as a first stage of the Graphics Unit (GPU) as shown in Figure 1. This way, communications between CPU and GPU are kept low. Next, the algorithm is detailed.

2.1 Displacement Map Generation

Generation of the displacement map is made through the following steps. First, the surface is sampled in $M \times N$ points and, for each point, a distance value is computed. In general, the distance is computed as the distance between the surface point and the plane of the triangle with a normal vector intersecting the point. In Figure 2 a 2D simplification of a surface and a coarse tessellation is shown. Surface point S_1 represents the general case, where the point is over the normal of only one triangle T_1 and distance d_1 is computed.

Exceptional cases can be found. For these cases we propose the following. In Figure 2, point S_2

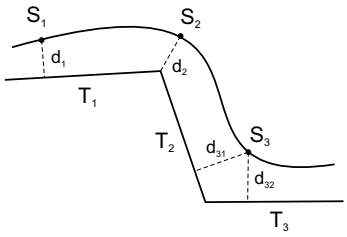


Figure 2: Cases in distance map generation

has no triangle with a normal vector intersecting it, so the distance is computed as the distance d_2 to the nearest edge (edge between triangles T_1 and T_2). Finally, both triangles T_2 and T_3 have normal vectors that intersect point S_3, so the nearest triangle is chosen for computing distance d_3.

The necessary resolution ($M \times N$) of maps can be computed through a Fourier analysis, using Nyquist criteria to select the minimum map resolution. However, the simulations of our algorithm show that lower resolutions do not decrease the quality of the generated meshes. This is due to the fact that maps are not used for surface reconstruction, but only for test purposes.

2.2 Distance Map Test

The edge analysis is performed comparing distances associated to vertices and inserting vertices in those regions where the distance differences are greater than a threshold value. Parametric coordinates are used to read the map values. Due to the limited resolution of the map, the distances are linearly interpolated from the available distances.

Three tests are distinguished depending on how many points are used for distance comparisons: two-point test ($Map2p$), four-point test ($Map4p$) and eight-point test($Map8p$). Figure 3 represents the data used for these tests: d_1 and d_2 are the

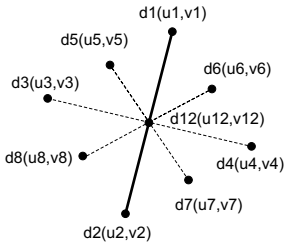


Figure 3: Distance values for additional test

Figure	T	cmesh(KB)	map(KB)	Error
Surface1	18	0.84	1.00	0.2566
Surface2	18	0.84	1.00	0.1604
Teacup	468	21.94	26.00	0.0273

Table 1: Initial tessellations

distances associated to the vertices of the edge; d_{12}, the distance of the candidate vertex to be inserted if the test is positive; and d_3 to d_8, the additional points for tests $Map4p$ and $Map8p$. The parameters u and v are the parametric coordinates of the vertices, used for accessing the distance map.

The test $Map2p$ only uses edge related information, i.e. points d_1, d_2 and the tested vertex d_{12}. Differences are computed and compared to a threshold value according to:

$$Map2p = (|d_1 - d_{12}| > th)\ OR\ (|d_2 - d_{12}| > th) \quad (1)$$

The $Map4p$ test uses two additional distance values (d_3 and d_4 in Figure 3) to improve the quality of the tessellation. The parametric coordinates needed for accessing the map are computed with:

$$\begin{array}{l}(u_3, v_3) = (u_{12} - v_e,\ v_{12} + u_e) \\ (u_4, v_4) = (u_{12} + v_e,\ v_{12} - u_e)\end{array} \quad (2)$$

where $(u_e, v_e) = (u_1 - u_{12}, v_1 - v_{12})$. The differences are compared together with differences of $Map2p$ to the threshold value using:

$$\begin{array}{rcl} Map4p & = & Map2p\ OR\ (|d_3 - d_{12}| > th) \\ & & OR\ (|d_4 - d_{12}| > th) \end{array} \quad (3)$$

where $Map2p$ are the comparisons of Eq. (1). This test is extended to $Map8p$ where four additional distances (d_5, d_6, d_7 and d_8) are considered. These tests have been selected due to their simplicity of computation and the good results obtained during simulation.

3 RESULTS

The algorithm has been simulated in C++ language and tested for several figures. Results are shown for three figures: two individual surfaces and a teacup.

Table 1 shows the memory requirements and mesh error of the coarse tessellations for the three figures. Column *cmesh* shows the memory requirement of the coarse mesh (18 triangles per surface[1] where *teacup* has 26 surfaces). Column *map* shows the requirements of the displacement map with a resolution of 16x16 points. Keeping these sizes, the communications requirements are kept very low (1.84KB in total). The last column of Table 1 shows the mesh error of the coarse meshes.

Table 2 shows the mesh error for the subdivided meshes with map tests (columns $Map2p$, $Map4p$ and $Map8p$). The error of flat test [Espin04] is also included in column $flat$. Results for different number of triangles of final mesh (columns T) are shown. Of course, any of the adaptive methods reduces the error respect to the coarse mesh.

Simulation data shows that using more distances in test improves quality of subdivided meshes. The difference in quality between $Map4p$ and $Map8p$ is small and the cost, larger. Therefore, $Map4p$ have the best quality and cost trade-off.

Quality of meshes subdivided with map tests are very close to those generated with flat test. For example, for $surface2$ tessellated with around 5000 triangles quality is 0.00164 and 0.00064 for $Map4p$ and $flat$ respectively; quality is even better for more complex figures, like *teacup*.

Final considerations about mesh quality can be made by looking at Figure 4. Figures 4(a), 4(b) and 4(c) shows the subdivided meshes using the $Map4p$ test with approximately 5000 triangles. As mentioned above, quality of the coarse tessellation is improved with higher detail in curved regions.

With the flat tests [Espin04] when a vertex is computed and not inserted, useless computations are performed. In our proposal the decisions are computed first, so useless computations are avoided and performance is improved.

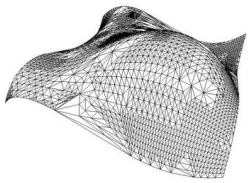
In summary, we propose to reduce the computations associated to vertices not finally inserted using high quality subdivided meshes and low communication and memory requirements.

[1] ıyer Strip [Espin04] requires 36 Bytes per vertex.

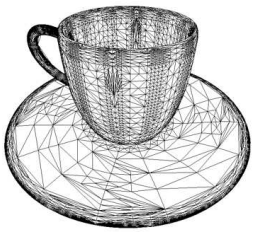
Figure	Map2p		Map4p		Map8p		Flat	
	T	E	T	E	T	E	T	E
surface1	5034	0.00025	5846	0.00167	5848	0.00168	4374	0.00025
	4388	0.00047	4695	0.00182	4697	0.00183	3600	0.00053
	2070	0.00160	2037	0.00438	2043	0.00437	1926	0.00132
surface2	5831	0.00344	6010	0.00162	5898	0.00164	5645	0.00064
	4384	0.00213	4045	0.00212	4306	0.00168	4003	0.00092
	2041	0.00404	2093	0.00403	2009	0.00375	2074	0.00183
teacup	20100	0.00175	23196	0.00145	17987	0.00238	20604	0.00242
	13380	0.00239	14859	0.00297	14589	0.00318	12220	0.00363
	7835	0.00429	7284	0.00406	7207	0.00427	9630	0.00390

Table 2: Tessellation results

(a) Surface1 (Map4p test)



(b) Surface2 (Map4p test)



(c) Teacup (Map4p test)

Figure 4: Tessellated meshes

4 CONCLUSIONS

In this paper an adaptive tessellation algorithm for Bézier surfaces is presented. It combines the methods of previous algorithms with the displacement map concept in order to obtain more efficient and implementable tests.

The method we propose is based on displacement maps and performs simpler tests that do not need the precomputation of vertices, increasing performance. Quality results for subdivided meshes are very close to previous algorithms, keeping communication requirements low.

The different tests based on using a set of distances (from 2 to 8) have been analyzed and compared, concluding that the use of 4 distances for test has the best trade off between quality and computation load.

ACKNOWLEGMENTS

This work was partially supported by the Ministry of Science and Technology of Spain under contract MCYT-FEDER TIC2001-3694-C02-01 and by the Secretaría Xeral I+D of Galicia (Spain) under contract PGIDIT03TIC10502PR.

REFERENCES

[Chung00] A.J. Chung and A.J. Field. A Simple Recursive Tessellator for Adaptive Surface Triangulation. *Journal of Graphics Tools: JGT*, 5(3):1–9, 2000.

[Espin03] F. J. Espino, M. Bóo, M. Amor, and J. D. Bruguera. Adaptive Tessellation of NURBS Surfaces. *Journal of WSCG*, 11(1):133–140, 2003.

[Espin04] F. J. Espino, M. Bóo, M. Amor, and J. D. Bruguera. Hardware Support for Adaptive Tessellation of Bézier Surfaces Based on Local Tests. Technical report, www.ac.usc.es, 2004.

[Kumar96] S. Kumar, D. Manocha, and A. Lastra. Interactive Display of Large-Scaled NURBS Models. *IEEE Trans. on Vis. and Comp. Graphics*, 2(4):323–336, 1996.

[Lee00] Aaron Lee, Henry Moreton, and Hugues Hoppe. Displaced Subdivision Surfaces. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 85–94, 2000.

[Moret01] H. Moreton. Watertight Tessellation Using Forward Differencing. In *ACM Siggraph/Eurographics Workshop on Graphics Hardware*, pages 25–32, 2001.

Visualization of 3D Scientific Datasets Based on Interactive Clipping

Gaurav Khanduja
Department of Computer Science
Louisiana State University
Baton Rouge, Louisiana 70803,
USA
khanduja@bit.csc.lsu.edu

Bijaya B Karki
Department of Computer Science
Louisiana State University
Baton Rouge, Louisiana 70803,
USA
karki@bit.csc.lsu.edu

ABSTRACT

We have exploited clipping in combination with multiresolution rendering to visualize large-scale three-dimensional (3D) scientific datasets. Our interactive clipping approach involves the dynamic manipulation of a clip plane to expose any cross-section of a given volume data and the subsequent adjustment of the clipped surface to the best view position. The data are rendered as stacks of 2D textures at high-resolution (HR, the original resolution of input data/image) and low-resolution (LR, sampling at reduced-resolution). While clipping operations take place in LR to achieve an interactive frame rate, the best-view position also supports HR. Our approach thus enables a real-time exploration of any interior region of 3D volumetric data at a desired resolution. Its successful demonstration has been carried out for two scientific datasets, which are the simulated electronic density distributions in a crystal and confocal microscopy images of tissues in a plan stem.

Keywords

Clipping, volume rendering, scientific visualization

1. INTRODUCTION

High-end scientific computation and three-dimensional imaging systems are producing ever-increasing amounts of volume data. Visualization of such datasets still poses tremendous challenges primarily because it involves mapping from 3D space to 2D display. A number of visualization techniques exist [sch97]. One category includes surface rendering techniques, which convert structures in 3D data into surface representations, e.g., isosurfaces. The other category deals with direct volume rendering techniques, which involve the generation of a RGBA volume from the data, reconstruction of a continuous function, and projecting it onto the 2D viewing plane. No interior information is thrown away in volume rendering but it has difficult interpretation of the cloudy interiors and slow rendering.

WSCG SHORT papers ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

The other important category of 3D visualization techniques is related to clipping, which allows us to uncover important, otherwise hidden details of a dataset to the extent which is often not feasible with other techniques. The interactive clipping approach we have adopted here is not an entirely new idea. Rather it is a practical implementation which shows how even a simple form of clipping (such as a planar clipping) can be exploited in an effective way to provide a real-time multi-resolution visualization of large-scale datasets. The clip plane once it is generated is manipulated in real time to expose an arbitrary cross-section through the volume data. The clipped surface is then adjusted to attain the best view position in which the surface is visible to its high exposure and rendered at high-resolution. A portable and flexible interactive 3D visualization system is being developed based on the proposed approach. It is expected to guarantee that one is able to view every possible cross-section of the volume data at desired resolution. Successful applications are demonstrated for two important scientific datasets. They represent the electronic charge density distributions in a solid material taken from large-scale simulations and the tissue images of a plant stem generated by confocal microscopy.

2. PREVIOUS WORK

Clipping was previously performed using a variety of clip geometry, from simple plane, to multiple planes, to complex polygons. For example, Van

Gelder and Kim used clip planes [Van96]. Weiskopf et al. have proposed techniques for volume clipping with complex geometries, which are based on the depth structure and voxelization of the clip geometry [Weisk02] and also involve subsequent shading of the clipping surface [Weisk03]. Some studies have exploited isosurface clipping [Ferg92]. Often clipping is combined with texture-based rendering of the data to exploit the advanced fragment operations supported by graphics hardware. Also hardware-based textures help improve interactivity, compared to point-based rendering [Wilson02]. However, for very large datasets now routinely produced by simulations and experiments, the amount of available texture memory and the fill rate of the graphics system can still pose limitations. To overcome these, multi-resolution rendering techniques such as combining texture mapping and point-based rendering [Wilson02] or using lower sampling rates for areas of low interest [LaMar99, Weiler00] were previously proposed. However, no clipping has yet been implemented in combination with any multi-resolution rendering option.

3. CLIPPING-BASED VISUALIZATION

We have explored how clipping can be exploited in an efficient-effective manner to visualize large-scale 3D scientific datasets by combining it with dual-resolution texture-based volume rendering. The overall goal is to support different specific services such as rapid navigation through the data, emphasis on data features and realistic rendering without sacrificing interactivity, which are crucial in real-time exploration of massive datasets. Here, interactive clipping is adopted to uncover every interior region of a given volumetric data so that one can visualize the clipped surface at its maximum exposure and at the desired level of details.

3.1 Interactive clipping

Our interactive clipping approach involves the generation and subsequent adjustment of clip geometry, which is simply a plane. The clip plane is represented by the equation $Ax + By + Cz + D = 0$ in a 3D space. OpenGL [Woo99] is used to define the clip plane and perform all transformations the plane undergoes subsequently. The clip plane like any other geometry is transformed according to the current viewing matrix. All points with eye coordinates (x_e, y_e, z_e, w_e) that satisfy

$$(A\ B\ C\ D)M^{1}(x_e, y_e, z_e, w_e)^{T} >= 0$$

lie in the half-space defined by the plane, where $\boldsymbol{M}$ is the current model-view matrix. All points not in this half-space are clipped away.

The clip plane is initiated by choosing any two points, (x_1, y_1) and (x_2, y_2), on the display screen, which is initially set parallel to the x-y plane. The initial plane is assumed to be parallel to the z-axis passing through the origin. The required coefficients are thus given by

$$A = y_2 - y_1;\ B = -(x_2 - x_1);\ C = 0;\ D = 0$$

Once the clip plane is defined, it's position and orientation can be changed in real time in order to cut away the given volume at any orientation and any position. Varying the coefficients A, B and C enables an arbitrary rotation of the clip plane in a 3D space. For translation, the coefficient D is varied; the plane moves away from the origin in either direction along the normal to the plane passing through the origin.

3.2 Best-view mode

In the process of interactively changing the position and orientation of the clip plane, it is likely that effective visible area of the clipped surface becomes smaller and eventually disappears. In other words, the user cannot see the surface clearly because the plane tends to be parallel to the viewing direction. In such case, it is important to adjust the position/orientation of the clipped geometry so that the surface is again visible at larger exposure. Here, we want to introduce the idea of the *best-view* mode by which we mean the following two properties associated with it. First, the clipped surface is oriented nearly perpendicular to the viewing direction so that the surface is portrayed at its nearly maximum exposure. Second, the cutaway surface is rendered at high-resolution so that the details of data are displayed. The second feature can further be extended to support multi-resolution and additional display options such as isolines or glyphs or streamlines depending on the properties of the data that need to be understood.

To guarantee an interactive frame rate, particularly, when the datasets become very large, it is desirable to support an automatic option to attain the *best-view* position whenever it is needed. This requires tracking all transformations the clip plane has gone through and using this information to change the current view of the clipped object accordingly. For instance, a simple approach, say for our initial setting of the clip plane, involves the following two rotations: The first rotation is about the z-axis since the initial clip plane is set parallel to the z-axis and is rotated about this axis. The second rotation is about the vector, which is defined by the intersection of the clip and x-y planes. As a result, the clipped surface becomes nearly perpendicular to the z-axis, which is also the current viewing direction. After reaching the *best-view* position, the clip plane can further be manipulated by bringing back the object to the low-resolution mode.

3.3 Rendering

A given 3D volume dataset is rendered as stack of 2D slices in the form of 2D textures [Cab94]. Rendering is performed at two levels of resolution, n

the form of 2D textures. The high-resolution (HR) mode corresponds to the original fine grid of the input data (or equivalently, the original resolution of the input images). On the other hand, the low-resolution (LR) mode uses the re-sampled data at a lower resolution than the original resolution. Rendering speed (number of frames per second) is shown to differ substantially between the HR and LR rendering modes. HR is not fast enough to yield an interactive frame rate even for the data sizes we have considered here. However, it carries the complete information contained in the data. On the other hand, LR gives a much better frame rate but it suffers from the loss of information. The observed trade-off between rendering speed and resolution is carefully balanced in our visualization approach. In particular, the planar clipping is supported in the LR mode whereas the *best-view* option is also supported in the HR mode. This means that one can navigate through the volume data in the LR mode without hampering interactivity and switch to the HR mode to retain the details of data.

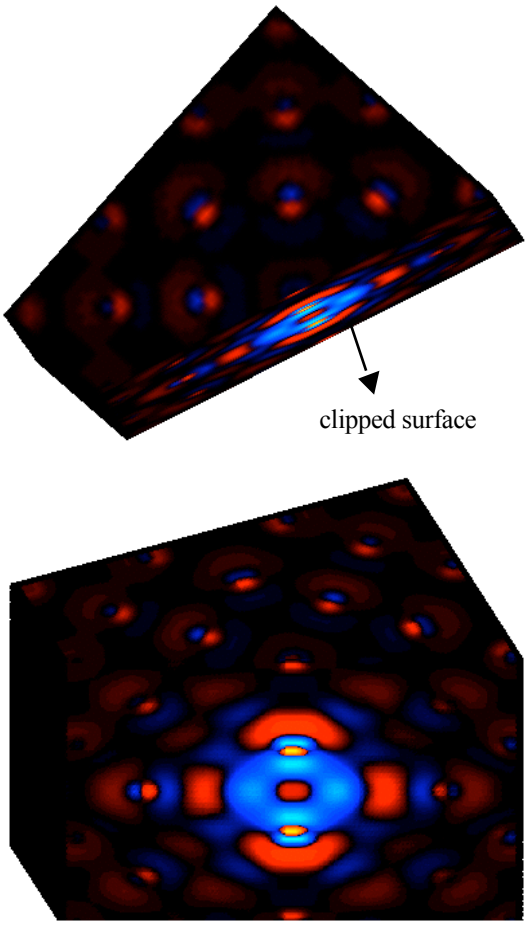


Figure 1: Charge density visualization: Clipping in the LR mode (top) and best-view rendering in the HR (bottom) modes. Electrons are depleted from blue regions whereas electrons are deposited in red regions due to a vacancy defect located at the center of the system.

4. APPLICATIONS

4.1. Charge density data

Here, we visualize the 3D electronic density data produced by massive quantum mechanical simulations. The simulated MgO system consists of 216 atoms and 864 valence electrons, which are distributed throughout the system. The electron density data on a regular Cartesian 512^3 grid is considered. Such simulations are expected to produce even larger datasets, which need to be visualized to gain insight into the electronic structures in the presence of static (such as defects) and dynamic (such as diffusion, deformation) processes in a material system. The given data is mapped to 512 textures; each texture has 512 x 512 pixels. Rendering them as a stack results in the HR mode. A stack of 128 textures at 128 x 128 resolution is used for the LR mode. Figure 1 shows clipped surface in the HR and LR modes. The performance measurements are carried out on Dell latitude of 2.40 GHz processor, 1.00 GB RAM for the window display size of 512^2. The calculated frame rates for clipping in the LR and HR modes are, respectively, 35 and 4. This difference in the clipping speed is substantial and should be important for larger datasets.

4.2. Confocal data

Three-dimensional imaging systems often produce large collections of 2D images of the interior of a 3D structure at different depths. The present application considers the confocal data containing 200 images, each with 512 x 512 pixels, of tissues of a plant stem. The visualization process starts with reconstruction of a 3D structure to which these images originally belong by rendering the given set of 200 tissue images as a stack of 2D textures. HR, which retains the original resolution of the input images, represents the one-to-one mapping between texels and pixels. LR corresponds to the stack of 50 textures at 128 x 128 resolution. Figure 2 shows clipped surface in the HR and LR modes. For 512^2 window display, the calculated frame rates for clipping in the LR and HR modes are, respectively, 28 and 0.6.

5. CONCLUSIONS

We have explored how clipping can be exploited in an interactive manner to visualize massive three-dimensional datasets (for calculated charge density and confocal tissue images) produced by large-scale scientific simulations and 3D imaging systems. In essence, the proposed approach involves the dynamic manipulation of the clip plane to expose any cross-

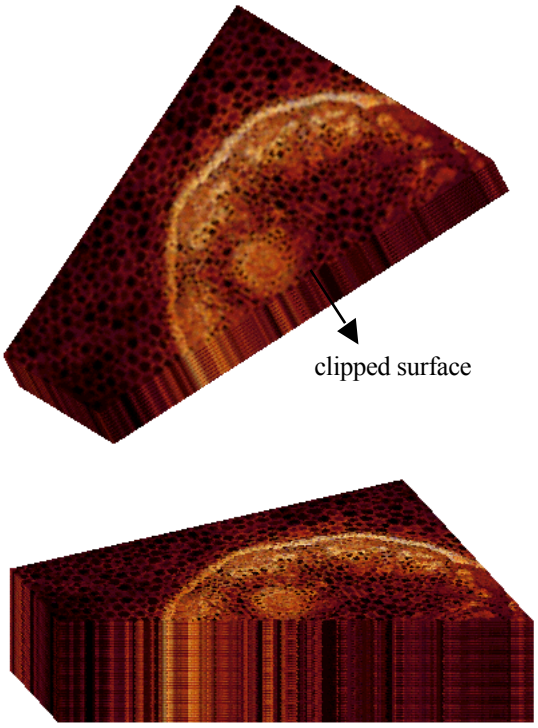


Figure 2: Confocal data visualization: Clipping in LR (top) and best-view rendering in HR (bottom). Twenty-five images (each with 512 x 512 pixels) were replicated to generate 200 images used in HR.

section of the volume data and subsequent adjustment of the clipped surface to the best view position. While all clipping operations are supported in the low-resolution (LR) mode to achieve interactivity, the best-view position is also supported in the high-resolution (HR) mode (i.e., the original resolution of the input data/image). It is expected that the proposed interactive clipping combined with dual-resolution rendering (which can essentially be extended to multi-resolution and multi-scale rendering) will be a powerful approach to interactive visualization of ever larger sets of three-dimensional scientific data.

6. ACKNOWLEDGMENTS

Our thanks go to J. Lynn for providing the confocal data, and S.S Iyengar and R. Kannan for useful discussion. This work is supported by NSF Career (EAR 0347204) and ITR (ATM 0426601) grants.

7. REFERENCES

[Cab94] B. Cabral, N Cam and J Foran. Accelerated volume rendering and tomographic reconstruction using texture-mapping hardware. In *1994 Symposium on Volume Visualization*, pages 91-98, 1994.

[Forg92] Forguson. Visual Kinematics, Inc. (Mountain View, CA USA), FOCUS User Manual, Release 1.2., 1992.

[LaMar99] E. LaMar, B. Hamann, and K. Joy. Multiresolution techniques for interactive texture-based volume visualization. In *Proceedings Visualization 99*, pages 355-361. ACM Press, New York, 1999.

[Sch97] W. Schroeder, K. Martin and B. Lorensen. *Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*, Prentice Hall, 1997.

[Weiler00] M. Weiler, R. Westermann, C. Hansen, K. Zimmerman, and T.Erl. Level-of-detail volume rendering via 3D textures. In *Proceedings of IEEE Volume Visualization 2000*, 2000.

[Weisk02] D. Weiskopf, K. Engel and T. Ertl. Volume clipping via per-fragment operations in texture-based volume visualization, *IEEE Visualization 2002 Proceedings*, pages 93-100, ACM Press, 2002

[Weisk03] D. Weiskopf, K. Engel, and T. Ertl Interactive clipping techniques for texture-based volume visualization and volume shading. *IEEE Trans. on Visualization and Computer Graphics*, 2003.

[Wilson02] B. Wilson, K. Ma and P. S. McCormick. A hardware-assisted hybrid rendering technique for interactive volume visualization. *Proc. of Volume Visualization and Graphics Symposium 2002,* 2002

[Woo99] M. Woo, J. Neider, T. Davis and D. Shreiner. *OpenGL Programming Guide: The Official Guide to Learning OpenGL*, Addison-Wesley, 1999.

[Van96] A. Van Gelder and K. Kim. Direct volume rendering with shading via three-dimensional textures. In *1996 Symposium on Volume Visualization*, pages 23-30, 1996.

Scaling for MEMS Virtual Prototyping: Size and Motion Dynamics Visualizations

Zhaoyi Li
Griffith University
Faculty of Engineering and IT
Australia, Gold Coast, QLD 9726
Z.Li@griffith.edu.au

Renate Sitte
Griffith University
Faculty of Engineering and IT
Australia, Gold Coast, QLD 9726
r.sitte@griffith.edu.au

ABSTRACT

This paper deals with the challenges found in scaling velocity and size for a CAD virtual prototyping system for Micro-Electro-Mechanical Systems (MEMS). These are miniaturized machines or devices used e.g. in medical or motorcar applications. In a MEMS the size and velocities of its moving parts can be vastly different, spanning several orders of magnitude. This makes it difficult to show the devices being designed in virtual action on a computer screen. A simple down scaling to bring the fast moving object into an observable range, can bring the slow one to a standstill. The image would no longer be a truthful scaling. A similar dilemma happens with the downscaling of small objects, because they would not be observable anymore. Our research aims at finding default values and their boundaries for the parameters that determine the scaled size and motion of objects relative to each other on a computer screen. This is required for informative and pleasant, truthful images that are suitable for a Virtual Prototyping system. In this paper we analyze and illustrate these issues with the example of a micropump and the layered fluidic flow in a microchannel.

Keywords

Virtual Prototyping, Scientific Visualization, MEMS, Stroboscopic Illumination.

1. INTRODUCTION

Since the proliferation of CAD tools, visualizations have gained importance, providing feedback at the time of design. Traditionally the results of simulations are displayed as plots and curves. Multi-dimensional multivariate visualizations have now been around for several decades, e.g. environmental maps of pluviosity. In our research, we are going a step further: displaying results of predictive calculations on the very design images of the structures they represent thus adding to the information content they can offer. Our environment is in Micro-Electro-Mechanical systems (MEMS) CAD development. MEMS are minute devices that are in widespread use, e.g. in airbag triggers, inkjet print heads, optical, medical, or other applications. With ever increasing new applications development tools with sophisticated modeling and simulation software are required to reduce the lengthy prototyping and optimization period. By their very nature, MEMS devices are microscopic, hence difficult to observe. In the macroscopic world, inertia and gravity dominate the motion of objects. To the contrary, in the microscopic domain of MEMS adhesion and friction are the dominant forces. Therefore, MEMS designers cannot use their intuition on how things behave. Because of the different dominant forces, MEMS cannot be simply downscaled counterparts of larger mechanical machines, requiring innovative designs and arrangements of their components, whose effects are often not fully understood.

Conference proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

Virtual reality can provide valuable assistance for data analysis. Human factors contribute significantly to the visual perception process. Perception, visual illusion, color, depth perception, contrast sensitivity, are vital in computer graphics and VR, but there is still need for human perceptual research [KS04]. The effectiveness of a visualization depends on perception, cognition, and the user's specific task and goals. Studies are usually confined to human-

computer-interaction (HCI). Studies on specific problems of visual data presentation are rare [TM04]. Our research aims at producing VR dynamic images of MEMS in design. We have addressed the problem of displaying different velocities using simulated stroboscopic illumination, but we still need to find the boundaries of to what extent this can be used. In this paper, we examine the boundaries to which scaling and filtering are meaningful from a human perspective. The idea is to place an image with a default size and motion on the screen, but allow a user to change those settings for an individually pleasant image, or zoom in/out for observation of detail. By pleasant we mean not flickering, not moving backwards, no confusing lines, or other optical illusions. One of the questions we seek to answer is, from what resolution values on is it worthwhile to show minute dynamics such as layers in a microchannel's fluidic flow, when initially a whole pump is shown with its flexing membrane, and channels that are very thin in comparison with the other structures. To do this we need not only default settings, but also any constraining boundary values as maximum settings

2. BACKGROUND ON MOTION PERCEPTION

The computational problem of motion perception is to convert information about image motion, stored in a space-time diagram, into information about the velocity of a moving object.

An observer can experience motion as long as it is neither too slow nor too fast. Motion perception, changes with time and object's vertical positioning [Pal99]. Experiments show that the threshold for motion perception depends on many factors, including the duration of the motion and whether the moving object is seen alone or against a background of stationary objects. A single light moving in the dark for a long period of time has a threshold for motion perception of 1/6 1/3° of visual angle per second [Aub86]. The same light moving against a textured background has a threshold for perceiving it as moving: about 1/60° of visual angle per second. This lower threshold for object-relative motion indicates that the visual system is much more sensitive to the motion of one object relative to another than it is to the motion of the same object relative to the observer. With short-duration motion (250 ms or less), the thresholds are higher, and the presence of a stationary reference does not change the motion threshold [Lei55] [Pal99]. The motion itself appears to slow down, as adaptation occurs when an observer stares for a prolonged period at a field of image elements moving at a constant direction and speed. Even motion after-effects can occur. A temporal contrast sensitivity function plot [DeL58] shows how flickering varies with contrast and vice versa. The eye appears to be most sensitive to a frequency of 15 to 20 Hz at high luminance (photopic vision). At photopic light levels, less than 1% contrast is required to detect the stimulus, the high temporal frequency cut-off is close to 60 Hz. At low light levels the maximum contrast is about 20% and the cut-off is around 15 Hz. To detect flicker of high frequencies, maximum contrast is required. Temporal resolution is not efficient at low luminance. Increasing the background illumination has different effects on the relative sensitivity for low and high temporal frequencies [Kel61].

3. UNSYNCHRONIZED MOTION

In our virtual prototyping of MEMS, we require very high and very low speed unsynchronized motion dynamics for virtual reality simulations. We use simulated stroboscopic illumination to display simultaneously the very fast (up to 500 Hz) and very slow (down to 10 Hz) moving objects on the screen [LS03a] [LS03b]. In this way, the relative motion of the objects is maintained. At the same time, it allows the designer to observe the MEMS being designed. For the stroboscopic simulations, we use two parameters: the Stroboscopic Illumination Interval (SII) and the Stroboscopic Flash Duration (SFD). The position P_n of the object after n stroboscopic cycles is calculated as $P_n = n * t_{SII} * f$, where f is the frequency of rotation of the object. The objects are shown at the intervals (SII) during the flash period (SFD). The SFD is used for the purpose of visibility, making the object in high speed visible at all. This is effectively filtering the images, and displaying only a subset at a rate such that they become observable, without sacrificing their relative movement.

In the displays, the type of movement and the shape of the object are critical e.g. a rotating gear at 400 Hz requires a different setting than a flicking cantilever. We performed a systematic combinatorial simulation experiment to find out what in what range of values of SII and SFD produce images that are comfortable to the eyes. We found that the SFD is best between 0.2 to 1 s, and the SII between 0.02 and 0.1 s. For a non-flickering display, both times, the SII and the SFD depend on the computer screen hardware and on the observer's own perception. However, we want to find default display settings, which on average would give reasonably good visualization, and which can be modified according to personal perception with an on-screen mouse activated sliding dial. To obtain a good stroboscopic effect, when the SII becomes longer, the SFD should become a bit longer too. For a high speed, the SII should be rather longer, but for

low speed, it should be rather shorter. This is shown in Figure 1 as the combinations of stroboscopic parameter settings that are suitable for different combinations of fast-fast, fast-slow, and slow-slow moving sets of objects.

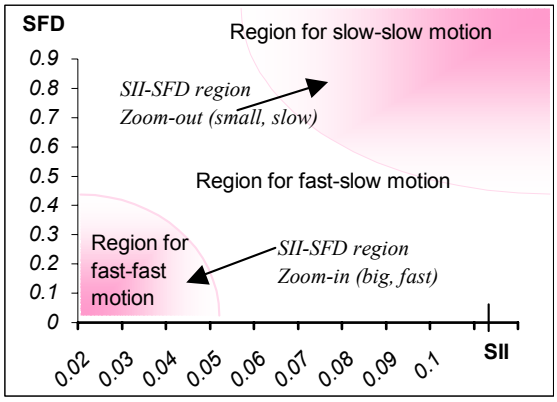


Figure 1 Suitable combinations for SII and SFD.

We use a scaling function to bring the natural speed of the fast and slow objects to be displayed together, into the range of SII and SFD where a good display is warranted, and display the animation at these ranges' mid value. This is the default we are looking for. Alternative personalized values can then be selected by the user via a 2D slider.

4. WHAT TO SCALE AND HOW TO SCALE

Stroboscopic simulation is suitable for very slow and very fast objects shown simultaneously. When objects move at similar speeds, simple down scaling, to an observable speed can be used. We use the following scaling function:

$$SV = \frac{DV - md}{MD - md} \times (UppS - lowS) + lowS \qquad \text{Eq 1}$$

DV and SV are the actual data and the scaled value respectively. MD and md are the maximum and minimum data values respectively. UppS and lowS are the upper and lower scaled boundary, they can be the visual system's threshold in size/velocity or the limits of computer capability.

It is difficult to display very big and small objects, rich in details, moving at different speeds because it is hindered by the threshold in the visual system or the computer capability. Too small/slow and too large/fast objects are not visible to humans. A nice list of perception threshold values has been compiled at Stanford University [Sta04]. A person can resolve a size of 0.15 mm at 50 cm distance, but the monitor cannot display it. The smallest size an ordinary monitor can display is one pixel, around 0.3 mm. To avoid eyestrain we use three pixels as our smallest dimension for MEMS structures. Scaling must be done equally in both directions to avoid distortion. We can only use only the shorter of the two measures height and width of the screen, but there are exceptions, as we shall see

5. SCALING MICRO-FLUIDIC FLOW

We show two examples of scaling in this paper, a micropump in operation flicking its membrane, and a microchannel showing the microfluidic flow.

Dimension	Original Size (µm)	Scaled size (pixels)
Membrane thickness	7.80	3
Membrane diameter	10000	421
Pump width	14000	588
Pump length	17500	734

Table 1 Scaled dimensions of the micropump.

Table 1 shows a subset of the critical dimensions. We need to scale such that the thinnest structure is just three pixels. Figure 2 shows a vertical section through the pump. The animated visualization shows the membrane flexing up and down at 50 Hz, and the valves opening and closing alternatively.

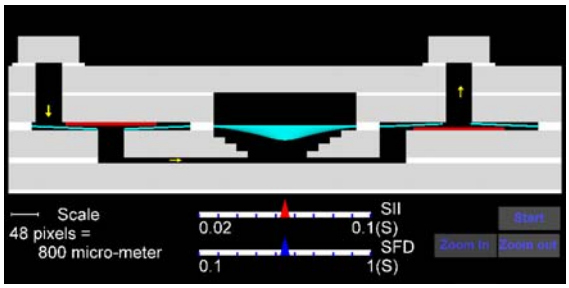


Figure 2 Vertical section through the pump.

For the microfluidic flow, we have a different pump [AS03] with a channel length of 4.8 cm, but with a diameter of just 152µm. The fluidic flow is made up three layers moving at different speeds, with the consequent formation of a bullnose. It is not possible to show the complete channel and the layers of the flow together. We must be above the threshold to see flows at the level of detail of layers. We need now five pixels for the thinnest layer (the external layer), as three are not enough for a good image and the formation of the bullnose. We can use screen *length* to display it, with still enough height to display the full height of the channel. The average flow rate (velocity) of 1.45e10, 4.35e10, and 8.30e10 µm/sec for each of the three layers in the circular channel are scaled to 11, 45 and 91 pixels/s respectively. There are several thresholds in perceiving dynamic images, however several factors, including color, depth, distance, size, texture, length of perceived duration, can affect this human threshold. Humans have a

field of view of about 200° horizontally and 135° vertically. A 17" computer display viewed at 50 cm distance spans roughly 37x30° [Red01]. Each pixel spans 1.7x1.7'of an arc. The upper limit for perceiving horizontal velocity comfortably is 91 pixels per second, or 2.67° per second; the lower limit is 11 pixels per second, or 0.33° per second.

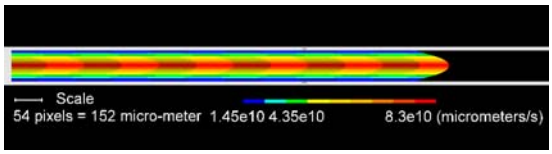


Figure 3 Microchannel showing fluid layers flowing at different speeds and bullet nose.

Our image is shown in 2-D as a cross section of the channel. The flows are parallel to the walls, moving at their relative flow velocity. We used a texture for each layer to show the flow. As the edge of each unit of the faster layer passes by the edge of the unit of a slower layer helps to see the relative motion between them. Experiments show that short tiles help to show slow movement, long tiles for fast movement. The arrangement of tiles influences the informative content and image, as shown in Figure 4. The virtual time unit is one second, as in real time.

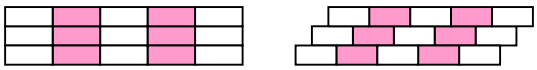


Figure 4 Textures for the fluidic flow

Each pump cycle forms a bullet nose, due to the higher speed in the internal layer, while the external layer is slowed down by friction with the channel walls. For the growth of the bullet nose the data were obtained by using Finite Element Analysis and simulated with ANSYS [AS03a]. To these resulting data a parabola was fitted, whose constant k is a varying function of the velocity, which in turn was used for the velocities of the fluidic flow.

The dynamic visual effects are affected by the distance of a viewer to the observed target. The further away the viewer is to the monitor, the faster animations can be observed, because at longer distance the visual degree becomes smaller. Visual systems are sensitive to light wavelengths from 400 to 700 nm [Fer01]. There is a smooth progression from violets, through blues, greens, yellows, oranges, and reds from shortest to longest wavelengths. We use color mapping to express the velocity of the flows: blue for low speed flows, red for high speed flows. We also use a progression of alternating shaded and non-shaded tiles, of equal length to visualize better the movement of the flow. The difference between shaded and non-shaded tiles is necessary for displaying the relative speed of flows of different layers.

6. CONCLUSIONS

In this paper we have analyzed and presented the scaling of size and speed of moving objects, for MEMS CAD Virtual Prototyping. Microtechnology images pose different challenges than large machine displays due to different dominating physical forces. The visualizations for such an environment must be truthfully scaled and undistorted, because critical dimensions of MEMS are vital to their good functioning. At the same time the moving VR images must be flicker free and pleasant to avoid eyestrain to the MEMS design engineer.

7. REFERENCES

[TM04] Tory M., Moller T., Human Factors in Visualization Research. IEEE Transactions on Visualization and Computer Graphics, Vol. 10, No. 1, pp 72-84, 2004.

[KS04] Kjelldahl L., Sousa Santos B., Visual Perception in Computer Graphics Courses. Computers & Graphics, 28 , pp 451-456, 2004.

[Pal99] Palmer S.E., Vision Science: Photons to Phenomenology, The MIT Press, Cambridge, Mass., 1999.

[LS03a] Li Z., Sitte R. Modelling Unsynchronized Motion Dynamics for Truthful VR Visualizations Proc. International Congress on Modelling and Simulation, Townsville, Australia, D.A. Post (Ed.), pp 1910-1914, 2003.

[LS03b] Li Z., Sitte R . Simulated Stroboscopic Illumination for Unsynchronized Motion Dynamics. Proceedings of the European Simulation and Modelling Conference ESMc, Naples, pp 299- 303, 2003.

[Sta04] http://white.stanford.edu/html/numbers/ numbers.html, December 2004

[Red01] Reddy M., Perceptually Optimized 3D Graphics, IEEE Computer Graphics and Applications, 21 (5), pp 68-75, 2001.

[AS03a] Aumeerally M., Sitte R., Modelling of Layered Fluid Flow in a Circular Microchannel. Proc.of the European Simulation and Modelling Conference ESMc, Naples, pp 458-462, 2003.

[AS03b] Aumeerally M., R.Sitte, Simulation and Modelling of Layered Fluid Flow in a Rectangular Microchannel. Proc. of the Middle East Symposium on Simulation and Modelling MESM, Dubai, pp 81 – 85, 2004

[Fer01] James A. Ferwerda. Elements of Early Vision for Computer Graphics. IEEE Computer Graphics and Applications, Vol.21, No.5, pp. 22-33, 2001

Towards real-time body pose estimation for presenters in meeting environments

Ronald Poppe Dirk Heylen Anton Nijholt Mannes Poel
University of Twente
Department of Computer Science
Human Media Interaction Group
P.O. Box 217, 7500 AE, Enschede, The Netherlands
{r.w.poppe, d.heylen, a.nijholt, m.poel}@ewi.utwente.nl

ABSTRACT

This paper describes a computer vision-based approach to body pose estimation. The algorithm can be executed in real-time and processes low resolution, monocular image sequences. A silhouette is extracted and matched against a projection of a 16 DOF human body model. In addition, skin color is used to locate hands and head. No detailed human body model is needed. We evaluate the approach both quantitatively using synthetic image sequences and qualitatively on video test data of short presentations. The algorithm is developed with the aim of using it in the context of a meeting room where the poses of a presenter have to be estimated. The results can be applied in the domain of virtual environments.

Keywords

3D human pose estimation, blob tracking, silhouette matching, avatar animation

1. INTRODUCTION

Being able to recognize gestures or body poses is a key issue in many applications. Human pose estimation based on computer vision principles is a technology that suits the requirements for an inexpensive, widely applicable approach to human pose estimation. We identify five application areas:

1. **Surveillance**. A subject is tracked over time and monitored for special actions, for examples monitoring a parking lot to detect thieves.
2. **Human-computer interaction**. The captured poses are used to provide controlling functionalities. Examples are the control of a crane or a computer game.
3. **Analysis**. Analysis of captured human poses can help understanding movement in clinical studies.
4. **Annotation**. Automatic annotation of video data is useful for automatic indexing. An example is annotation of meeting video data.
5. **Animation**. Body poses are used to animate avatars in virtual environments.

WSCG SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

In this paper we describe a real-time human pose estimation algorithm that uses low resolution images from a single camera. The approach is suitable for the estimation of body poses in an indoor setting. We have developed it with the aim of using it in the context of a meeting room where the poses of a presenter have to be estimated. The results of the system can also be used in gesture recognition applications. Tracking of multiple people is not supported. Currently the body must be facing the camera at all times. This limitation is realistic for presenter behavior recognition and for human-computer interaction (HCI) purposes.

Related Work

Human motion tracking has received a significant amount of attention from the computer vision research community in the last decade. Many pose estimation approaches use images from multiple cameras to obtain more accurate 3D data. However, a broader range of applications can benefit from a single-camera approach. An overview of computer vision-based human motion capture approaches has been made by Moeslund and Granum [Moes01].

Features are used to describe the subject in the scene in a convenient way. A variety of features has been used in previous approaches, among others colors, edges, silhouettes and textures. Silhouettes are often used because they are quite robust and do not contain ambiguous information such as inner edges. Silhouettes are used in multi-view settings, for example by Kakadiaris and Metaxas [Kaka96]. They recover all the body parts of a subject by monitoring the changes over time to the shape of the deforming silhouette. Delamarre and Faugeras [Dela99] also use silhouette contours in a multi-camera setting. They use a variation of the Iterative Closest Point algorithm to fit a 3D human model inside the silhouettes. A manual initial estimation is needed. Mittal and others [Mitt03] use silhouette decomposition and are able to estimate the poses of multiple persons in cluttered scenes viewed from multiple angles. Sminchisescu and Telea [Smin02] use monocular image sequences and a similarity measure that uses attraction and area overlap terms on smoothed silhouettes. In recent monocular work from Agarwal and Triggs [Agar04] a sparse Bayesian regression method is used to estimate 3D body poses directly from silhouettes. They are able to estimate angles for a 54 degree of freedom (DOF) human body model. Our algorithm uses monocular video sequences and is able to work in real-time, without a complex body model and without relying on previously learned motion patterns.

The pose estimation process estimates the human pose from these features. The output of a pose estimation process is usually a set of angles for each DOF. A DOF corresponds with a single joint rotation.

In the remainder of this paper, we discuss feature extraction and pose estimation (Section 2 respectively Section 3), experiment setup and obtained results (Section 4), and conclusions and future work (Section 5 respectively Section 6).

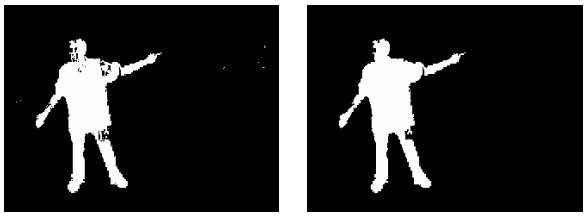
2. FEATURE EXTRACTION

We used silhouettes to represent the subject in the scene. Simple background subtraction was applied, using a pre-recorded image of the background. Lighting compensation was used to correct small variations in light intensity. The algorithm is computationally fast and performs reasonably well for indoor situations. The next step in the extraction process is removal of shadow. Shadow detection was carried out in HSV color space. The saturation channel (S) was ignored since it was not a good predictor.

A novel approach to noise removal using a contour tree was used. All nodes in the contour tree except the root node have a parent which is the containing contour in the image. Since a maximum of one person was allowed in the scene and the human body is rigid, only the largest contour on the first level in the tree needed to be examined. To construct the tree, contours were extracted using an efficient region growing technique. The contour tree algorithm proved to yield better results than traditional methods such as erosion and dilation. The algorithm is computationally fast and can easily be adapted to work with other assumptions.

The results of the image subtraction, shadow detection and noise removal process are shown in Figure 1.

(a) Original image (b) Image subtraction



(c) Shadow removal (d) Noise reduction

Figure 1: Silhouette extraction process example.

Skin Color Detection

Skin color was extracted from the silhouette, using the hue and saturation channel in the HSV color space. An upper and lower threshold were set. On a set of 10 images with hand-labeled skin regions, the precision was nearly 85 % at a recall rate of 95 %. Noise was removed from the results of the skin color detection with the contour tree algorithm described in the previous section. This resulted in several skin color blobs. Results of the skin color detection before and after noise removal are shown in Figure 2. Using color can cause robustness problems. But since only the largest skin color contours are evaluated, very few limitations on clothing color have to be imposed.

(a) Skin color detection (b) Noise removal

Figure 2: Skin detection process example.

3. POSE ESTIMATION

We used the silhouette and skin region features as input for the pose estimation process, which produces a set of joint angles for each degree of freedom in the human body model. This model is described in the next section.

Human Body Model

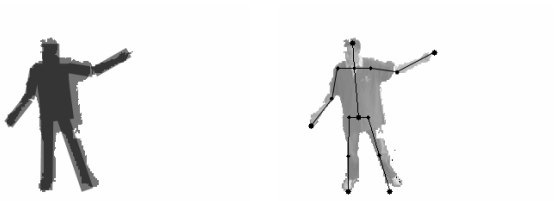
Various body models have been proposed in literature. Some contain a small number of joints or focus on a part of the human body such as the upper body or an arm. The model that is used here is derived from the H-anim standard which models the human body as a hierarchic skeleton composed of joints and rigid segments. The simplified model contains 14 segments that are modeled as cylinders. The 10 joints have a total of 16 DOF, three for each hip and shoulder and one for each knee and elbow. Because of the low resolution of the image frames the rotation of the hands and feet is not modeled. The model contains segment lengths and joint angle limitations.

Calculating 3D Positions

The distance of the person to the camera can be calculated by looking at the bottom of the silhouette. This distance was used to calculate the 3D locations of the head and hands. The skin regions were tracked using a simple and fast tracking algorithm. A minimum frame rate of approximately 10 frames per second is necessary to track fast hand movements. Likely positions of the hands and head were used to get an initial labeling. Since the body always had to face the camera, shoulders and hips were in the same plane. The 3D location of a hand can lie on a line through the camera center and the obtained coordinate in the shoulder-hip plane. Different values for this location were evaluated in the inverse kinematics process.

Inverse Kinematics and Silhouette Matching

An analytical inverse kinematics approach was used to compute the location of the elbows and knees and to calculate the rotations of all DOF. Different values for the hand location and the elbow position were evaluated for the arms and legs. Joint angle limits were applied to reduce the search space. The projection of the human body model was matched against the extracted silhouette. Since the estimation with the best silhouette match is not always the most likely body pose, the previously estimated pose was used to generate a smooth estimation of the movement over time.



(a) Silhouette matching (b) Pose estimation

Figure 3: Silhouette matching and pose estimation example.

4. EXPERIMENTAL RESULTS

Tests with a camera have been carried out to evaluate the system's performance. Since no ground truth about joint angles could be inferred from these data, quantitative experiments were carried out on movie sequences generated by Curious Labs Poser.

System Implementation

The system uses AVI movies or camera streams at a resolution of 320 × 240 pixels. Tests were carried out at a frame rate between 15 and 25 frames per second.

Tests with Poser Movies Sequences

Four movies with a total of 240 frames were generated with frame rate of 15 frames per second. Sample frames of the four movie sequences are depicted in Figure 4. To evaluate the results, the angles for each of the 16 DOF were measured and compared to the estimated angles. Two body models were evaluated, one with default segment lengths and one with 10 % shorter limbs. Shortening the segments yielded better results for the hip and knee rotations but slightly worse results for the shoulder and elbow rotations. The average and standard deviation of the joint angle errors are summarized in Table 1.

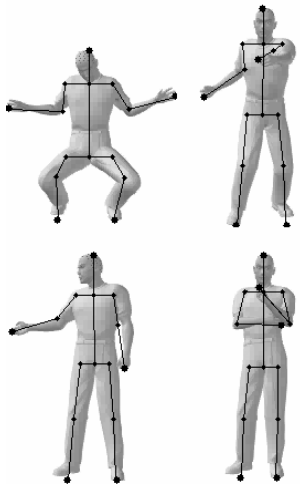


Figure 4: Frame 15 of each of the four synthetic test movies with superimposed skeleton.

The results can be explained by the human body model used. If the segment length in the model does not match the actual segment length, miscalculations occur. One of the causes can be found in the matching method. When arm segment lengths are shortened, the elbow angle increases.

Tests with Camera

Qualitative tests using a camera were carried out on five short presentations by graduate students. The presentations were between two and five minutes long and dealt with graduation topics. The presentations contained typical presenter movements such as pointing to the whiteboard. Few errors were made by the tracker and the pose estimation process. Visualization showed smooth movements. However, dropping the frame rate from 25 to 5 frames per second caused the multiple blob tracker to fail.

Rotation	Default		Shortened	
	Avg	SD	Avg	SD
L. sh. flexion	8.64	6.70	11.44	9.14
L. sh. rotation	0.72	0.61	3.49	0.78
L. sh. adduction	8.95	3.56	4.78	2.27
L. el. flexion	7.17	3.61	7.70	5.85
L. hip flexion	1.05	0.20	0.97	0.21
L. hip rotation	2.11	1.60	1.02	0.34
L. hip adducution	0.97	1.93	0.78	1.19
L. knee flexion	11.97	3.91	8.72	3.99
R. sh. flexion	11.83	8.70	16.37	7.51
R. sh. rotation	4.12	1.61	2.50	1.37
R. sh. adducution	14.58	8.85	14.68	7.23
R. el. flexion	9.56	3.32	13.73	7.25
R. hip flexion	0.92	0.92	0.97	0.87
R. hip rotation	3.78	1.79	1.58	0.34
R. hip adduction	0.83	1.70	0.65	1.24
R. knee flexion	10.51	1.23	7.21	3.44
Average	**6.11**	**3.14**	**6.04**	**3.31**

Table 1: Average and standard deviation of joint angle error distance in degrees for the synthetic test movies with default and shortened limbs.

5. CONCLUSION

In this paper we discussed a pose estimation approach that is computationally fast thanks to efficient algorithms such as a novel noise removal algorithm. Silhouette matching is used to match a projection of a 16 DOF human body model to the extracted silhouette. The system works reasonably well, with an average joint angle error distance below 10 degrees. The system is suitable for the estimation of presenter poses in meeting environments.

6. FUTURE WORK

Future work will aim at allowing multiple persons and removing the limitation that the person has to face the camera. This will make the approach suitable for automatic pose estimation not only for presenters but also for meeting participants. Figure 5 shows a virtual meeting room with participants. Extending the approach with gesture recognition will allow for recognition of meeting acts.

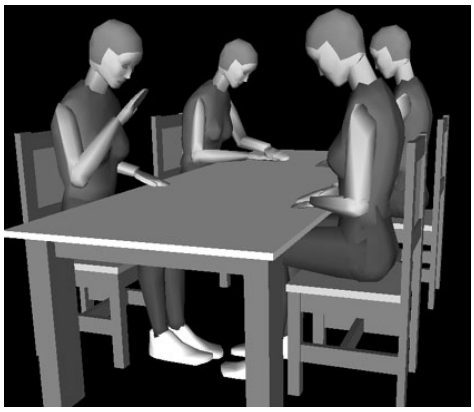


Figure 5: Virtual meeting room.

7. ACKNOWLEDGEMENTS

This work was partly supported by the European Union 6th FWP IST Integrated Project AMI (Augmented Multi-party Interaction, FP6-506811, publication AMI-12).

8. REFERENCES

[Agar04] A. Agarwal and B. Triggs, "Learning to track 3D human motion from silhouettes," in Proceedings of the 21st International Conference on Machine Learning (ICML), July 2004.

[Dela99] Q. Delamarre and O. Faugeras. 3D articulated models and multi-view tracking with silhouettes. In International Conference on Computer Vision: ICCV 1999, Corfu, 1999.

[Kaka96] I.A. Kakadiaris and D. Metaxas, "Three-dimensional human body model acquisition from multiple views," IJCV, vol. 30, no. 3, pp. 191–218, December 1998.

[Mitt03] A. Mittal, L. Zhao, and L.S. Davis, "Human body pose estimation using silhouette shape analysis," in Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, July 2003, pp. 263–270.

[Moes01] T.B. Moeslund and E. Granum, "A survey of computer vision-based human motion capture," Computer Vision and Image Understanding: CVIU, vol. 81, no. 3, pp. 231–268, 2001.

[Smin02] C. Sminchisescu and A. Telea. Human pose estimation from silhouettes, a consistent approach using distance level sets. In WSCG International Conference on Computer Graphics, Visualization and Computer Vision, 2002.

Multimodal Interaction Metaphors for Manipulation of Distant Objects in Immersive Virtual Environments

Frank Steinicke
WWU Münster
Institut für Informatik
Einsteinstraße 62
48149 Münster, Germany
fsteini@uni-muenster.de

Timo Ropinski
WWU Münster
Institut für Informatik
Einsteinstraße 62
48149 Münster, Germany
timo@math.uni-muenster.de

Klaus Hinrichs
WWU Münster
Institut für Informatik
Einsteinstraße 62
48149 Münster, Germany
khh@uni-muenster.de

ABSTRACT

In this paper we discuss direct interaction metaphors for selection and manipulation of distant objects in immersive virtual environments and we propose extensions of the improved virtual pointer (IVP) metaphor. In particular, we describe how the process of object selection with the IVP metaphor can be enhanced by modifying the distance calculation used to determine the closest object to be selected. Furthermore we introduce direct 6 DOF manipulations of virtual objects using the IVP metaphor. We demonstrate how the task of object selection can be improved by combining existing interaction metaphors with multimodal feedback.

Keywords

Virtual Reality, Interaction Techniques, Multimodal Interaction Metaphors

1. INTRODUCTION

Virtual environments (VEs) have shown considerable potential as an intuitive and natural form of human-computer interfaces. Many scientific application areas benefit from virtual reality (VR). To improve the acceptance of VR technologies, the most basic interaction techniques need to be optimized to enable efficient human-computer interaction (HCI).

In this paper *direct interaction* metaphors for selection and manipulation of both local as well as distant objects in VEs are discussed and evaluated. In direct interactions the user directly manipulates objects with the input device, whereas in indirect interactions the user performs changes indirectly using menus, icons or widgets.

WSCG 2005 SHORT papers proceedings,
ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

Before directly interacting with virtual objects, the user needs to specify the target for the interaction from the set of *selectable* objects. After *selecting* a virtual object, the user may manipulate any of the object's attributes, e.g., change the color or add a texture. In this paper we focus on six *degree of freedom* (DOF) manipulations, i.e., changing position and orientation of the virtual object.

Many VR application areas have shown that virtual pointer metaphors are natural and require less effort for both local and distant direct object interaction [Bow97a]. Even though virtual pointer metaphors can be used intuitively, their way of aiming at virtual objects and performing 6 DOF manipulations needs to be improved.

In order to achieve these goals we have proposed the *improved virtual pointer* (IVP) metaphor [Ste04a], which avoids most of the aforementioned disadvantages of current interaction metaphors. Our approach allows the user to select a desired object without requiring an exact hit. A straight ray is used to indicate the direction of the virtual pointer, while an additionally visualized bendable ray points to the closest selectable object (see Figure 1).

The closest selectable object which would be chosen if the user would perform a selection (e.g., by press-

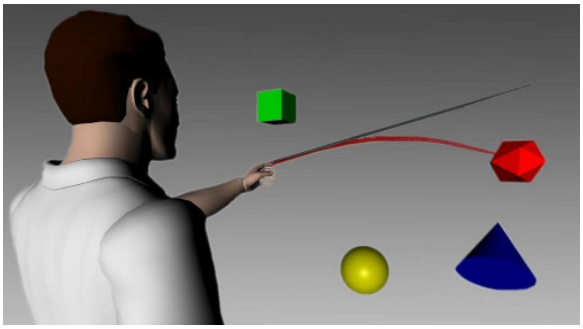


Figure 1: Illustration of the IVP metaphor.

ing a button or by pinching a glove) is called *active object*. After selecting the active object, successive manipulations can be accomplished.

In this paper we extend the previously introduced IVP metaphor by specifying a modified mechanism to determine the closest selectable object and proposing concepts how to use the IVP metaphor to perform manipulations of distant objects. Furthermore we give examples how the interaction process can be extended by giving adequate multimodal feedback.

2. IVP METAPHOR EXTENSIONS

In this section we present extensions of the IVP metaphor concerning selection and manipulation of virtual objects.

Selection of Virtual Objects

With the IVP metaphor a user performs a direct object selection by roughly pointing at the desired object. Thereupon the flexible ray bends to this object if it is the one closest to the straight ray. Determination of this active object is a major task of the IVP metaphor and is achieved by computing the distances of all selectable objects to the virtual ray. The scene graph structure used in most computer graphics systems can be exploited to enhance the performance by calculating all distances during a pre-evaluation phase of the scene graph. Thus interactive frame rates are maintained when using the IVP metaphor. The results of the distance calculations are stored in an ordered list, called the *ActiveObjectList*. This list contains all selectable objects and their distances to the virtual ray; the entries are sorted with respect to increasing distance, the first object with minimal orthogonal distance is the active object. This list provides the possibility to switch between active objects. Thus difficulties occurring during the selection of partially or fully occluded objects can be solved.

2.1.1 Distance Calculation

The distance between the virtual ray and a selectable object may refer to different *reference points* of a virtual object, e.g., the center of its bounding box or the closest edge or vertex. To compute the world coordinate distance between the reference point of a selectable object and the ray, we consider the line perpendicular to the ray which connects the reference point and the ray. Sorting the objects on the basis of this world distance within the ActiveObjectList may result in disadvantages when using perspective projection since the displayed distance may be distorted.

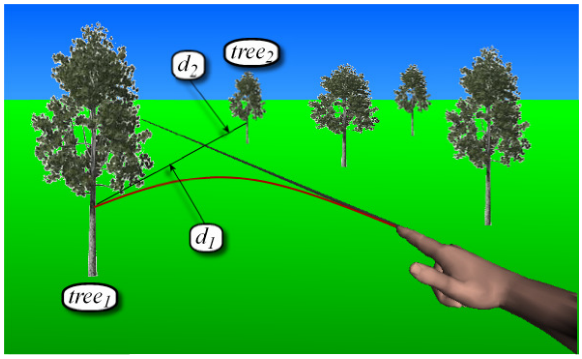


Figure 2: Problem of distant object selection caused by perspective distortion.

Figure 2 illustrates this problem. From the user's point of view, $tree_2$ seems to be closer to the ray. However, $tree_1$ attracts the curve and gets active since the distance d_1 between $tree_1$ and the virtual ray is less than the distance d_2 between the ray and $tree_2$, even though d_1 seems to be larger because of the perspective distortion. To prevent this drawback, we introduce two different approaches.

2.1.2 Image Plane Approach

An obvious approach is to evaluate the distances used for sorting the objects in the ActiveObjectList in image space coordinates. Therefore each world space distance vector d_i is transformed into the corresponding image space distance vector id_i as illustrated in Figure 3.

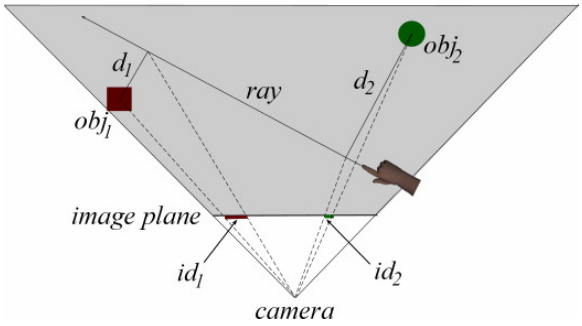


Figure 3: Projection of world space distance vectors onto the image plane.

The resulting lengths of the image space distance vectors are compared, and the objects are sorted accordingly into the ActiveObjectList. A world space distance vector appears shortened after the projection, although it may be quite long in world coordi-

nates. Figure 3 illustrates this characteristic of projective geometry. Although the object obj_2 is located farther from the virtual ray than obj_1, its projection id_2 is clearly shorter than id_1. Thus sorting the ActiveObjectList based on image space distance vectors may lead to inconvenient results.

2.1.3 Distance Scaling Approach

A better approach considers both the world space distance between a particular object and the ray as well as the distance between the virtual input device and the object.

The world space distance is multiplied by a factor s, and the resulting value is used to sort the objects stored in the ActiveObjectList. The factor s is the inverse of the length of the vector from the position of the virtual input device to the position on the ray of the orthogonally mapped reference point of the considered object. Hence, with increasing distance between the virtual input device and a virtual object the value of s decreases, and multiplying the world space distance with this factor yields a smaller value as basis for sorting the ActiveObjectList. Analogously a decreasing distance between input device and object leads to a larger s and therefore a larger value is used for the comparison.

Although both approaches use different values for the sorting process, access to the distances in world space needs to be guaranteed, e.g., to measure distances between virtual objects.

Manipulation of Virtual Objects

For the manipulation of virtual objects we extend the idea of the HOMER technique described in [Bow97a] to provide an intuitive and natural alternative for 6 DOF direct manipulations. In contrast to the HOMER technique, the virtual input device and the selected object both remain at their initial positions after a selection is performed. Afterwards rotations and translations are accomplished as described in the following subsections.

2.1.4 Rotation

Although the virtual input device and the desired object both remain at their initial position, all rotations are implemented by a one-to-one mapping between the rotational movements of the virtual input device and the object. First the manipulated object is translated to the origin, then the rotational components of the transformation matrix of the virtual input device are copied to the corresponding components of the transformation matrix of the object. After finishing all rotational manipulations the object is moved back to its initial position or to a modified position according to the translation carried out during the manipulation process. Thus rotations are applicable in a natural and intuitive way like real world rotations, except that the manipulated object remains at its original position without being relocated into the user's hand. By using this approach manipulations can be accomplished accurately without occluding the desired object by the virtual input device, as it may happen when using the HOMER technique.

2.1.5 Translation

During manipulation of distant virtual objects, perspective distortion may cause a user to perceive the translation of such an object as distorted when compared to the movement of the virtual input device. Therefore, for the translation of a distant virtual object we scale its translational movement by a linear mapping function.

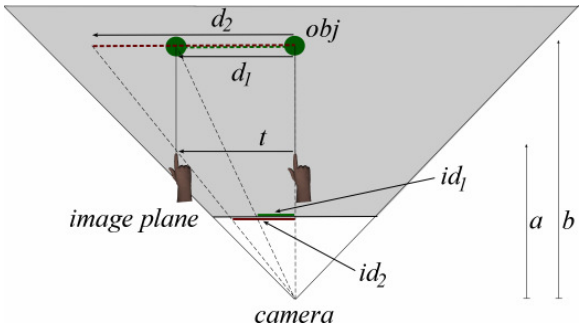


Figure 4: Distant object translation affected by perspective distortion.

Figure 4 clarifies this issue. After the user selects the object *obj*, a translation vector t of the virtual input device is mapped to this object. The projection vector id_1 of the one-to-one mapped translation d_1 of this translation vector t appears curtailed on the image plane (compared to id_2, which is the projection of d_2). Users anticipate that a translation vector t of the virtual input device would result in the scaled translation vector d_2 corresponding to id_2 on the image plane. To obtain the longer vector d_2, all translational movements are scaled with the factor $f = b / a$ where a is the distance between the camera and the virtual input device and b is the distance between the camera and the selected object (see Figure 4). The translation vector t is scaled by the value f and applied to the selected object. Indeed, small and accurate translation of distant objects is complicated by this approach, but existing VR applications have revealed that precise manipulations are accomplished primarily by local interaction within the immediate reach of the user. Direct 6 DOF translations of distant objects are mostly used for moving objects close to the user for exploration or for performing larger translations.

3. MULTIMODAL INTERACTIONS

As described in [Rai99a], multimodal interactions have the potential to enhance HCI and support the

user during the manipulation process [Ric94a]. In this section we will describe the adaptation of multimodal interaction concepts to fit the needs of object selection and manipulation in immersive VEs.

Multimodal Input

To improve object selection two-handed interaction can be used. In our responsive workbench environment we use a pinch glove in combination with a haptic input device, which is shown in Figure 5. Position and orientation of both devices are tracked using an optical tracking system. The haptic input device is used to control the virtual input device with the IVP metaphor, simply by pointing at the desired object as described above. The pinch glove is used by the non-dominant hand for accessing menus or to assist the user when performing an object selection, e.g., selection of occluded objects by *tabbing* through the ActiveObjectList.

Multimodal Output

We use multimodal feedback to inform the user about a possible selection, i.e., visual, acoustic and tactile senses are addressed.

To get an adequate visual feedback of a possible selection, we visualize the ray direction vector as well as a beziér curve graph. This curve bends to the active object as described in [Ste04a] and illustrated in Figure 1.

To improve the user's perception of the active object's position, we inform the user acoustically when the active object is changed. By moving the virtual input device a different selectable object gets active, i.e., the beziér curve bends to the new active object, the position and orientation of both the active object and the user are used as parameters for the sonification process. Thus a change of the active object can be emphasized by a gentle sound dispersing from the position of the active object towards the user's position. As a result, the active object can be spatially located more easily.

In addition to the visual and acoustic cues, we added haptic feedback, i.e., the user gets haptic information regarding the active object. During a change of the active object, the user receives a light and short vibration signal emitted by the haptic input device (see Figure 5). Since the signal is emitted using a BlueTooth® connection, no cables constrain the interaction process. The level of vibration can be altered depending on the distance between the virtual input device and the active object. Starting from an initial minimal level of vibration, a decreasing distance between input device and active object results in a higher level of vibration.

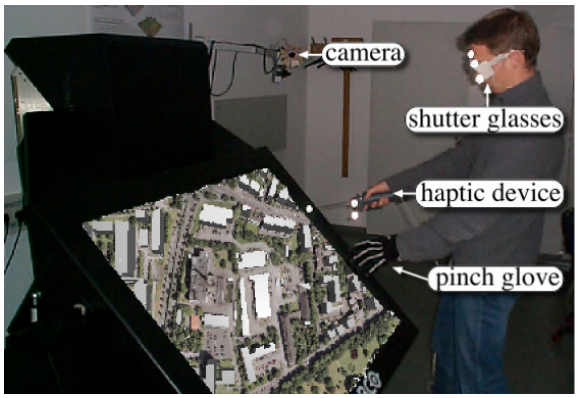


Figure 5. Workbench environment with an optical tracking system.

4. CONCLUSION AND FUTURE WORK

For evaluation we have compared the IVP metaphor to some of the techniques described in [Bow97a]. Our tests have shown that object selection can be accomplished faster with the IVP metaphor, especially when using the scaled distance approach. In a survey users have evaluated the use of multimodal feedback as very helpful, especially the haptic feedback given when a new object becomes active. The questionnaires as well as the detailed results of the user study are available upon request.

To further improve usability of direct interaction metaphors in general appropriate combinations of constraints restricting the available DOF may be useful. Therefore we will examine the possibilities of controlling 3D widgets with the IVP metaphor.

5. REFERENCES

[Bow97a] Bowman, D., and Hodges, L. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In Symposium on Interactive 3D Graphics, pp. 35-38, 1997.

[Rai99a] Raisamo, R. Multimodal human-computer interaction: A constructive and empirical study. PhD thesis, University of Tampere, 1999.

[Ric94a] Richard, P., Burdea, G., Gomez, D., and Coiffet, P. A comparison of haptic, visual and auditive force feedback for deformable virtual objects. In Proceedings of the Conference on Artifical Reality and Teleexistence (ICAT94), pp. 49–62, 1994.

[Ste04a] Steinicke, F., Ropinski, T., and Hinrichs, K. Object selection with an improved virtual pointer metaphor. To appear in Proceedings of International Conference on Computer Vision and Graphics (ICCVG), 2004.

Approximating Real-World Luminaires with OpenGL Lights

Georg Zotti Attila Neumann Werner Purgathofer
Institute of Computer Graphics and Algorithms
Vienna University of Technology, Austria
{gzotti,aneumann,wp}@cg.tuwien.ac.at

ABSTRACT

Dynamic illumination in real-time applications using OpenGL is still usually done with the classical light forms of point lights, directional lights and spot lights. For applications simulating real-world scenes, e.g. architectural planning, finding parameter sets for these simple lights to match real-world luminaires is required for realistic work. This paper describes a simple approach to process a luminaire data file in IESNA IES-LM63-95 format to create an approximation using at most 2 OpenGL lights to represent one luminaire.

Keywords
Real-world luminaires, OpenGL, interactive illumination planning

1. INTRODUCTION

Many rendering systems still use very simple light sources for illumination, typically point, directional and spot lights. In global illumination renderering systems, idealized area light sources are also commonly found. But, apart from the Sun, often modelled as directional light for terrestrial scenes, and special spotlights mostly used in theatres, most real-world luminaires as built by the illumination industry have a more complex radiation distribution, and in some global-illumination rendering systems photometric data files provided by the illumination industry can be used.

Still, for OpenGL-based real-time rendering, direct illumination is mostly performed with these simple lights (point, directional, spot), which are fast to evaluate and are built into the OpenGL standard.

In this paper, we want to demonstrate how to process data provided by the illumination industry to create a combination of at most two OpenGL lights which achieves an acceptable approximation of the luminaire for use in a real-time rendering context. The result however depends on the easiness with which a luminaire data set can be approximated with these simple lights.

WSCG 2005 SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. DEFINITIONS AND RELATED WORK

For simulation of real-world light sources in computer graphics, their radiance distribution has to be known. Most real-world light sources are area light sources, so accurate simulation should take also the shape of the luminaire and variations of the light distribution along its surface into account. A method by GOESELE et al. [GGHS03] for acquisition of *near-field photometry* data of light sources and rendering of them with high accuracy in a global-illumination renderer yields good results, but at high computational expense.

However, in industrial practice still only *far-field photometry* data are available almost exclusively, and a common rule of thumb allows its application in cases where the luminaire's distance is at least five times greater than its greatest extension [Ash01]. Here the luminaire is modeled as point light source, and its radiation distribution is given as a set of candela measurements in a regular grid of angular values, either printable as polar 2D *goniometric diagram* along fixed planes or displayable as *photometric solid* [Ash99], where distance from the center represents the luminaire's brightness into a certain direction.

A good introduction into the complex field of illumination for industrial applications is presented in [Ash01]. A method of reconstruction of directionally dependent light sources from goniometric data for a global illumination rendering system can be found in [AP03], who also cite more papers on this topic.

Usage of luminaire description files is especially useful in fields like architectural modelling, where architects can simulate the placement and orientation of their light sources in buildings during the planning phase, e.g. in the well-known global illumina-

tion renderer Radiance [LS98]. Here, long rendering times usually make interactive display rates impossible. [WAL*97] presents a real-time simulation of *static* non-diffuse illumination by precomputing up to 8 directional "virtual lights" per specular object.

However, a simple, yet good approximation of a real-world luminaire with known goniometric data by combination of standard OpenGL lights for the purpose of *interactive illumination* in a real-time rendering context appears not to have been done yet.

3. APPROXIMATION OF GONIOMETRIC DATA

IES-LM63-95 [IES95] defines 3 types or variants of polar coordinates in which goniometric measurements can be given, depending on the luminaire application. In this paper, we will concentrate on Type C photometry data, which are typically used for luminaires for architectural and outdoor roadway lighting. The luminaire data are given as a series of candela measurements in a polar coordinate system. Vertical angles can be limited to $0^\circ \ldots 90^\circ$ (lower hemisphere only) or $90^\circ \ldots 180^\circ$ (upper) or cover all $0^\circ \ldots 180^\circ$. Similarly, only a subset of horizontal angles may be given in case of rotational or lateral symmetry.

Our task can be described as follows: For any IESNA Type C luminaire description, build a simple representation using at most 2 OpenGL lights [SWND04]. With only 8 light sources available in OpenGL implementations, this limitation had to be chosen, also because the number of light sources in a scene significantly influences rendering speed. Our work should be integrated into an existing scene graph system and run also on older, non-programmable graphics hardware, so exploiting programmable hardware was not required.

In this context, luminaires emitting light in one hemisphere can only be modeled with single spot lights. Except for highly collimated lights (e.g., searchlights), most real-world light sources show quadratic attenuation in far-field photometry context, so we can set linear and constant attenuation to zero. So, apart from a scaling factor w, only the cut-off angle θ_{cut} and the exponent p of the cosine can influence the difference between data and approximation.

If light data are given for the full sphere, we are trying a combination of a single spot directed upwards or downwards (in case the file just contains zeros for one hemisphere), a single point light, one point light and one spot directed up- or downwards, or a set of 2 spots pointing into arbitrary directions in opposite hemispheres. Here, also the orientation of the spots around vertical and horizontal axes may have to be taken into account.

Distance Metric

All fitting methods make use of some error function decribing a "distance" between the data set of the original (given) lightsource and the approximated lightsource, which in our case consist of one or more OpenGL lightsources. The goal of any search process is to find an approximation which has the smallest possible error value obtained with this error function. Thus the role of this error function is central, because it controls any optimisation, that is, it defines the meaning of the desired solution. So the sort of "distance" to be taken into consideration is crucial. In contrast to the Hausdorff metric used in [Ash99], we are interested only in pointwise radial differences, since the distance of two functions is conceptually different from the distance of their geometrical representations.

The Error Function

As a first step, and also for visualisation, the candela data given in the luminaire data file are scaled to relative "magnitudes" m_{lum} so that the largest candela value is scaled to $\max(m_{lum}) = 1$.

Then, for all directions given in the IESNA data file, a synthetic light which can be modeled directly by our OpenGL light combination is evaluated. This light consists of 3 parts, of which at most 2 may be active: a spotlight of relative brightness w_1 and cosine exponent p_1 pointing into the lower hemisphere with orientation angles θ_{ori1} from vertical down and φ_{ori1} around vertical axis, another spotlight of relative brightness w_2 pointing into the upper hemisphere with cosine exponent p_2, orientation angles θ_{ori2} from vertical up and φ_{ori2} around vertical axis, and a pointlight of relative brightness w_3. These partial lights are combined to a synthetic magnitude m_{approx} according to:

$$\begin{aligned} m_{approx}(\theta,\varphi) = {} & m_{down}(w_1,p_1,\theta_{cut1},\theta_{ori1},\varphi_{ori1},\theta,\varphi) \\ & + m_{up}(w_2,p_2,\theta_{cut2},\theta_{ori2},\varphi_{ori2},\theta,\varphi) \\ & + m_{point}(w_3) \end{aligned}$$

where e.g. $m_{down}(w_1,p_1,\theta_{cut1},\theta_{ori1},\varphi_{ori1},\theta,\varphi)$ describes the magnitude of the spotlight pointing into the lower hemisphere. To evaluate, we have to project the requested direction (θ,φ) along the orientation $(\theta_{ori1},\varphi_{ori1})$ of the OpenGL light, with $\theta_I = -90 + \theta,\ \varphi_I = \varphi,\ \theta_O = 90 - \theta_{ori1},\ \varphi_O = \varphi_{ori1}$:

$$\begin{aligned} x_I &= \cos\theta_I \cos\varphi_I & x_O &= \cos\theta_O \cos\varphi_O \\ y_I &= \sin\theta_I & y_O &= \sin\theta_O \\ z_I &= \cos\theta_I \sin -\varphi_I & z_O &= \cos\theta_O \sin -\varphi_O \end{aligned}$$

The projected cosine is $r = x_I \cdot x_O + y_I \cdot y_O + z_I \cdot z_O$, so

$$m_{down}(w_1,p_1,\theta_{cut1},\theta_{ori1},\varphi_{ori1},\theta,\varphi) = \begin{cases} 0 & \text{if} \quad r \le \cos\theta_{cut1} \\ w_1 \cdot r^{p_1} & \text{if} \quad r > \cos\theta_{cut1} \end{cases}$$

A similar evaluation can be done to compute $m_{up}(w_2, p_2, \theta_{cut2}, \theta_{ori2}, \varphi_{ori2}, \theta, \varphi)$.

m_{approx} represents a magnitude value just for one direction (θ, φ), where θ is counted from vertically downward ($\theta = 0°$) to vertically upward ($\theta = 180°$) and φ from the x-axis ($\varphi = 0°$) in a positive sense around the vertical y-axis. To estimate the total difference between this light combination and the measured data, we have to sum over all $n_\theta \cdot n_\varphi$ angular pairs (θ, φ) given in the IESNA data file:

$$err = \sqrt{\frac{\sum_{\substack{\text{vert. angles } \theta \\ \text{horiz. angles } \varphi}} \left(\frac{m_{approx}(\theta,\varphi) - m_{lum}(\theta,\varphi)}{m_{lum}(\theta,\varphi) + C} \right)^2}{n_\theta \cdot n_\varphi}}$$

Here, C is an arbitrary constant to avoid a division by 0 in case $m_{lum}(\varphi, \theta) = 0$.

Obviously, θ_{ori1}, φ_{ori1}, θ_{ori2}, φ_{ori2}, θ_{cut1}, θ_{cut2}, p_1 and p_2 are directly mapped to the orientation angles, cutoff angles and cosine exponents of the spotlights, if applicable. The weights w_i can be used to modify the lamp colours L_d and L_s. If we have more than one luminaire in the scene, we may have to adjust all lamps' colours (i.e. brightnesses) to match each other, so here the total lumen value given in the IESNA data file has to be taken into account. Also, w_i can be used to scale simulated photometric solids of the approximated lights in the interactive scene viewer application. In case a w_i is found which is very small in comparison with the others, so that its contribution in the scene is negligible, the light may be even omitted entirely to speed up the rendering.

4. OPTIMISATION

In order to optimise the overall error function we use a general function minimizer core algorithm applicable to different dimensionalities, in our cases 1–15, which correspond to the different light combinations. The situation is even more complicated because in some cases these parameters can be divided into independent subsets, e.g., when the two spotlights are oriented to directly opposite directions, having no relation between their weights on the overall errorfunction. Exploitation of this independence clearly decreases the computational cost in practice, and it is worth to exploit it in case of aiming for a fast optimisation for a case where the complexity of the problem is similar to ours.

Any commonsense error function must belong to the class C^2, meaning twice continuously derivable functions. More exactly, this property shall be fulfilled "almost everywhere" within the original domain. This means that discontinuities of the function or its derivatives can form only a very sparse subset in the entire domain, and also within commonsense less-dimensional subsets of it, like hyperplanes, lines or on its (hyper)spheres. We rely on these properties implicitly in the optimisation algorithm.

OverallOptimization(domain D**, errorfunction** f**, point** $firstGuess$**, clustersOfDirections** $(cl_i)_{i<I}$**):**
$optimalResultExternal := \infty$
point $p := firstGuess \in D$
for $attempts := 0$ to attemptsnum **do**
  $optimalResultInternal := \infty$
  repeat
    for all $i < I$ **do**
      $dir_i :=$ GoodCombinationOfDirections(p, f, cl_i)
    end for
    $goodDir :=$ GoodCombinationOfDirections(p, f, $(dir_i)_{i<I}$)
    ($result$, p) := DirectionalOptimization(p, f, $goodDir$)
  until $result \leq optimalResultInternal$
  if $optimalResultInternal < optimalResultExternal$ **then**
    $optimalResultExternal := optimalResultInternal$
    $optimalPoint := p$
  end if
  select new random multidimensional point $p \in D$
end for
return $optimalPoint$

GoodCombinationOfDirections(point p**, errorfunction** f**, direction set** $(dir_j)_{j\in J}$**):**
complete $(dir_j)_{j\in J}$ by the generalized gradient, i.e. the linear combination of $(dir_j)_{j\in J}$ with their corresponding directional derivatives $(d_j)_{j\in J}$: $\sum_{j\in J} d_j * dir_j$
$optimalResult := \infty$
for $attempts := 0$ to attemptsnum **do**
  $dir := \sum_{j\in J} c_j * dir_j$ {some linear combination of the (completed) set $(dir_j)\, j \in J$ by some random coefficients within the unit sphere $(c_j)_j \in J$}
  $result :=$ directional derivative at p toward dir
  if $result < optimalResult$ **then**
    $optimalResult := result$
    $optimalDir := dir$
  end if
end for
return $optimalDir$

DirectionalOptimization(point, errorfunction, direction):
reduce problem to a one variable problem, and use a combination of Newton and bisection methods.
Directional first and second derivatives are obtained by using finite difference forms.
return ($point$, $value$)

Figure 1: Pseudocode of the multidimensional solver

The optimisation algorithm consists of a structure of 3 nested loops, they are realized in the functions of Figure 1. The function **OverallOptimization** contains the outermost loop of independent attempts, each of them provides a quasi optimal result following from a random starting point. The core of its loop first defines a quasi optimal direction by calling the routine **GoodCombinationOfDirections** in a special way, then **DirectionalOptimization** is triggered by taking this direction.

Definition of the aforementioned quasi optimal direction uses information about classification of dimensions, in such a way that first defines a quasi optimal direction in each subspace belonging to the clustered dimensions' axis in question, and finally an overall quasi optimal direction is defined in the subspace defined by the obtained individual directions. This two pass definition with the appropriate **GoodCombina-**

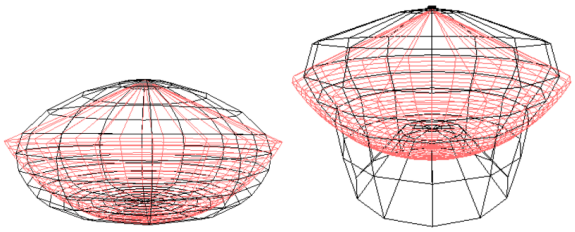


Figure 2: Photometric solids (black) and their OpenGL approximations for typical street lamps. The light bulbs sits on top of the lamps. *Left:* This light is modelled as a flattened spot with cutoff. Only some light emitted almost sideways is omitted by the approximated light. *Right:* The central obstruction of this luminaire causes the maximum brightness to fall along a ring. Such a distribution cannot be modelled properly with the simple OpenGL lights.

tionOfDirections function exploits the separation of dimensions properly.

The innermost **DirectionalOptimization** moves the point to an optimal position. This optimisation works fast on the different levels, exploiting the advantages of the regularities and conforms to the singularities as well. This flexibility characterises the innermost and also the middle loop. Consequently the algorithm provides a fast and reliable solution.

5. RESULTS AND CONCLUSIONS

We have implemented an IESNAlight class for use in a scene modeler and previewer application based on the OpenSG scene graph system [R*] to prepare scenes for a high quality rendering system. By showing wireframe models of the real and approximated photometric solids in addition to the approximative illumination given by the light, the scene modeler can easily see the quality of approximation for different kinds of luminaires (Fig. 2). The approximation computation takes a few seconds on a 2GHz class Athlon PC, so it is fast enough to include into an interactive application with IESNA file loader (Fig. 3).

Our method has shortcomings with lights that do not show spotlight characteristics (e.g. lights with central obstruction). An entirely different approach using the programmable shaders now available should be able to deal with the luminaire data directly and eventually bring even more realism into interactive applications.

Acknowledgements

This work was supported by the European Union within the scope of project IST-2001-34744, "Real-time Visualization of Complex Reflectance Behaviour in Virtual Prototyping" (RealReflect). Car model courtesy DaimlerChrysler.

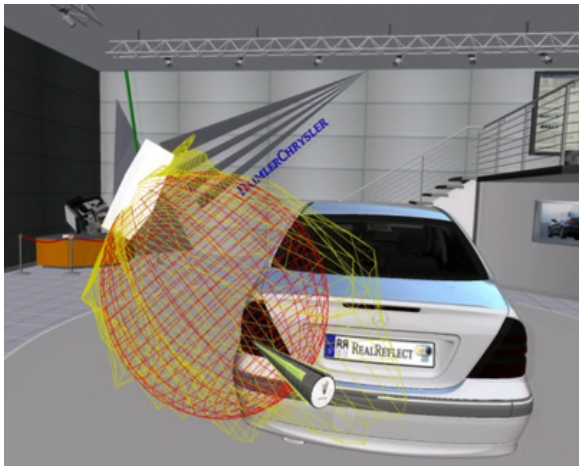


Figure 3: Usage of the visualization of a luminaire with asymmetric distribution in interactive illumination planning for an exhibition stage.

REFERENCES

[AP03] ALBIN S., PÉROCHE B.: Directionally dependent light sources. In *WSCG SHORT PAPERS proceedings* (Plzen, Czech Republic, 2003), UNION Agency – Science Press.

[Ash99] ASHDOWN I.: *Comparing Photometric Distributions*. Tech. rep., Department od Computer Science, University of British Columbia, 1999.

[Ash01] ASHDOWN I.: Thinking Photometrically Part II. LIGHTFAIR 2001 Pre-Conference Workshop, March 2001.

[GGHS03] GOESELE M., GRANIER X., HEIDRICH W., SEIDEL H.-P.: Accurate Light Source Acquisition and Rendering. In *Proceedings of ACM SIGGRAPH 2003* (2003), pp. 621–630.

[IES95] IESNA COMPUTER COMMITTEE: *IESNA Standard File format for Electronic Transfer of Photometric Data*. Illum. Eng. Soc. of N. America, New York, 1995. IESNA LM-63-95.

[LS98] LARSON G. W., SHAKESPEARE R.: *Rendering with Radiance*. Morgan Kaufmann Publishers, San Francisco, California, 1998.

[R*] REINERS D., ET AL.: OpenSG. Website. `http://www.opensg.org`.

[SWND04] SHREINER D., WOO M., NEIDER J., DAVIS T.: *OpenGL Programming Guide*, fourth ed. Addison-Wesley, 2004.

[WAL*97] WALTER B., ALPPAY G., LAFORTUNE E., FERNANDEZ S., GREENBERG D. P.: Fitting Virtual Lights for Non-Diffuse Walkthroughs. In *Computer Graphics Proceedings (SIGGRAPH)* (1997), pp. 45–48.

Shading by Quaternion Interpolation

Anders Hast
Creative Media Lab
University of Gävle, Sweden
aht@hig.se

ABSTRACT

The purpose of this paper is to show that linear interpolation of quaternions can be used for true Phong shading and also for related techniques that use frames, like bump mapping and anisotropic shading. Quaternion interpolation for shading has not been proposed in literature and the reason might be that it turns out to be mostly of academic interest, and it will here be explained why. Furthermore some pros and cons of interpolation using quaternions will be discussed. The effect of using this approach is that the square root in the normalization process disappears. The square root is now implemented in modern graphics hardware in such way that it is very fast. However for other types of platforms, especially hand held devices, the square root is computationally expensive and any software algorithm that could produce true Phong shading without the square root might turn out to be useful. It will be shown that linear interpolation of quaternion could be useful for bump mapping as well. However, quaternion arithmetic operations are not implemented in modern graphics hardware, and are therefore not useful until this is done.

Keywords

Phong Shading, Quaternion Interpolation

1. INTRODUCTION

Shading makes faceted objects appear smooth. Two widely used techniques are known as Gouraud [Gou71] and Phong Shading [Pho75]. Gouraud shading suffers from the Mach band effect and handles specular reflections poorly. This problem is diminished when Phong shading is used since the normals are interpolated instead of the intensities. However, the drawback with this approach is that all interpolated normals must be normalized. Otherwise, Phong shading will not be much different from Gouraud shading [Duf79].

This paper will show that it actually is possible to avoid the square root in the normalization process and still obtain normalized normals for Phong shading. This can be done when quaternions are used for the interpolation of the normals. If modern graphics hardware is used, then this is not really a problem anymore since these operations are as fast as multiplications and additions nowadays. However, quaternion arithemtics is not implemented in modern graphics hardware. If such arithmetics were implemented then it also could be used for bump mapping, as explained in this paper.

WSCG 2005 SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. PREVIOUS WORK

Shoemake [Sho85] introduced spherical linear interpolation (slerp) to the computer grapics society. Kuijk and Blake [Kuij89] showed how such angular interpolation could be used for faster Phong shading. They use spherical trigonometry to derive an equation for how both the normal and the vector in the direction to the light source varies over the polygon. A cosine has to be evaluated for each pixel. However, they propose a quadratic approximation that will make the evaluation faster. Abbas et al. [Abb00] elaborates this idea further for a suitable hardware implementation. Barrera et al. [Bar04] showed that equal angle interpolation of normals could be done in a very efficient way, removing the need for the division and square root, since this approach yields normalized normals in the interpolation process. However, the setup for each scanline involves the computation of several trigonometric functions. Hence, the gain in speed will be diminished by the extra setup, unless tables are used.

Bump mapping was introduced by Blinn [Bli78] as a method for making surfaces appear rough or wrinkled without increasing the number of polygons. Instead, the normals used in the lighting computations are perturbed to achieve this effect. Peercy et al. [Pee97] use an orthonormal frame on the surface to rotate the vector in the direction to the light source into that local frame. An overview of other bump map approaches is given by Ernst et al. [Ern98] and Kilgard [Kil00].

3. QUATERNIONS

Unit quaternions are related to orthonormal rotation matrices. The quaternion consists of four elements

$$q = (x, y, z, w) \tag{1}$$

A more compact form is to describe the three first elements as a vector since they constitute the imaginary part of the quaternion. The fourth part is the real part

$$q = (\mathbf{v}, s) \tag{2}$$

A quaternion may be used to rotate θ degrees around a vector **n** and it can be shown that the quaternion can be written as

$$q = (\mathbf{n}\sin(\theta/2), \cos(\theta/2)) \tag{3}$$

The quaternion rotation [Sva00] of a point in 3D space is defined as

$$p' = qpq^{-1} \tag{4}$$

where p is a vector in quaternion form

$$p = (\mathbf{r}, 0) \tag{5}$$

such as $\mathbf{r}=(x_p, y_p, y_p)$ and q^{-1} is the same as the conjugate of a quaternion for unit quaternions

$$q^{-1} = (-\mathbf{v}, s) \tag{6}$$

A quaternion can be transformed into a rotation matrix and vice versa [Wat92]. The rotation matrix corresponding to q in equation (1) is

$$\mathbf{M} = \begin{bmatrix} 1-2(y^2+z^2) & 2(xy+zw) & 2(xz+yw) \\ 2(xy+zw) & 1-2(x^2+z^2) & 2(yz+xw) \\ 2(xz+yw) & 2(yz+xw) & 1-2(x^2+y^2) \end{bmatrix} \tag{7}$$

The conversion from a matrix to a quaternion is a bit messier.

Frames

Even though it is not mentioned in literature, quaternions could be used for bump mapping and anisotropic shading [Pou90, Ban94], since they use frames on the surface, and quaternions are just another representation of frames. And orthogonal frames are rotation matrices. This fact is used in moving frame bump mapping, where the bump normal is rotated by the frame. However, in order to use quaternions as frames, there are several obstacles that must be handled. A quaternion can not be rotated by a frame. Hence frames, or at least the normal and the tangent, must be stored per vertex, rotated and then converted into quaternions. When bump mapping is used, the interpolated quaternion must either be converted into a rotation matrix for the rotation of the bump map normal, or the bump map normal can be rotated directly by the quaternion. However, this operation is not directly supported by modern graphics hardware, as matrix multiplication is. The advantage by interpolating quaternions, over interpolating the frame itself is that the qutaernion will still always define an orthonormal frame as long as the quaternion is normalized. When a frame is interpolated, the normal and tangent are interpolated. Both vectors must be normalized for each pixel. The binormal can be computed as the cross product of these two vectors. The resulting frame will usually not be orthogonal, but close enough.

Elimination of the Square Root

Another advantage by using quaternions is that the square root disappears. This will not have much effect for modern graphics hardware since the square root nowadays is very fast. However, for software shading it will have a larger impact. Quaternions are normalized in the same way as vectors are

$$q' = \frac{q}{\sqrt{q \cdot q}} \tag{8}$$

Nonetheless, equation (7) shows that each element of the normalized quaternion is multiplied with another element of the same quaternion, when the elements of the matrix is computed. For an example we have

$$q'_x q'_y = \frac{q_x}{\sqrt{q \cdot q}} \frac{q_y}{\sqrt{q \cdot q}} = \frac{q_x q_y}{q \cdot q} \tag{9}$$

Hence, the square root disappears, but the division is still necessary. Moreover, it can be shown that the square root disappears for quaternion rotation as it is defined in equation (4). The rotation can be simplified as [Wat92]

$$qpq^{-1} = \mathbf{r}(ss - \mathbf{v} \cdot \mathbf{v}) + 2\mathbf{v}(\mathbf{v} \cdot \mathbf{r}) + 2s(\mathbf{v} \times \mathbf{r}) \tag{10}$$

Clearly, any element of the quaternion q, will be multiplied with some other element of q and the square root can be removed. This fact is nothing new,

but it is possible to utilize it for shading as is shown in the next section.

4. SHADING

Shading can utilize quaternion interpolation, since the normal can be obtained from the resulting matrix. Thus, it is not necessary to compute the tangent and binormal, in order to obtain the normal itself. The normal is found in the third column of the matrix in equation (7). Hence, the normal could be computed as

$$\mathbf{n}=\begin{bmatrix}2(xz+yw)\\2(yz+xw)\\1-2(x^2+y^2)\end{bmatrix} \tag{11}$$

Equation (11) can be compared to ordinary vector interpolation where each element is linearly interpolated and the dot product can be quadratically interpolated [Duf79], followed by a division and a square root. When quaternions are interpolated, there are four linear interpolations followed by equation (8) and (11). Even though multiplications and additions are several times faster than the square root on ordinary CPU's, there is not much time saved by this method. However, we can still do better.

Fast Normal interpolation

Note, that equation (11) would be much simpler if it would be possible to arrange so that the fourth element *w* is always set to zero. The normal becomes

$$\mathbf{n}=\begin{bmatrix}2xz\\2yz\\1-2(x^2+y^2)\end{bmatrix} \tag{12}$$

This would correspond to rotating the frame at the vertices in such way that the normal is still the same, but the tangent and binormal is rotated around the normal. This can be done. Since the quaternion should be normalized, but *w* is zero, then

$$z^2+x^2+y^2=1 \tag{13}$$

Thus

$$x^2+y^2=1-z^2 \tag{14}$$

Substitute equation (14) into equation (11) for computing $\mathbf{n}_z$

$$\mathbf{n}_z=2z^2-1 \tag{15}$$

Solving for z gives

$$z=\sqrt{\frac{\mathbf{n}_z+1}{2}} \tag{16}$$

Now, equation (16) could be used to compute *x* and *y* by substituting the expression for *z* into equation (12). Hence

$$\begin{aligned}x&=\frac{\mathbf{n}_x}{2z}\\ y&=\frac{\mathbf{n}_y}{2z}\end{aligned} \tag{17}$$

Equations (16) and (17) need to be computed per vertex. Hence, the square root is necessary three times per polygon instead of one square root per pixel. The following vertex shader code in the OpenGL shading language exemplifies the idea

```
varying vec3 q;
...
q.z=sqrt((normal.z+1.0)*0.5);
float t=1.0/(2.0*q.z);
q.xy=normal.xy*t;
```

Here, *q* is the quaternion that is interpolated over the polygon, *normal* is the normal at the vertices and *t* is temporary variable used to optimize the computation.

The normal at each pixel is computed by equation (12), where the interpolated quaternion first is normalized by equation (9). Using equation (15) instead of computing $\mathbf{n}_z$ with equation (12) gives the following fragment shader code in the OpenGL shading language

```
float t= 2.0*q.z/dot(q,q);
vec3 N=t*q.xyz;
N.z-=1.0;
```

Here, *q* is the quaternion and *t* is temporary variable used to optimize the computation.

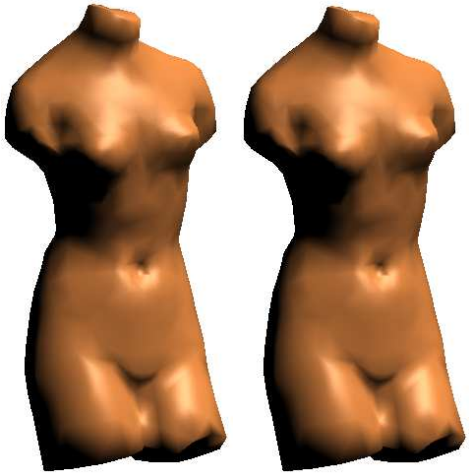


Figure 1. Left: Phong shading. Right: Fast linear quaternion interpolation

Figure 1 shows the famous Venus De Milo statue Phong shaded to the left. It can be compared to the

image to the right, which is shaded using linear interpolation of quaternions. The figures are indistinguishable form each other. It can be shown that the normals produced by this type of interpolation, does not necessarily lie in the same plane, as normals obtained by linear interpolation does. However, it is not a requirement that the normals along a scanline should lie in the same plane. If the object is slightly rotated, then the scanline will be a new cross section of the surface that the polygon covers, which did not have their normals in the same plane in the previous frame.

5. DISCUSSION

Even though the square root is replaced by just a few simple arithmetic operations, the gain in speed when using this type of interpolation is small. It is not in the scope of this paper to implement quaternion interpolation on different platforms, since the calculation speed of different operations may vary quite a lot among platforms.

It should be noted that is possible to compute the inverse square root [Tur95] instead of normalizing a vector using division and the ordinary square root. This algorithm is based on the Newton-Raphson method and involves no divisions, but the proposed algorithm has one division. On the other hand it is possible to compute the division using the Newton-Raphson method more effectively than the inverse square root is computed. Hence, the proposed method may turn out to be faster on some platforms.

Quaternion arithmetic operations are not implemented in modern programmable graphics hardware. Therefore, bump mapping will not be faster using quaternion interpolation. However, if these operations were implemented, the conversion between quaternions and rotation matrices would be fast. Moreover, quaternion rotation could be very fast if it was implemented in hardware. If this is done, then quaternion interpolation could be standard procedure in the future, for bump mapping and anisotropic shading.

6. CONCLUSIONS

It is possible to perform true Phong shading by linear interpolation of quaternions representing the frame of the surface. The square root disappears in this process. A faster version can be obtained of this scheme, by rotating the frame in such way that the fourth element is always zero. This scheme may be useful for some software implementations.

It was also discussed that linear interpolation of quaternions could be useful for bump mapping and anisotropic shading if quaternion arithmetics were implemented in modern graphics hardware in the future.

7. REFERENCES

[Abb00]A. M. Abbas, L. Szirmay-Kalos, T. Horvath, Hardware Implementation of Phong Shading using Spherical Interpolation, Periodica Polytechnica, Vol. 44, Nos 3-4, 2000.

[Ban94] D. Banks. Illumination in Diverse Codimensions. In Proceedings SIGGRAPH (July 1994), pp. 327–334.

[Bar04] T. Barrera, A. Hast, E. Bengtsson, Faster shading by equal angle interpolation of vectors, IEEE Transactions on Visualization and Computer Graphics, pp. 217-223, 2004.

[Bli78] J. F. Blinn, Simulation of Wrinkled Surfaces, Proceedings SIGGRAPH 1978: pp. 286-292.

[Duf79] T. Duff, Smoothly Shaded Renderings of Polyhedral Objects on Raster Displays, ACM, Computer Graphics, Vol. 13, pp. 270-275, 1979.

[Ern98] I. Enrst, H. Rüssler, H. Schultz, O. Wittig. Gouraud Bump mapping. Workshop on Graphics Hardware, pp. 47-53. 1998.

[Gou71] H. Gouraud, Continuous Shading of Curved Surfaces, IEEE transactions on computers vol. c-20, No 6, June 1971.

[Kil00] M. J. Kilgard A Practical and Robust Bump-mapping Technique for Today s GPUs Game Developers Conference, Advanced OpenGL Game Development. 2000.

[Kui89] A. A. M. Kuijk, E. H. Blake, Faster Phong Shading via Angular Interpolation, Computer Graphics Forum, vol. 8, No 4, pp. 315-324 1989.

[Pee97] Peercy, A. Airey, B. Cabral, Efficient Bump Mapping. Hardware In proceedings of SIGGRAPH, pp. 303-306. 1997.

[Pho75] B. T. Phong, Illumination for Computer Generated Pictures, Communications of the ACM, Vol. 18, No 6, June 1975.

[Pou90] P. Poulin, A. Fournier, A model for anisotropic reflection, Proceedings ACM SIGGRAPH, Vol. 24 No 4, pp 273 - 282 , 1990.

[Sho85] K. Shoemake, Animating rotation with quaternion curves, Proceedings ACM SIGGRAPH, Vol. 19 No 3, July 1985.

[Sva00] J. Svarovsky. Quaternions for Game Programming, Game Programming Gems. Charles River Media, pp. 195-299. 2000.

[Tur95] K. Turkowski, Computing the Inverse Square Root. Graphics Gems V, Academic Press, pp. 16-21. 1995.

[Wat92] A. Watt, M. Watt. Advanced Animation and Rendering Techniques - Theory and Practice. Addison Wesley, pp. 363-364.

Memory Efficient Adjacent Triangle Connectivity of a Face Using Triangle Strips

Hidekuni Annaka
annaka@src.ricoh.co.jp

Tsukasa Matsuoka
matsuoka@src.ricoh.co.jp

Akihiro Miyazawa
miyazawa@src.ricoh.co.jp

Multimedia Lab, Software R&D Group
Ricoh Company, Ltd.

ABSTRACT

We often need to refer to adjacent elements (e.g., vertices, edges and faces) in triangle meshes for rendering, mesh simplification and other processes. It is, however, sometimes impossible to prepare the enormous memory needed to represent element connectivity in gigantic triangle meshes. We proposed a memory efficient scheme for referring to adjacent faces around a vertex in non-manifold triangle meshes [AM04]. But the scheme has a redundancy in case of two-manifold triangle meshes. This paper proposes new schemes for referring to adjacent faces around a face in two-manifold triangle meshes. First, as our previous scheme, we introduce the constraints to allow random access to a triangle in a sequence of triangle strips. Then, for each face, we construct a list of references to adjacent strips as a representation of triangle connectivity. Experimental results show that, compared with conventional indexed triangle set based methods, our schemes reduce total strage for a triangle mesh and adjacent triangle connectivity by less than 50%.

Keywords: strip, mesh, memory efficient,adjacent connectivity

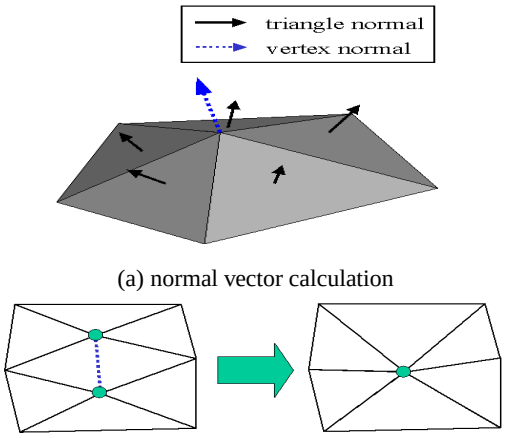
1 INTRODUCTION

One of the most popular representations for 3D models is a triangle mesh. It has often been used not only for rendering but also for various processes in many scenes. In such scenes, adjacent triangle connectivity is frequently used: to calculate vertex normals, to simplify meshes to reduce storage, and to detect collisions for virtual reality systems.

The rendering of triangle meshes requires not only vertex coordinates but also vertex normals. To reduce memory, without storing vertex normals, adjacent triangles around a vertex must be acquired to successively calculate its normal from three triangle vertices (figure 1(a)). Adjacent connectivity of vertices and triangle faces is also required to simplify triangle meshes in the popularly used *vertex contract* operation [Gar99], which unifies two vertices of triangle meshes. When faced with gigantic triangle meshes, however, it is impossible to prepare the enormous memory needed to represent adjacent connectivity. Those gigantic meshes are generated from the latest 3D shape input devices, for example laser range scanners, and we must process triangles of those models numbering in the hundreds of millions.

Therefore, we devised a method based on a triangle strip reference to represent adjacent faces of a vertex [AM04]. Required total storage for a triangle mesh and adjacent triangle connectivity is half that of conventional representation based on an **ITS** (indexed triangle set mesh).

WSCG 2005 SHORT papers proceedings
ISBN 80-903100-9-5, January 31-February 4, 2005
Plzen, Czech Republic.



(a) normal vector calculation

(b) mesh simplification

Figure 1: Usage of adjacent relations

But most of triangle meshes are 2-d manifolds where adjacent connectivity can be represented using adjacent faces of a face. This is why, in this paper we propose a more memory efficient scheme for referring to adjacent faces of a face. In this new scheme, required total storage is about 75% of our previous scheme. Our representation is characterized by the following:

- **CTS (Constrained Triangle Strips)**
  In accessing arbitrary triangles of multiple strips in a strip buffer, we use constrained index triangle strips. We insert one or two delimiters between strips so that the strip size is always an even number. Three vertices orders of triangles at even indices in the strip are counterclockwise and three vertices orders of triangles at odd indices are clockwise in the strip.

- **reference to adjacent faces of a face using CTS**
  We use strip indices to suppress storage size for adjacent face connectivity of a face. A triangle has three adjacent triangles, and three triangle indices are required to store adjacent face references. In triangle strip representation, two adjacent triangles are typically in the same strip and one adjacent face is in a different strip (figure 4). Therefore, one strip index is usually required to store adjacent face references. This configuration is three times more efficient than ITS based representation [Gar99].

Here are some details on how we suppress storage size: We use 63.6%–63.7% the storage of ITS based representation by not using an indexed triangle set. Instead, we use a constrained index triangle strip for triangle connectivity. Further, we use 46.6%–46.7% the storage of ITS based representation by using strip indices for adjacent faces connectivity of a face rather than triangle indices.

The triangle strip is supported by OpenGL APIs and provides fast rendering on popular hardware accelerators. Our method is especially efficient both for adjacent triangle connectivity of vertices and for rendering meshes.

The remainder of this paper is organized as follows: Section 1.1 summarizes related work. Section 2 describes indexed triangle sets and indexed triangle strips for triangle mesh representations. Section 3 introduces our constraints against indexed triangle strips to suppress adjacent triangle connectivity of a vertex. Section 4 describe adjacent faces connectivity of a face by using constrained index triangle strips. Section 5 presents the results of experiments using the proposed method and Section 6 discusses conclusions.

1.1 Related work

Some proposed data structures for adjacent connectivity between elements of 3D models can represent various genus models. The winged-edge [Bau72] is a representative structure that can represent two-manifold models. 3D models often consist of non-manifold surfaces in actual scenes. The radial-edge [Wei88] is a representative structure that can represent non-manifold models.

Those structures are popular representations. The following triangle mesh representation [NDW93] can represent non-manifold models with no explicit adjacent connectivity between elements but vertex connectivity of each triangle.

- triangle set
- triangle strip
- triangle fan

The above representations are suitable for sequential accessing and the rendering of triangles because vertex coordinates are stored directly in the coordinate table. This being the case, common vertex coordinates are stored separately in the coordinate table, so the table cannot be used for referencing neighborhoods.

The following representations are derived from the above representations where each triangle consists of three indices to a vertex coordinate table to share coordinates with other triangles.

- indexed triangle set
- indexed triangle strip
- indexed triangle fan

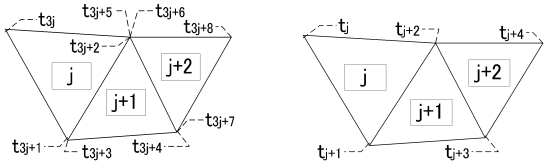
Because of the absence of explicit adjacent connectivity, it takes a long time to refer to adjacent elements of these triangle meshes, for example, to adjacent vertices of a vertex and adjacent triangles of a vertex. It is faster to refer to adjacent elements with explicit adjacent connectivity than without. Hoppe uses adjacent triangles of a triangle in the Progressive Mesh [Hop97] techniques for two-manifold meshes. Garland et al. use adjacent triangles of a vertex in the triangle mesh simplification [Gar99] techniques for non-manifold meshes. In the Hoppe techniques, adjacent connectivity consists of adjacent triangles around a triangle and is equal to the size of triangle connectivity. With Garland techniques, adjacent connectivity is equal to the size of the triangle connectivity. We proposed a scheme for referring to adjacent triangles around a vertex in non-manifold triangle meshes [AM04]. But the scheme may have a storage redundancy in case of two-manifold meshes. Because we can refer to all faces referring adjacent faces around a face in case of two-manifold mesh. This paper proposes a memory efficient method to acquire adjacent triangles around a face for two-manifold meshes.

2 TRIANGLE MESH REPRESENTATIONS

A triangle mesh M is represented widely as a tuple consisting of vertex coordinates table V and vertex indices table T. Coordinates table $V = (v_0, v_1, v_2, ...v_m)$ lists vertex coordinates, each element of which corresponds to a vertex coordinate $v_i = [x_i, y_i, z_i]$. Indices table $T = (t_0, t_1, t_2, ...t_n)$ lists vertex indices that compose triangles. Each element of T corresponds to an index t_j to V. We classify meshes into two representations according to interpretation of T: an indexed triangle set representation and an indexed triangle strip representation.

2.1 Indexed triangle sets

Figure 2(a) shows an indexed triangle set representation where a triplet of consecutive indices, such as $[t_{3j}, t_{3j+1}, t_{3j+2}]$, $[t_{3j+3}, t_{3j+4}, t_{3j+5}]$, and $[t_{3j+6}, t_{3j+7}, t_{3j+8}]$, denotes three vertex indices of a triangle. For each triplet $[t_{3j}, t_{3j+1}, t_{3j+2}]$, we refer to j as a *triangle index*, which is one third of the subscript of the first vertex index. The orientations of a triangle are decided by the order of the three indices that compose the triangle. Generally speaking, the side on which the order is counterclockwise is the front side. It is easy to refer to an arbitrary triangle by triangle index j in the case of an indexed triangle set, because three vertex indices $[t_{3j}, t_{3j+1}, t_{3j+2}]$ are uniquely derived from the triangle index j.



(a)triangle set (b)triangle strip

Figure 2: Triangle mesh representations

2.2 Indexed triangle strips

Figure 2(b) shows an indexed triangle strip representation where a triplet of consecutive indices, such as $[t_j, t_{j+1}, t_{j+2}]$, $[t_{j+2}, t_{j+1}, t_{j+3}]$, and $[t_{j+2}, t_{j+3}, t_{j+4}]$, denotes three vertex indices of a triangle. Note two indices in each triplet are also used in the previous and next triplets. For each triplet $[t_j, t_{j+1}, t_{j+2}]$, we refer to j as a *triangle index*, which is the subscript of the first vertex index and starts from 0. The orientations of a triangle are decided by the order of three indices [NDW93] in this representation also. Unlike the indexed triangle set, however, the order flips alternately depending on whether the triangle index is even or odd.

The indices table of an indexed triangle strip mesh is about half to one third the size that of an indexed triangle set mesh. This is because the indices are shared in an indexed triangle strip representation. In addition, we can render indexed triangle strip meshes faster than indexed triangle set meshes because the number of vertices processed by the rendering libraries (e.g., OpenGL) is smaller.

Indices table T usually represents a sequence of indexed triangle strips. It is difficult, however, to refer to an arbitrary triangle by triangle index j in the case of a sequence of indexed triangle strips. Specifically, if the indices table stores multiple strips to represent a model, we can't determine the orientation of a triangle by the triangle index. This is because the parity (being odd or even) of triangle index j, which holds true in one strip by definition, doesn't match the parity defined by the number counted from the beginning of T, which consists of a sequence of multiple strips of different lengths, where each strip ends with a delimiter -1.

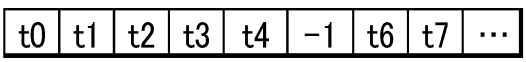
3 REVIEW OF CONSTRAINED TRIANGLE STRIPS

This section reviews a method that enables random access to a triangle in a sequence of triangle strips [AM04]. Access is accomplished by imposing a constraint on the triangle strips described in the previous section.

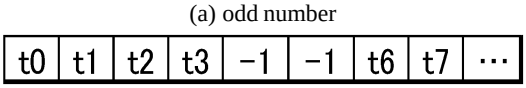
In this method, we insert delimiters depending on the size of strip indices as follows:

- If the size of the strip is an odd number, we insert one delimiter (Figure 3(a)).
- If the size of the strip is an even number, we insert two successive delimiters (Figure 3(b)).

As a result, the three vertices compose a triangle in counterclockwise order if the triangle index is even, or in clockwise order if the index is odd. This enables us to find the orientation of any triangle in a sequence of strips.



(a) odd number



(b) even number

Figure 3: Constrained triangle strips

4 ADJACENT STRIPS OF A FACE

This section describes a new scheme for referring to adjacent triangles around a face in two-manifold triangle meshes, by using constrained index triangle strips.

In this section, let T be an index table that represents the constrained triangle strips. We define a data structure that represents *adjacent strips of faces* $A = (a_0, a_1, \ldots, a_n)$ whose size is the same as that of T. Each element of A is an index to T. As shown in Figure 4, for a given triangle j, an element a_{j+1} of A generally represents a triangle that is adjacent to j and belongs to a strip different from j's. If j is a starting or ending triangle of a strip, a_j or a_{j+2} represents another adjacent triangle of j.

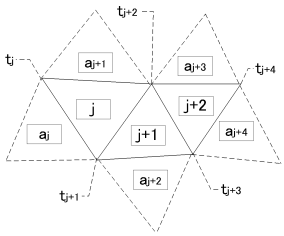


Figure 4: Adjacent strips of faces

For a given triangle j, which is composed of three vertices $[t_j, t_{j+1}, t_{j+2}]$, we can refer to its three adjacent triangles as follows:

1. If $t_{j-1} < 0$, the first adjacent triangle is a_j, which is composed of $[t_{a_j}, t_{a_j+1}, t_{a_j+2}]$. Otherwise, the first one is $j-1$, i.e., $[t_{j-1}, t_j, t_{j+1}]$.
2. The second adjacent triangle is always a_{i+1}, which is composed of $[t_{a_{j+1}}, t_{a_{j+1}+1}, t_{a_{j+1}+2}]$.
3. If $t_{j+3} < 0$, the third adjacent triangle is a_{j+2}, which is composed of $[t_{a_{j+2}}, t_{a_{j+2}+1}, t_{a_{j+2}+2}]$. Otherwise, the third one is $j+1$, i.e., $[t_{j+1}, t_{j+2}, t_{j+3}]$.

Figure 5 shows an example of referring adjacent strips of faces in Figure 4.

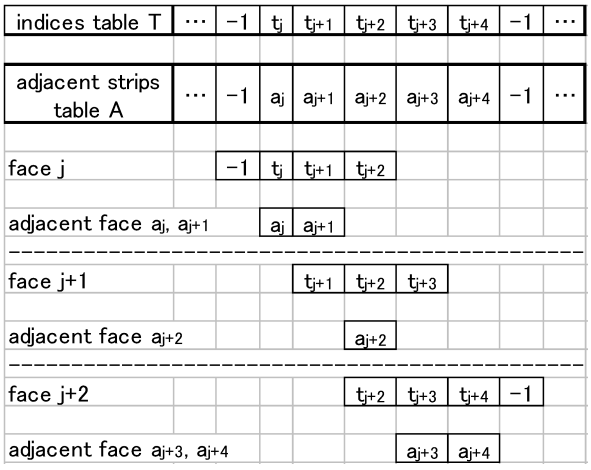


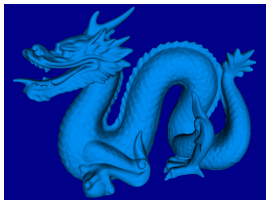
Figure 5: Example of referring adjacent strips of faces

5 RESULTS

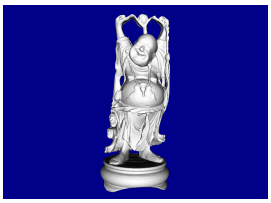
We implemented the schemes described in the previous sections and compared performance results with those of ITS schemes on a personal computer with a Pentium 4, 1.8 GHz CPU, 512MB of main memory and an NVIDIA GeForce2 MX 400 32MB VGA card. Performance was measured for the following items:

- total storage size ($|T+J|$ or $|T+A|$) : the total storage size of the vertex indices table ($|T|$) and the adjacent triangle connectivity of vertices ($|J|$) or triangles ($|A|$)[MB]
- access time (t_a) : the time to refer to adjacent triangles around all i) vertices or ii) triangles[ms]. The number of references is the same in both cases i) and ii). This is because the number in the case i) is eventually equal to the sum of the number of vertices (three) of each triangle and the number in the case ii) is equal to the sum of the number of edges (three) of each triangle.
- rendering time (t_r) : the time to render triangle meshes [ms]

We used the following triangle meshes, (a) Dragon and (b) Happy Buddha [1], in experiments and used the stripification algorithm [ESV96] for the meshes.



(a) Dragon
vertices : 437645
triangles : 871414



(b) Happy Buddha
vertices : 543652
triangles : 1087717

Figure 6: Experimental models

Table 1 lists experimental results of the following four methods.

- M_{tv} : ITS based method for referring to adjacent triangle around a vertex [Gar99].
- M_{sv} : Our previous method [AM04].
- M_{tt} : ITS based method for referring to adjacent triangle around a face [Hop97].
- M_{st} : Our proposed method.

The results show that M_{st} decreases total storage size by 46.6%–46.7% compared to M_{tt}, and by 75.0% – 75.1% compared to M_{sv}.

Regarding the access speed, the results show that M_{st} increases access time (t_a) by 113%–116%, compared to M_{tt}, and decreases by 82.7% – 82.9% compared to M_{sv}.

Considering the rendering time as well, the results show that M_{st} decreases total time ($t_a + t_r$) by 77.7%–78.0%, compared to M_{tt}, and by 87.6% – 87.7% compared to M_{sv}.

mesh	method	$\|T\|$	$\|A\|$ or $\|J\|$	$\|T+A\|$ or $\|T+J\|$	t_a	t_r	t_a+t_r
(a)	M_{tv}	19.9	26.6	46.6	210	345	555
	M_{sv}	9.3	15.5	24.8	411	161	572
	M_{tt}	19.9	19.9	39.9	300	345	645
	M_{st}	9.3	9.3	18.6	340	161	501
(b)	M_{tv}	24.9	33.2	58.1	260	441	701
	M_{sv}	11.6	19.3	30.9	520	202	722
	M_{tt}	24.9	24.9	49.8	371	441	812
	M_{st}	11.6	11.6	23.2	431	202	633

Table 1: Experimental results of referring to adjacent triangles around faces or vertices

6 CONCLUSION

Our proposed method achieves positive results for storage size and total time compared to the ITS based methods and our previous method. Regarding the access time, our proposed method is better than our previous method, although our proposed method is at a disadvantage compared to the ITS based methods. Our proposed method, however, decreases the total time with respect to the ITS based methods, because it exploits OpenGL triangle strips

Future work will focus on reducing access time.

References

[AM04]Hidekuni Annaka and Tsukasa Matsuoka. Memory efficient adjacent triangle connectivity of a vertex using triangle strips. In *Computer Graphics International 2004 (CGI2004) Conference Proceedings*, June 2004.

[Bau72]Bruce Guenther Baumgart. Winged edge polyhedron representation. Technical report, Stanford Artificial Intelligence Laboratory, October 1972. CS-320.

[ESV96]Francine Evans, Steven Skiena, and Amitabh Varshney. Optimizing triangle strips for fast rendering. In Roni Yagel and Gregory M. Nielson, editors, *IEEE Visualization '96*, pages 319–326, 1996.

[Gar99]Michael Garland. *Quadric-Based Polygonal Surface Simplification*. PhD thesis, School of Computer Science Carnegie Mellon University, 1999. CMU-CS-99-105.

[Hop97]Hugues Hoppe. View-dependent refinement of progressive meshes. In *SIGGRAPH'97 Conference Proceedings*, pages 189–198, August 1997.

[NDW93]Jackie Neider, Tom Davis, and Mason Woo. *OpenGL Programming Guide*. Addison-Wesley Publishing Company, August 1993.

[Wei88]Kevin Weiler. The radial-edge structure: A topological representation for non-manifold geometric boundary representations. In *Geometric Modelling for CAD Applications*, pages 3–36, North Holland, 1988.

[1] Models are courtesy of Stanford 3D Scanning Repository.

Reconstruction and Representation of Tubular Structures using Simplex Meshes

Alexander Bornik Bernhard Reitinger Reinhard Beichel

Institute for Computer Graphics and Vision
Graz University of Technology
In eldgasse 16/II, A–8010 Graz, Austria
contact: bornik@icg.tu-graz.ac.at

ABSTRACT

Modelling and reconstruction of tubular objects is a known problem in computer graphics. For computer aided surgical planning the constructed geometrical models need to be consistent and compact at the same time, which known approaches cannot guarantee. In this paper we present a new method for generating compact, topologically consistent, 2-manifold surfaces of branching tubular objects using a two-stage approach. The proposed method is based on connection of polygonal cross-sections along the medial axis and subsequent re nement. Higher order furcations can be handled correctly.

Keywords

mesh generation, branched tubular objects, simplex meshes, medical context

1 INTRODUCTION

Modelling surface representations of tubular objects has been addressed by many researchers in the eld of computer graphics leading to numerous approaches. Since many anatomical structures like vessels are of tubular type, medical applications are one important application eld for such techniques. Arrangements of such tubular tubular structures like the portal vein tree inside the liver or airway trees are highly complex and heavily branched. New algorithms are able to detect even very thin tubular structures [Beich04] in e.g. CT images, leading to medial axis representations organized in graphs, which are not suitable for direct rendering. Therefore surface models, which can be e ciently rendered need to be set up rst for applications like computer aided surgical planning.

WSCG SHORT papers, ISBN 80-903100-9-5
WSCG'2005, January 31–February 4, 2005
Plzen, Czech Republic.

Medical application require consistent surface models without gaps or self-intersections. In case the model is a polygonal or triangular mesh, it should be a 2-manifold. The mesh must be adaptive to account for changing diameter. The overall primitive count should be small, while still accurately representing small geometrical details. The presented work was done in the context of developing a virtual liver surgery planning system [Borni03] supporting physicians with tools for planning and staging of liver tumor resections using a virtual reality (VR) environment. This application involves visualization of the portal vein inside the liver. Since VR techniques are used, the graphical representation of the portal vein needs to be compact in order to meet the real-time requirement, while still providing the physicians with the correct topological information needed for planning a surgical intervention.

2 RELATED WORK

The probably most popular surface reconstruction algorithm is named Marching Cubes (MC) [Loren87]. It can be used for direct reconstruction of surface models from 3D data organized in a grid and is often used in the medical context. However, one would have to convert a centerline representation to a voxel dataset rst, in order to employ MC. Besides, MC is not adap-

tive and produces complex geometry. For VR applications mesh compression has to be employed, introducing processing overhead. Staircase artifacts produced by MC must be removed using ltering techniques. Connectivity problems for 26-connected structures, which are not acceptable in surgical planning, require a modi ed algorithm [Lewin03]. Besides the MC there are generic algorithms involving deformable models directly [Delin99], which is time consuming for tubular objects. A second class of algorithms involves surface tiling from cross-sections and algorithms connecting tubular objects to a single model [Oliva96, Meyer92, Lluch04]. Most of these algorithms have problems representing furcations of higher order correctly, some do not support changing radii along branches. An algorithm related to ours can be found in [Felke04], however the technical solution is di erent. Since the approach is not based on a deformable model, mesh re nement is not possible. An approach mainly focussing on rendering performance, not mesh consistency, can be found in [Hahn01].

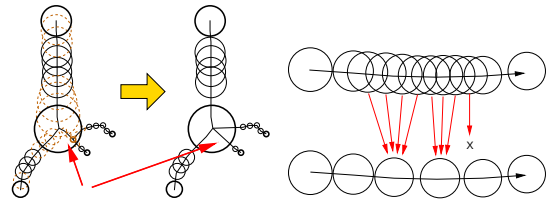
3 METHODS

The proposed algorithm generates compact surfaces representations of branched tubular objects. It can handle bifurcations, trifurcations and higher order furcations. The output is a topologically consistent 2-manifold mesh. Geometrically our approach is based on a simplex mesh structure [Delin99]. The mesh for the tubular object is constructed in two stages:

1. **An initial mesh is constructed** connecting cross-section polygons along the medial axis.
2. **Mesh Refinement** based on simplex meshes as deformable model.

3.1 Preprocessing

We use a directed acyclic graph data structure as input data for mesh initialization. The nodes of the graph are either branching points or endpoints. Each node stores its location, local radius, cross-section normal and connected links. Links (branches) store their endpoints, direction and a list of intermediate centerline points with cross-section radii and normals. This data is provided by a vessel-mining approach [Beich04]. Since a deformable model is employed in the second stage of our algorithm, limiting to circular cross-sections during the initialization is possible without loss



(a) (b)

Figure 1: (a) Thinning near branching points: Cross-sections intersecting the in uence area of a branching point (sphere) are removed. Sometimes this requires graph restructuring (see arrows). (b) Removal of centerline points within branches.

of generality. Spheres (with cross-section radius) spanned by neighboring centerline points and radii can intersect. Sometimes a whole child branch is located within the sphere at its parent's branching point, which can lead to artifacts. Preprocessing by thinning leads to initial meshes almost free of visual artifacts, since possibly overlapping cross-sections are removed. Thinning also makes initial meshes compact, because the number of remaining cross-sections directly in uences the number of mesh polygons. Figure 1 shows details of the thinning process.

3.2 Initial Mesh Setup

The initial mesh is set up by traversing the preprocessed input data in a breadth- rst-search manner starting with a simplex mesh cylinder at the root node.

Connection of cross-sections

Further cross-sections of the root branch are successively connected to the existing geometry at the top polygon. This is achieved by connecting each vertex of the cross-section polygon to the existing geometry as shown in Figure 2. According to this schema, each inserted cross-section of the tubular object has to have the same number of vertices as the polygon it connects to. This limitation can be overcome by either not splitting each edge to connect to or inserting intermediate polygons. For maximum visual quality, polygons are oriented so that the overall connection edge length is minimized. The visual quality can be further improved by placing additional vertices at the barycenter of the triangles formed by the base edge of the polygon to connect to and the connecting cross-section

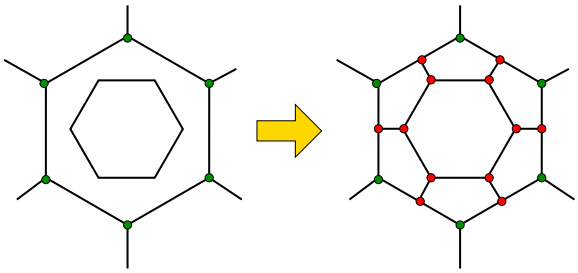


Figure 2: Polygon insertion schema.

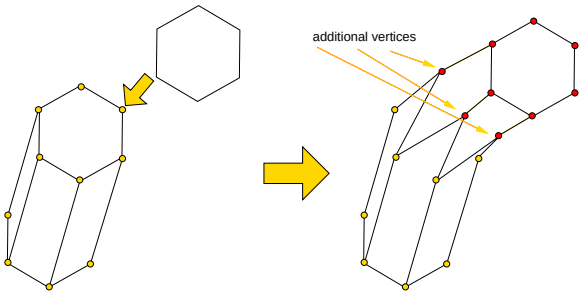


Figure 3: Polygon insertion operation in 3D.

vertex. Figure 3 shows a 3D example.

Connection of branches

New branches are added by connecting their rst medial axis cross-section to the existing geometry. The best-suited polygon is found in the vicinity of the branching point, the so-called *hot spot region*, which is maintained for all branching points. When it comes to connecting the rst cross-section i with center $\vec{c}_i$ and normal $\vec{n}_i$ of a new branch, the following criterion is calculated for each hot spot region polygon p with $\vec{c}_p$:

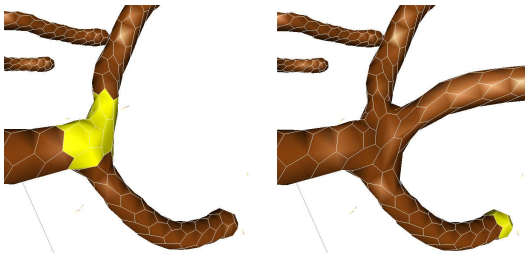
$$\zeta_i = \frac{1}{\|\vec{c}_p \quad \vec{c}_i\|}\vec{n}_p \cdot \vec{n}_i$$

The polygon that maximizes ζ_i is connected using the same technique as for intermediate cross-sections. Figure 4 shows the hot spot region of a tubular mesh before connecting a new branch as well as how the same region looks like in the nal mesh.

3.3 Mesh Refinement

The overall visual quality of the mesh and particularly the quality near high order furcations is improved using the simplex mesh as a deformable model based on a Newtonian law of motion, which is iteratively solved:

$$\vec{P}_i^{t+1} = \vec{P}_i^t + (1 \quad \gamma)(\vec{P}_i^t \quad P_i^{t-1}) + \alpha_i \vec{F}_{int} + \beta_i \vec{F}_{ext}$$



(a) (b)

Figure 4: (a) Simplex mesh of a tubular objec before connecting a new branch. The hot spot region is marked bright. (b) Mesh after the new branch has been added.

Each vertex $\vec{P}_i$ is displaced based on internal regularizing forces $\vec{F}_{int}$ and external forces F_{ext}. α_i and β_i are control parameters, while γ is a damping factor. If only a medial axis representation is available external $\vec{F}_{ext}$are calculated towards sampling points de ned by the cross-sections and radii. Performing several iterations having centerline/radius input mainly results in geometrical changes near branching points. In case of a present binary segmentation in form of a 3D volumetric dataset, external forces are calculated by searching the closest boundary voxel in the direction of each vertex' normal. In this case mesh iterations also a ect regions, with other than circular cross-section shape, leading to more accurate results.

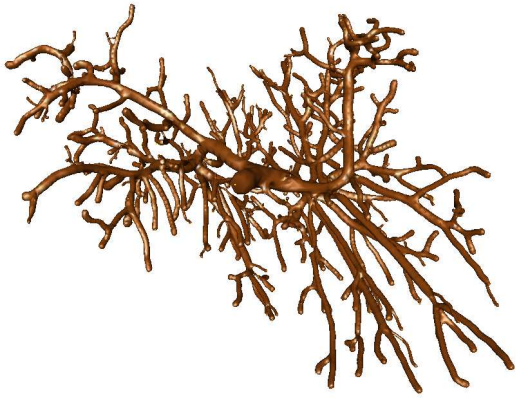
4 RESULTS

We have tested our algorithm with di erent types of medical datasets, including the vasculature of the liver, sheep and human airway trees and data from high-resolution scans of portal vein corrosion casts. Our approach produced consistent and visually appealing meshes in all cases. The number of polygons produced by our approach is the same order of magnitude as output produced by a simpli ed and ltered MC approach. Figure 5 shows results for the datasets described in Table 1. Table 2 gives an overview of the mesh complexity and timings of the proposed algorithm.

5 CONCLUSION

This paper presented an algorithm for generating compact consistent 2-manifold meshes of branched

(a)



(b)

Figure 5: Results: (a) DS 1: Small arti cial tubular dataset. (b) DS 2: Human portal vein tree.

DS	Description	#b	#cs
1	Arti cial	8	64
2	Human portal vein	693	2680

Table 1: Test dataset overview: *#b*: number of branches. *#cs*: number of input cross-sections.

DS	#vtx	#pol	t_i	t_r	n
1	950	477	28	25	40
2	24199	48394	527	1738	10

Table 2: Complexity/timings: *#vtx*, *#pol*: number of vertices/polygons. t_i: mesh initialization time in milliseconds. t_r time for one iteration in the re nement stage. n: number of iterations performed for the images in Figure 5.

tubular objects based on contour-connection using simplex meshes. The visual quality of the output mesh is improved in an optional re nement step.

ACKNOWLEDGMENTS

This work was supported by the Austrian Science Foundation (FWF) under grant **P17066-N04**.

References

[Beich04] R. Beichel, T. Pock, Ch. Janko, and et al. Liver segment approximation in CT data for surgical resection planning. In J. Michael Fitzpatrick and Milan Sonka, editors, *In SPIE Medical Imaging '04*, volume 5370, pages 1435–1446, San Diego, 2004. SPIE Press.

[Borni03] A. Bornik, R. Beichel, B. Reitinger, and et al. Computer aided liver surgery planning: An augmented reality approach. In R.L. Galloway Jr., editor, *In SPIE Medical Imaging '04*, volume 5029, pages 395–406. SPIE Press, May 2003.

[Delin99] Herve Delingette. General object reconstruction based on simplex meshes. *International Journal on Computer Vision*, 32(2):111–146, 1999.

[Felke04] Petr Felkel, Rainer Wegenkittl, and Katja Bühler. Surface models of tube trees. In *Proceeding of the Computer Graphics International CGI'04*, 2004.

[Hahn01] Horst K. Hahn, Bernhard Preim, Dirk Selle, and Heinz Otto Peitgen. Visualization and interaction techniques for the exploration of vascular structures. In *Proceedings of the conference on Visualization '01*, pages 395–402. IEEE Computer Society, 2001.

[Lewin03] T. Lewiner, H. Lopes, A. Wilson Vieira, and G. Tavares. Efficient implementation of marching cubes: Cases with topological guarantees. *Journal of Graphics Tools*, 8(2):1–15, 2003.

[Lluch04] J. Lluch, R. Vivó, and C. Monserrat. Modelling tree structures using a single polygonal mesh. *Graphical Models*, 66(2):89–101, March 2004.

[Loren87] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–169, 1987.

[Meyer92] David Meyers, Shelley Skinner, and Kenneth Sloan. Surfaces from contours. *ACM Trans. Graph.*, 11(3):228–258, 1992.

[Oliva96] J.-M. Oliva, M. Perrin, and S. Coquillart. 3D reconstruction of complex polyhedral shapes from contours using a simplified generalized voronoi diagram. *Computer Graphics Forum*, 15(3):397–408, 1996.

Modeling of Real 3D Object using Photographs

Kateřina Dařílková
Faculty of Mathematics, Physics
and Informatics
Comenius University
Mlynská Dolina
SK 84248, Bratislava, Slovakia
darilkova@sccg.sk

ABSTRACT

The goal of this work is to create several functions for processing of two input images of one object to uncover a geometry of the scene. There are well-known techniques how to compute a fundamental matrix and reconstruct 3D coordinates. Several techniques were tested to find the method that is fast and rather precise. The functions are implemented in the environment of Borland C++ Builder. Calibrated cameras and start in a situation when several points are marked correctly in both pictures are assumed.

Keywords

two view geometry, epipolar geometry, fundamental matrix, eight point method, algebraic error minimization.

1. INTRODUCTION

Computer graphics aims to model virtual reality with high realism. Several problems arise when real world objects are modeled. A model of the real object should meet some metric constraints, which are sometimes difficult or too complex to learn exactly. It is more suitable for most of situations to estimate the approximation of values. If two images of the object are available, the reconstruction of selected object points is possible, with precision up to scale, by computing some geometry around. The theory behind is called a multiple-view geometry, specifically two-view geometry.

2. BACKGROUND

The basis of theory of two-views can be dated back to year 1855. In this year french mathematician Chasles formed the problem of recovering the epipolar geometry from a seven-point correspondence. Eight years later task was solved by Hesse and in the year 1981 the original eight-point algorithm for the computation of essential matrix was introduced by Longuet-Higgins. The problem of fundamental matrix estimation is studied quite extensively from that time. A nonlinear minimization approach to estimate the essential matrix [Weng] and a distance minimization approach to compute the fundamental matrix [Luong] were described. In practice, the geometric (nonlinear) minimization approach is more reliable but computationally more expensive. Current methods improve the linear methods or accelerate the geometric minimization approach.

WSCG SHORT papers proceedings, ISBN 80-903100-9-5

WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

3. EPIPOLAR GEOMETRY

Epipolar geometry is a natural projective geometry between two views of the scene. It is usually a picture from a camera. The type and the properties of projection are given by construction and adjustment of the camera. The epipolar geometry does not depend on a structure of the scene. It is derived from intrinsic parameters of cameras and their mutual position. Taking pictures of the scene by the camera is an arbitrary projective transformation of a 3D scene in world coordinates to a 2D image. The 3*4 *matrix of projection* P represents it.

Most of usually used cameras can be approximated by a pinhole camera model. Let us define the co-ordinate system by a position and orientation of the first camera. The matrix **P** can be simplified in this coordinate system as $\mathbf{P} = \mathbf{K}\,[\mathbf{I}_{3x3} \mid \mathbf{0}]$ where **K** is a 3*3 *calibration matrix* of the camera. Using general world coordinate system the general matrix of

projection **P** has form **P = K R** [**I** |**t**]. The difference is just the orientation and translation of camera towards the world coordinate system origin. **R** is a matrix of rotation and **t** is a vector of translation, called together *extrinsic parameters of camera*.

4. FUNDAMENTAL MATRIX

The epipolar geometry arises from two images. Our goal is to find an equation, which describes the relationship between the pictures. Let us find the relation which binds an image point from the first image **x**= (x, y) with an image point from the second image **x'** = (x', y') holding a constraint that they both are projections of some point **X** from the scene. Such a matrix is called the *fundamental matrix* **F**. It is the algebraic representation of the epipolar geometry. For any pair of valid points **x**, **x'** exists a 3x3 fundamental matrix **F** for which $\mathbf{x'}^T \mathbf{F}\,\mathbf{x} = 0$. **F** is of rank 2. To make the work with matrices easier, let us set the coordinate system defined by the first camera as the world coordinate system. The first camera projection matrix is then $\mathbf{P} = \mathbf{K}\,[\mathbf{I}_{3x3}\,|\mathbf{0}]$ and the second camera matrix is **P'** = **K'** [**R** | **t**], where **R** and **t** describe a rotation and translation of the second camera towards the first one and **K** and **K'** are the calibration matrices of cameras.

Fundamental matrix and projections

The fundamental matrix depends only on the mutual position of the two cameras and their calibration. For the pair of canonic camera matrices **P** = [**I** | **0**], **P'** = [**R** | **t**] corresponding to a fundamental matrix **F** equations as follows are valid:

$$\mathbf{P} = [\mathbf{I} \mid \mathbf{0}], \quad \mathbf{P'} = [[\mathbf{e'}]_x \mathbf{F} + \mathbf{e'}\mathbf{v}^T \mid d\mathbf{e'}],$$

where **e** is an epipole (a projection of the first camera centre in the second image), $[\mathbf{e}]_x$ is a cross product, a skew-symmetric matrix, **v** is any 3-vector and d is a non-zero scalar. This result can be used only in cases when calibration matrices of both cameras are equal to identity. This can be achieved by normalization. The fundamental matrix corresponding to the normalized cameras is called *essential matrix* **E** with the form $\mathbf{E} = [\mathbf{t}]_x \mathbf{R}$. It is singular and two of its singular values are equal. There are only two possible factorizations of **E** (ignoring signs) to a skew-symmetric matrix and a rotation matrix**.** Taking into consideration direction of the translation from the first to second camera, there are four possible choices for the second camera matrix **P'**. The reconstructed point X is in front of both cameras only in one of these four solutions.

Another way how to separate the projection matrices is given by [Faugeras, Luong].

5. FUNDAMENTAL MATRIX ESTIMATION

The problem of searching for projection matrices has changed to seeking for the fundamental matrix. There exist several methods how to find it. The robust ones search for the number of corresponding features in two images and start from this statistically huge set. Our input comprises several corresponding points from two images and a calibration matrix of both cameras (usually the same). Methods used in these situations can be divided into several groups: linear algorithm, algebraic minimization algorithm, distance minimization. Until now two of them were tested.

The basic linear 8-point algorithm

The best approximation of fundamental matrix is searched. The equation which defines the fundamental matrix **F** is $\mathbf{x'}^T \mathbf{F}\,\mathbf{x} = 0$ where **x** and **x'** is a pair of the matching points in the first and the second image. Their projective coordinates are $\mathbf{x}=(x, y, 1)^T$, $\mathbf{x'}=(x', y', 1)^T$. Each point match results in one linear equation in the unknown entries of **F.** For more points matches $\mathbf{x_i} \leftrightarrow \mathbf{x'_i}$ (i = 1… n) linear equations can be stacked up into a matrix. The solution can be found by the least-square algorithm using singular value decomposition of **A**. The normalization (simple translation and scaling) of input data is very useful to make the algorithm stable. An important property of fundamental matrix is the singularity. This method, in general, does not produce matrix **F** of rank 2.

The algebraic minimization algorithm

The remaining problem is how to guarantee singularity of the constructed fundamental matrix. One possible solution is to construct the singular matrix as a product $\mathbf{F} = \mathbf{M}\,[\mathbf{e}]_x$ where M is non-singular matrix and $[\mathbf{e}]_x$ is any skew-symmetric matrix, with **e** corresponding to the epipole in the first image. To guarantee the fundamental matrix properties in such matrix **F**, a constraint on **F** is added. Matrix **F** can be computed from the image point correspondences and known epipole **e** by minimization. [Hartley, Zisserman] The estimation inaccuracy can be evaluated by an algebraic error **ε**. It describes a transformation which maps the estimate of the epipole $\mathbf{e_i}$ to the algebraic error $\boldsymbol{\varepsilon_i} : \mathbf{R}^3 \rightarrow \mathbf{R}^8$. The exact epipole is unknown, in reality. We acquire it's estimate using iterative methods. The Levenberg – Marquardt iterative method can be used [Numerical Recipes], [Pollefeys]. An estimation of the fundamental matrix $\mathbf{F_0}$ is calculated using different methods (the 8-point linear algorithm) to get the zero approximation of the epipole $\mathbf{e_0}$ (a right null vector of matrix $\mathbf{F_0}$). Each iteration aims to change $\mathbf{e_i}$ so that the value $\|\boldsymbol{\varepsilon_i}\|$ is minimized.

6. IMPLEMENTATION

The aim of our implementation is to determine a sufficiently correct method to obtain the fundamental matrix, which is fast for available data. It is assumed, that points are assigned by operator (manually).

The 8-point normalized algorithm is fast and easy to implement method. Usually, it offers quite precise results. It is very suitable as the first step for iterative methods. If higher precision is required, the algebraic error minimization method is recommended. Distance minimization method using Sampson error is appropriate as an alternative algorithm.

Methods were implemented using a Borland C++ Builder application. For calculation of a singular decomposition of matrix and inverse matrix a suitable library was chosen (Open Computer Vision Library [OpenCV]). The reconstructed points and predefined faces can be visualized in a 3D scene generated by the OpenGL library.

The images used were not made by wide-angle lens. Information about a barrel distortion is available for some of them. The elimination of deformation did not produce increase of results correctness.

Used data

A couple of images with different accuracy and resolution were used to test implemented methods. Synthetic data, pictures from tutorial of PhotoModeler [PhotoMod] and self-made pictures done by a standard camera were used. Examples of used scenes are on figure 1.

Figure 1: Scenes
"Bench" (Res. 280x1024, Foc.l. 6.97mm),
"Boxes" (Res. 2272x1704, Foc.l. 7.19mm),
"Car" (Res. 2267x1520, Foc.l. 30.75mm).

7. TESTS AND RESULTS

Error computation

To evaluate precision of acquired fundamental matrix residual error is calculated. The tests performed show the dependency of the error value on the increasing number of corresponding points. It is important to evaluate the error over a wider group of matched points, not just for the point correspondences used to compute **F** (first 8 points). This is shown in a shape of graph curves. The comparison of errors of two tested methods are displayed in figure 2.

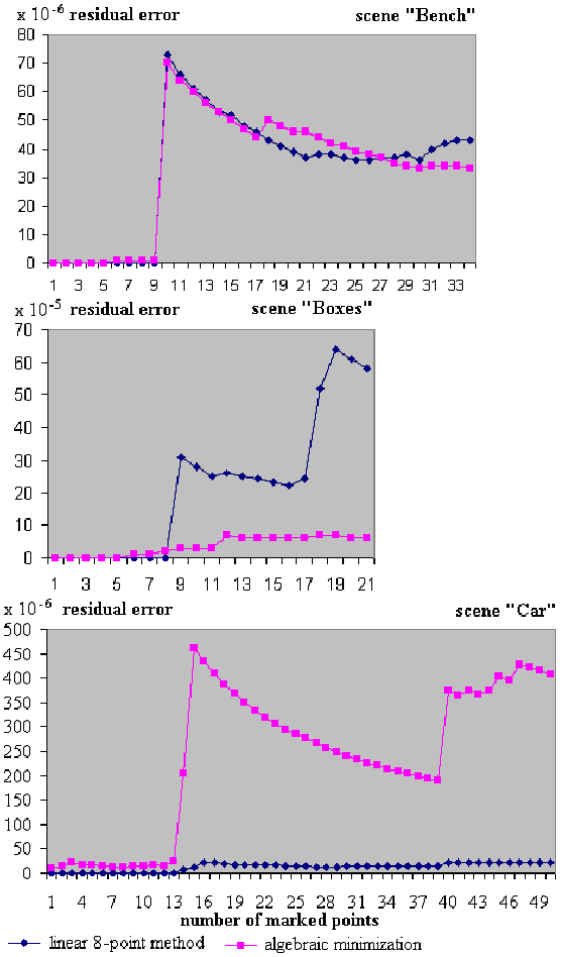


Figure 2: Graphs of methods residual error.

The graphs show the impossibility to decide positively, which method is more accurate, considering the residual error. In scenes, which are considered less stable (i.e. where the used matched points are almost coplanar) linear method works better. In scenes defined with higher precision algebraic minimization method is more accurate (e.g. in scene "Car" the result is much better).

The fundamental matrix estimate used as initial step in the first iteration is the essential part for iteration

method. If the initialization is too deflected, the method diverges or converges seemingly.

Synthetic data

An idealized scene was created to understand the methods and their convergence better (Figure 3).

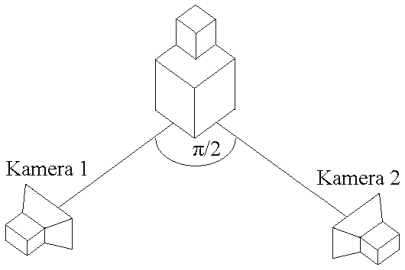


Figure 3: Synthetic scene.

The residual error was quantified here for the linear method and algebraic minimization too. Moreover, exact fundamental matrix derived from the known geometry of the scene was applied (Figure 4).

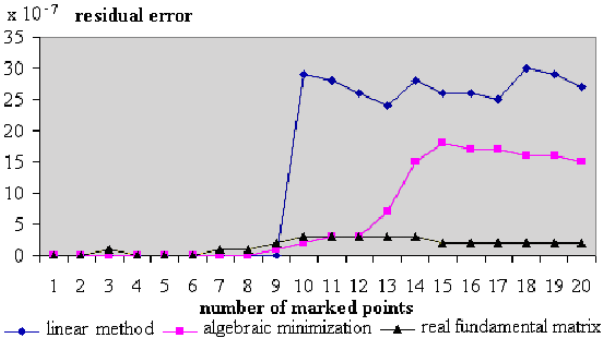


Figure 4:Graph of error for synthetic scene.

The algebraic minimization seems more suitable in this situation. All the error values are 10 times lower than in the real scenes. It is caused by much more precise selection of matching points.

Visual results

An indirect proof of the calculation accuracy is a visualization of the results as a graphical depicting of the reconstructed 3D position of the points in a virtual world. Such a visualization shows how much the reconstruction fits, that is, how the fundamental matrix fits. An experienced operator uncovers in visualization which pair of corresponding points is set incorrectly or improperly.

8. CONCLUSION

This work compares two methods for acquisition of the fundamental matrix by 8 points marked in two pictures of a scene. Linear method and the method of algebraic minimization were exploited. The presented comparisons show similarity of the methods results. Well-defined scenes have significantly better results with the fundamental matrix improved by algebraic minimization starting from a matrix given by the linear method. If the scene is described by improper set of points, the linear method is more suitable. It is possible to recognize this case by a monitoring of several properties of computation and of partial results. The choice of another set of marked points can be more efficient step to get the accurate fundamental matrix.

Standard camera suffices to take two pictures of any scene and to reconstruct its geometry. Some of camera parameters are required to be known (published usually by the producer). Much more importance is given to right choice of pictures. The user ought to arrange the scene to be heterogenous enough, to take pictures suitable for selection of 8 non-coplanar points placed in the scene uniformly. Another important condition is the precise determination of marked points in images.

9. ACKNOWLEDGEMENTS

This work is the part of project supported by APVT grant 20- 025502.

10. REFERENCES

[Faugeras, Luong] Faugeras, O., Luong, Q.-T. (2001). The geometry of multiple images. Massachusetts Institute of Technology, The MIT Press Cambridge, London, ISBN 0-262-06220-8.

[Hartley, Zisserman] Hartley, R., Zisseman, A. (2000). Multiple View Geometry in Computer Vision, Cambridge University Press, Cambridge, United Kingdom, ISBN 0 521 62304 9.

[Luong] Luong, Q.-T. (1992). Matrice Fondamentale et Calibration Visuelle sur l'Environnement-Vers une plus Grande autonomie des Systemes robotiques. PhD thesis, Université de Paris-Sud, Centre d'Orsay.

[Numerical Recipes] Numerical recipes in C: the art of scientific computing, Cambridge University Press, ISBN 0-521-43108-5

[OpenCV] Open Computer Vision Library, Copyright © 2000, Intel Corporation, www.intel.com/ research /mrl/research/opencv

[PhotoMod] PhotoModeler Pro 5.0 Demo, Copyright 1993 – 2002 Eos Systems Inc. http://www.photomodeler.com/ddlp.html, (8.9.04)

[Pollefeys] Pollefeys, M.,(2002),Visual 3D Modeling from Images, Tutorial Notes, University of North Carolina - Chapel Hill, USA, www.cs.unc.edu/ ~marc/tutorial/node160.html

[Weng] Weng, j., Ahuja, N, Huang,T. (1993). Optimal motion and structure estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(9):864-884.

Visibility-Based Simplification of Objects in 3D Scenes

Jérôme Grasset
3IL Ecole d'Ingénieurs
43 rue Ste Anne
87000 Limoges, France
grasset@3il.fr

Dimitri Pléménos
Laboratoire MSI, Université de Limoges
83 rue d'Isle
87000 Limoges, France
plemenos@unilim.fr

ABSTRACT

We present a method to perform occlusion culling on moving rigid objects in a 3D scene. This method computes potentially visible sets of polygons (PVS) from cells that are based on a bounding box of the object and have the particularity to be overlapping. The visibility is pre-processed from several renderings of the object. The implementation is easy, with a data structure that does not require additional memory compared to the original data used to describe the object. It provides interesting acceleration and smooth animation even when the list of polygons sent to the display device has to be updated. Since it is not an exact method, we discuss the errors that may occur and the ways to fix them.

Keywords
Visibility culling, Occlusion culling, Simplification, Bounding Box.

1. Introduction

The visualization of 3D scenes made of polygons uses "hidden surface removal" algorithms, like the Z-Buffer method that is integrated in common graphics cards. But, even if these hardware Z-Buffers are very fast, it is still interesting to decrease the number of polygons they have to process by determining that some are "obviously" invisible: that is the point of visibility culling. It is composed of three phases :

- view-frustum culling, to reject the objects which are outside the view frustum
- back-face culling to remove the polygons that do not face the camera , for closed opaque objects
- occlusion culling to suppress polygons that are hidden behind others.

Short papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

The first two steps are very easy to implement, the last one is much more difficult. The approach is usually to do some preprocessing to determine which polygons are potentially visible from different subspaces of the viewpoints space. Then, when the camera is in a subspace, only the potentially visible polygons associated with this subspace are sent to the display device. Subspaces are called "cells" and polygons are grouped into "Potentially Visible Sets" (PVS), a PVS being associated with each cell.

Various classifications of visibility culling methods are described in the survey by [Coh03a]. We will use the following terminology, which is slightly different but more simple and sufficient for our needs:

- **exact** method : a PVS contains all the visible elements and none of the invisible ones
- **conservative** method : a PVS contains all the visible elements, but maybe some invisible ones too
- **aggressive** method : a PVS may lack some visible elements, but it does not include any invisible one
- **approximate** method : a PVS may lack some visible elements and may include invisible ones.

2. Previous Works

One can refer to [Coh03a] for a detailed survey on occlusion culling, or [Dur00a] for the more general scope of visibility computation.

Theory – Exact Methods

The exact methods use the concept of "visual events". This is the base of aspect graphs ([Egg92a]), visibility skeleton ([Dur02a]), or the method proposed by [Nir02a].
Exact methods are, in theory, the ideal. Using their implementation experience, authors that proposed such methods explain that they are difficult to implement and require long computation time and large data structures ([Nir02a], [Dur02a]).

Non-Exact Methods

The non-exact methods come from the ideal frame and trade some precision against computation speed and implementation easiness. The usual approach here is to define the cells first and then to compute the visibility from each one. The visibility is not constant in each cell: the problem is to calculate all the polygons that can be seen from at least one viewpoint of the cell.
The first algorithms ([Tel91a], [Lue95a]), were made for architectural scenes. More recent works address general scenes. A major direction of research is the detection of good "occluders", that are polygons or sets of polygons that hide numerous other polygons (see for example [Sch00a], [Won00a]).

Application Fields

The occlusion culling methods are mainly dedicated to virtual walkthrough applications. Some properties of occlusion culling methods are to be mentioned to help situate the differences with our work:

- the global scene is static, moving objects - like cars in the streets - are not simplified
- objects that are inside a cell are entirely in the PVS of the cell
- the scene is very complex but it also has to be "highly occluded" : from the common viewpoints most of the polygons can not be seen.

3. Visibility Through Bounding Box

Differences From Existing Methods

Our method is not dedicated to simplify large complex models like a whole town, and it does not fit for such a goal. We want to simplify moving objects (for example cars), or "small" objects around which the virtual walker can move. The simplification is pre-processed for each object individually and is stored with it.
Another difference is the viewpoints space: our 3D objects are seen from the outside while the scenes of previous works are seen from their interiors.
The objects that we consider are not "highly occluded" so the simplification ratio is very different.

Visibility Pre-processing

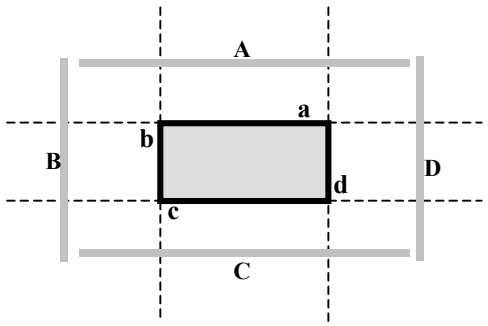


Figure 1. 2D schema of 4 half spaces (A,B,C,D) around a bounding rectangle

The viewpoints space is the whole space minus the interior of the bounding box. The cells are 6 infinite half spaces defined by the plane of each face of a bounding box of the object (see figure 1 for a 2D schema). Considering a mathematical approach, these cells do not define a partition since they overlap.
For each cell the bounding box is supposed closed except for its face lying in the plane defining the cell (see figure 2).

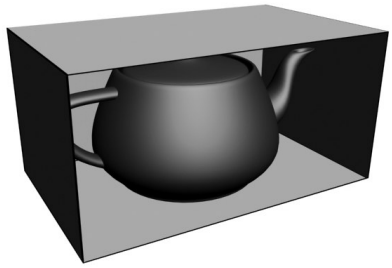


Figure 2. Bounding box with a unique open face

The first step is to compute which polygons of the object are visible through each face of the bounding box. This defines our PVS. Then, to display a scene we will only consider the polygons of the PVS associated with every face of the box that are visible from the current viewpoint. With figure 1, if the observer can see the faces 'a' and 'd' then the union of the PVS associated with areas A and B have to be displayed.

The viewpoints space is sampled. That is why our method is "aggressive": the PVS do not contain any invisible polygons, but they may miss visible ones.

For each face of the bounding box, the sampling viewpoints are disposed on the surface of a half ellipsoid. If all the viewpoints of this surface could be taken into account the PVS would be exact: there is no need to move the camera further or closer.

To pre-process the visibility for each face we use the hardware Z-Buffer. Every polygon is given a unique colour and an image is rendered from each sample viewpoint: the colours in the image denote the visible polygons. The PVS of a half space is made of the polygons detected in at least one image.

Data Structure and Display

Since the cells overlap, if we just sent "raw" PVS to the display device there would be redundancies for all the polygons visible through several faces. To avoid this we define an intermediate data structure that insures that a polygon can only be sent once to the display device for a given frame.

The idea is to store for each polygon the list of PVS it belongs to, instead of storing for each PVS the list of polygons it includes. There are 6 half spaces, so 6 PVS. Using a binary code we can associate each polygon with a 6 bits number, where a bit indicates if a polygon belongs to a PVS (see table 1). Every polygon is given a code, and there are 2^6 different codes: an easy way to store this information is to create a set of polygons for each of these codes. Then a polygon belongs to one an only one set. The storage of the pre-processed visibility is only a classification of the polygons: no extra data is needed.

Faces through which a polygon is visible	Label of the set it belongs to (6 bits)
6 faces	
0,1,2,3,4,5	111111
5 faces	
0,1,2,3,4	011111
0,1,2,3,5	101111
….	
No visible face	
/	000000

Table 1. Classification of the polygons

To display the object from any viewpoint, we first create the bit mask of the visible faces of the bounding box, using the same coding as for the PVS. Then the obtained mask is applied with a logical "and" to the label of every set of polygons: if the test succeeds the set has to be displayed.

We implemented this technique with the OpenGL display lists to represent the sets of polygons: the storage cost is reduced to the use of 64 display lists. The same has been implemented with Direct3D vertex buffers, with the same negligible cost.

4. Results

Overall Performance and Motion Fluidity

The time needed to compute which faces of the bounding box of an object are visible is definitely negligible (6 tests on the normals of these faces), so the gain in frames per second (FPS) is the same as the simplification ratio in polygons count. That is why in the following table we present only the FPS values.

Figure 3. Two objects : a tree and a "tea fountain"

Since the set of polygons sent to the display device has to be changed when the observer crosses a half-space border one could worry about potential freezing in animation. With the proposed data structure and implementation there is no delay at all, the motion is smooth. Of course if a much higher number of polygons has to be displayed the motion may slow down, but there is no extra latency induced by the change of active display lists or vertex buffers.

Pre-processing Time

In our implementation, the precomputation time depends on the size of the images rendered to pre-process the visibility because most of the time is spent reading every pixel to know which polygons have been seen. It varies from about 1 min 15 s when using 9 viewpoints by face and a resolution of 800 x 600 to about 14 min with 25 viewpoints in 1280 x 1024.

The test material is a Pentium IV 2.8Ghz with an ATI Radeon 9600 Pro.

Culling and Errors

For the results presented here we use a resolution of 1280x1024 for visibility pre-processing and 25 viewpoints for each face of the bounding box. The display resolution is 1024x768.

4.3.1 Culling results

The table 2 gives the the worst FPS, the best one, and the one obtained when the observer is in front of the

object. The percentages indicate the gain in FPS and therefore the amount of polygons that were discarded.

	without culling		with culling FPS / gain	
Object	**Faces**	**FPS**	**Worst**	**Best**
Tree	49534	72	85 - *15.2%*	117 - *38.5%*
Cup	11290	273	290 - *5.8%*	520 - *47.5%*
Ship	31216	112	145 - *22.7%*	200 - *44%*
Character	18984	175	222 - *21.1%*	455 - *61.5%*
Fountain	78984	47	77 - *39%*	300 - *84.3%*

Table 2. Culling results

Back face culling is always off, since it can not be applied to most of our objects (incompatible for example with the leafs of the tree or with the teapots).

The culling ratio directly depends on the number of faces of the bounding box that are visible. As a consequence the bounding box should be oriented to show a number of faces as small as possible from the most common viewpoints. For instance, for an object usually on the floor the bounding box should have a face on the floor.

4.3.2 Errors

Since our method is aggressive (see section 1), it is important to study the errors it makes.

First of all, with the parameters given above, most of the objects do not exhibit any error.

The first cause of errors is the sampling of the viewpoints space. When some polygons can only be seen from a very restricted area ("critical area") they are missed if the viewpoints are not in this area.

The second sampling is made when the visibility is preprocessed: we render images to detect visible polygons. If several polygons are in the same pixel only one is rendered, and thus only one is detected.

These two types of errors can be significantly reduced or suppressed, by increasing the appropriate sampling rate.

5. Conclusion and Further Works

We presented a method to perform occlusion culling on rigid 3D objects moving in a scene. Existing methods mainly focus on static scenes and do not deal with objects inside them. Our method, based on a preprocessing of visibility through a bounding box using Z-Buffer techniques, is easy to implement. It does not increase the amount of memory used by the objects since the visibility data are stored as a classification of the polygons in various display lists. The obtained culling is quite efficient. This method can be applied in any application that needs a 3D scene visualization. It can also be a complement of a method of occlusion culling of the whole scene.

Further works should aim at reducing preprocessing time and maintain the culling rate as close as possible as the best one, obtained when only one face of the bounding box is visible. Furthermore, some other particular problems should be addressed, like objects with transparent parts.

6. References

[Coh03a] Cohen-Or, D.; Chrysanthou, Y.; Silva, C.; Durand, F. A survey of visibility for walkthrough applications. IEEE Transactions on Visualization and Computer Graphics 2003

[Dur00a] Durand, F. A multidisciplinary survey of visibility. ACM Siggraph course notes Visibility, Problems, Techniques, and Applications 2000,

[Dur02a] Durand, F.; Drettakis, G.; Puech, C. The 3D visibility complex. ACM Trans. Graph. 2002, 21, 176-206

[Egg92a] Eggert, D. W. ; Bowyer D.W.; Dyer C.R Aspect graphs: State-of-the-art and applications in digital photogrammetry. In Proc.SPRS 17th Cong.: Int. Archives Photogrammetry Remote Sensing, pp 633–645, 1992.

[Lue95a] Luebke, D.; Georges, C. Portals and mirrors: simple, fast evaluation of potentially visible sets. Proceedings of the 1995 symposium on Interactive 3D graphics 1995, 105-ff.

[Nir02a] Nirenstein, S.; Blake, E.; Gain, J. Exact from-region visibility culling. Proceedings of the 13th Eurographics workshop on Rendering 2002, 191-202

[Sch00a] Schaufler, G.; Dorsey, J.; Decoret, X.; Sillion, F. Conservative volumetric visibility with occluder fusion; ACM Press/Addison-Wesley Publishing Co.: 2000; pp 229-238

[Tel91a] Teller, S. J.; Séquin, C. H. Visibility preprocessing for interactive walkthroughs. Proceedings of the 18th annual conference on Computer graphics and interactive techniques 1991, 61-70

[Wil03a] Williams, N.; Luebke, D.; Cohen, J. D.; Kelley, M.; Schubert, B. Perceptually guided simplification of lit, textured meshes; ACM Press: 2003; pp 113-121

[Won00a] Wonka P; Wimmer M; Schmalstieg D Visibility preprocessing with occluder fusion for urban walkthroughs. Rendering Techniques 2000: 11th Eurographics Workshop on Rendering, pages 71–82, June 2000.

Rendering Ghost Ships and Other Phenomena in Arctic Atmospheres

Diego Gutierrez Francisco J. Seron Adolfo Muñoz Oscar Anson
Grupo de Informática Gráfica Avanzada (GIGA), Universidad de Zaragoza
Instituto de Investigación en Ingeniería de Aragón
María de Luna 1, Edificio Ada Byron
50018, Zaragoza, Spain
{diegog|seron|orbananos|oanson@unizar.es}

ABSTRACT

The unique characteristics of the arctic atmosphere make for very interesting effects that cannot be seen anywhere else in the planet. The extremely cold temperatures, along with the existence of inversion layers in the temperature gradients, make the medium highly inhomogeneous. Its properties, including the index of refraction that rules the behavior of light, are no longer constant. As a consequence, light rays get bent while traversing the atmosphere, and the result is some spectacular phenomena; several examples are the arctic mirages (that have probably given rise to numerous ghost ship legends), the Fata Morgana or the Novaya-Zemlya effect.

We present here an implementation of a ray tracer that can render all these effects, thus depicting phenomena never ray-traced before, as far as the authors know. We first build an accurate temperature profile for the arctic atmosphere, based on experimental data, then calculate the curved paths of the light rays as the index of refraction changes as a function of temperature, by solving the physically-based differential equation that describes their trajectory. The scenes are modelled using real data for the Earth and Sun dimensions and relative distance, thus maintaining accuracy in the results obtained.

Keywords

Curved ray tracing, inhomogeneous media, arctic atmosphere

1. INTRODUCTION

Refraction of sunlight due to cold dense polar air causes a number of spectacular effects almost impossible to see anywhere else in the planet. Probably the best-known is the arctic mirage, but even that one is constantly overshadowed by its hot-air equivalent, the inferior mirage (also known as desert mirage or the "water on the road" effect). Other phenomena such as the Fata Morgana or the Novaya-Zemlya remain mostly unknown.

We present here our ray tracing method to reproduce some of these spectacular phenomena. It is based on an accurate model of the temperature profile of the atmosphere, along with a physically-based resolution method that allows us to bend light correctly. The method is obviously valid for any inhomogeneous medium, but here we concentrate on the arctic atmospheres, where weather scientists have been gathering data for decades now.

The rest of the paper is organized as follows: section 2 offers an overview of previous works on curved ray tracing. Section 3 describes how light bends in an inhomogeneous atmosphere. Section 4 presents our rendered simulations, including the model of the atmosphere and the 3D scene settings. The last two sections discuss the conclusions and the future work.

WSCG 2005 SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. PREVIOUS WORKS

There are several light effects in nature that are owed to the atmosphere being inhomogeneous, which causes the rays to travel in curved paths. There are few works which have already been published on the matter. In fact, none of them can handle generic situations where the index of refraction varies in an arbitrary way. They are either limited to a specific distribution of the index of refraction in the medium [Ber90] [Mus90] [Sta96], or they do not specify a

direct relation between the curved paths of the rays and the index of refraction [Gro95]. In this paper, we use a general ray tracing method, based on Fermat's Principle, which is valid for any inhomogeneous medium [Ser04].

3. LIGHT IN THE ATMOSPHERE

Fermat's Principle

When a medium is inhomogeneous and the index of refraction changes continuously from point to point, the trajectory of light is affected at any differential step. The final result of this differential change is a curved path that distorts the normal view of a real scene, which cannot be calculated by traditional ray tracing. We apply Fermat's principle to obtain the curved trajectory of the light rays, formulated as "light, in going between two points, traverses the route l having the smallest optical path length L". From this Principle, the equation to obtain the trajectory of a light ray in an inhomogeneous medium with a known index of refraction is [Ser04]:

$$\frac{d}{dl}\left(n\frac{d\vec{r}}{dl}\right)-\nabla n=0 \leftrightarrow \frac{d}{dl}\left(n\frac{dx_j}{dl}\right)-\frac{\partial n}{\partial x_j}=0$$

where l is the length of the arc, n is the index of refraction of the medium and $\vec{r}=x_j$ with (j=1,2,3) are the coordinates of the point. If the index of refraction n is known for each point, the equation can be solved numerically by applying an embedded Runge-Kutta Dormand-Prince method.

Inversion layers

The atmosphere is not a homogeneous medium. Properties such as pressure, temperature or density change from point to point. It is the inversion layers the one that really explains the light effects that we show in this paper. Without inversion layers, none of the effects shown here can take place. In a normal atmosphere, temperature decreases with height. An inversion layer reverses the situation, having colder air below and warmer air higher up. This obviously occurs more easily in the polar regions, where the ice or the cold seawater chills the lowest air layers. Since cold air is heavier than warm air, they tend to be very stable, lasting even for hours.

4. RENDERED SCENES

The most important step in our simulations is to obtain a correct temperature profile, and to derive from it an accurate index of refraction profile for the whole atmosphere. That index alone is what will curve the rays as they traverse the medium. For modeling the atmosphere, we have used our previsouly published Atmosphere Profile Manager [Gut04].

Scenes setup

We have designed scenes with very simple geometry, but with very accurate atmospheric profiles and using real-world distances and dimensions. The Earth is always modeled as a sphere of 6371 kilometers; the atmosphere is 40-kilometer high; the features to appear as mirages in the horizon are textured cards placed up to 26 kilometers away from the camera; the camera is finally placed ten meters above the Earth's surface. For the Novaya-Zemlya effect, the sun is a circle with a 1392000-kilometer diameter, placed 150 million kilometers away from the camera, and the camera has been placed 100 meters above sea level. All the inversion layers will have a thickness of fifteen meters, whereas their location and their gradients will be specified for each effect.

The simplicity of the whole setup is irrelevant, though, since it is the curvature of the ray traversing the medium the only factor that matters. This correct curvature is guaranteed by a) a precise atmosphere profile, b) using real geometric dimensions and c) accurately solving the physically-based equation that governs the paths of the light rays.

The images were rendered at 400x200 pixel resolution, one ray per pixel, on a Pentium IV at 2.8Ghz, and each one took between two and two and half minutes to render. The error tolerance for the numerical resolution of the equation was one centimeter.

Figure 1. Northern Bering Sea, rendered with straight rays.

Figure 2. Rendered image of the distortion in the horizon.

Figure 3. A real arctic mirage.

Figure 4. Render of the simulated arctic mirage.

Figure 5. An arctic mirage of a ship in Finland.

Figure 6. Rendered simulation of the arctic mirage

Results

When the temperature of the lower atmosphere rises at a rate of 11.2ºC per 100 meters, the light rays bend in an arc exactly equal to the curvature of the Earth, and the horizon will appear flat, like an infinite plane. If the inversion gradient becomes even stronger, the light rays exceed the curvature of the Earth, and the horizon will appear to rise vertically from the flat position. This way the **distortion of the horizon** can be seen. Figure 1 shows the scene rendered without curvature of the rays, and therefore the horizon appears undistorted. Figure 2 shows how the horizon gets distorted when taken into account the real paths that light travels in a polar atmosphere.

The **arctic mirage** occurs when there is an increase of temperature with height near the ground. As a consequence, the light rays approaching the ground are curved downward. Figures 3 and 5 show real pictures of arctic mirages while figures 4 and 6 show the respective rendered simulations. For both simulations, an inversion layer was placed centered at 100 meters, with a temperature gradient of 10ºC.

The **inferior mirage** occurs when there is a strong decrease of temperature with increasing height. As a consequence, the light rays approaching the ground are curved upwards, generating an inverted image of the object in the ground. Although less frequent, inferior mirages can sometimes be seen also in arctic latitudes. Figure 7 shows a picture of a polar inferior mirage, and figure 8 shows our rendered simulation of the phenomenon, with an inversion layer at ground level and a 5ºC gradient.

The mirage known as **Fata Morgana** occurs when there are several alternating cold and warm layers of air near the ground or the sea, causing a multiple concatenation of inferior and superior mirages, that result in a complicated superposition of images of the same object (figure 9). The number of inversion layers usually varies between two and five, with the images alternating between upright and inverted. Figure 10 shows our ray-traced image, with two inversion layers: one centered at 80 meters with a 5ºC gradient, and the second one at 120 meters with a steeper, 15ºC gradient.

Figure 7. A real inferior mirage.

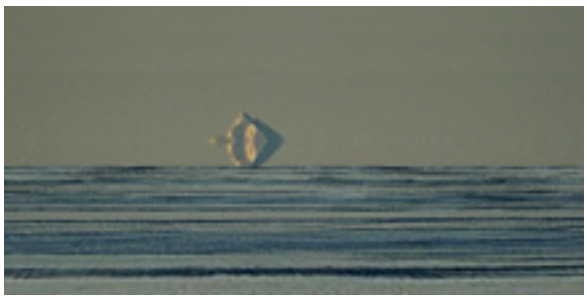


Figure 8. Render of the simulated inferior mirage.

Figure 9. Fata Morgana mirage in Greenland.

Figure 10. Rendered image of the Fata Morgana.

Figure 11. Real picture of a Novaya-Zemlya.

Figure 12. Rendered simulation of the effect

The **Novaya-Zemlya** mirage happens when celestial objects, such as the sun, can be seen even when they are situated below the horizon. Steep arctic temperature inversions trap the sunrays, making them travel within an inversion layer for hundreds of kilometers (a phenomenon known as ducting), so the light bends following the curvature of the Earth over that long distance, effectively showing up above the horizon and distorting the sun into an unusual rectangular shape. Figures 11 and 12 show a real Novaya-Zemlya effect and our ray-traced image. The inversion layer at ground level has a gradient of 11.2ºC per 100 meters.

5. CONCLUSIONS

We have presented here a work that overcomes the limitations of previous works (section 2), by accurately modeling a profile of the index of refraction in the atmosphere, based on real inversion layers data obtained by scientists doing climate research in the poles. Using also real-world dimensions for our 3D scenes, we obtain the curved paths of the rays by solving the differential equation that describes their trajectory, based on Fermat's Principle. We have simulated several of the most spectacular effects that are owed to light traveling in curved paths: the distortion of the horizon, the arctic mirage, the inferior mirage, the Fata Morgana and the Novaya-Zemlya.

6. FUTURE WORK

Still lots of work lay ahead. The number of light effects owed to inhomogeneous media does not end with the ones presented here. Adding polarization effects to our system, we could also aim at obtaining halos, sun dogs, glories... We also want to produce animations of the effects, so we need further research to find out how the index of refraction changes with time. Finally, we would like to be able to speed up the rendering times, either by solving Fermat's Principle with a faster numerical method or by implementing a parallel ray tracer.

7. ACKNOWLEDGMENTS

This research was partly done under the sponsorship of the Spanish Ministry of Education and Research through the projects TIC-2000-0426-P4-02 and TIC-2001-2392-C03-02.

The authors would like to thank Guillermo Gutiérrez for his previous research and Esther Marco for her suggestions.

8. REFERENCES

[Ber90] Berger, M., and Trout, T. Ray tracing mirages. IEEE Computer Graphics and Applications, 11(5):36-41, may 1990.

[Gro95] Groeller, E. Nonlinear ray tracing: visualizing strange worlds. Visual Computer 11(5):263-274, Springer Verlag, 1995.

[Gut04] Gutierrez, D., Seron, F.J., Anson, O., Munoz, A. Chasing the green flash: a global illumination solution for inhomogeneous media. Proceedings of the Spring Conference on Computer Graphics SCCG'04. Alexander Pasko (ed.). ISBN 80-223-1918-X, pp. 95-103, 2004.

[Mus90] Musgrave, F. K. A note on ray tracing mirages. IEEE Computer Graphics and Applications, 10(6):10-12, 1990.

[Ser04] Serón, F.J., Gutiérrez, D., Gutiérrez, G. and Cerezo, E. Visualizing sunsets through inhomogeneous atmospheres. Proceedings of Computer Graphics International '04 (ed. IEEE Computer Society), pp. 349-356, 2004.

[Sta96] Stam, J. and Languenou, E. Ray tracing in non-constant media. Proceedings of Rendering Techniques '96, pp. 225-234, 1996.

Fresnel Term Approximations for Metals

István Lazányi
Budapest University of Technology
Magyar Tudósok Krt. 2
H-1117, Budapest, Hungary
ilazanyi@freemail.hu

László Szirmay-Kalos
Budapest University of Technology
Magyar Tudósok Krt. 2
H-1117, Budapest, Hungary
szirmay@iit.bme.hu

ABSTRACT

Colored reflections are governed by the Fresnel term, which can be expressed from the refraction index of the material. This function, especially for metals where the refraction index becomes a complex number, is rather computation intensive. This paper presents an accurate simplification, which can also cope with complex refraction indices. In order to establish the approximation, Schlick's formula is rescaled and the residual error is compensated by a simple rational approximation. The resulting formula can present realistic metals and is simple enough to be implemented on the vertex or pixel shader, and used in games.

Keywords: BRDF, Fresnel term, Schlick's approximation

1. INTRODUCTION

Material models are usually defined by Bidirectional Reflectance Distribution Functions (BRDFs) that describe the chance of reflection for different pairs of incoming and outgoing light directions. Programmable vertex and pixel shaders allow sophisticated material models instead of the simple Phong-Blinn reflection model. Unlike the Phong-Blinn model, these sophisticated models can be physically plausible, that is, they do not violate basic rules of optics, including the Helmholtz symmetry and energy conservation.

2. PHYSICALLY PLAUSIBLE BRDF MODELS

A microfacet based specular BRDF model usually has the following product form [Cook81, He91]:

$$P(\vec{N}\cdot\vec{H})\cdot G(\vec{L},\vec{N},\vec{V})\cdot F(\vec{L}\cdot\vec{H},\lambda),$$

where λ is the wavelength of light, $\vec{N}$ is the surface normal, $\vec{L}$ is the illumination direction, $\vec{V}$ is the viewing direction and $\vec{H}$ is the halfway vector between the illumination and viewing directions.

Microfacet distribution $P(\vec{N}\cdot\vec{H})$ defines the roughness of the surface by describing the density of microfacets that can ideally reflect from the illumination direction to the viewing direction.

SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005,
Plzen, Czech Republic.

Geometric term $G(\vec{L},\vec{N},\vec{V})$ shows how much of these ideal reflections can actually occur, and is not blocked by another microfacet (called masking or self-shadowing). The geometric term is independent of the material properties, and it causes a general reduction of the specular term for certain illumination and viewing directions. Such reduction should be compensated in the diffuse reflection, since we can assume that photons reflected on the microfacets by multiple times contribute to the *diffuse* (also called *matte*) part. It also means that the matte and specular parts are not independent, as assumed by most of the BRDF models, but are coupled by an appropriate weighting, which depends on the viewing direction [Ashik00, Kele01].

Fresnel term

Finally, Fresnel term $F(\vec{L}\cdot\vec{H},\lambda)$ equals to the probability that a photon is reflected from the microfacet considered as an ideal mirror. According to the law of ideal reflection, the normal of those microfacets that can reflect from illumination direction $\vec{L}$ to viewing direction $\vec{V}$ is exactly the halfway vector $\vec{H}=(\vec{V}+\vec{L})/|\vec{V}+\vec{L}|$. This is why we included the angle of the halfway vector and the light direction in the Fresnel function, which depends on the angle of light incidence.The Fresnel term is the only factor that is wavelength dependent, thus it is the primary source of coloring. That is why the accurate computation of the Fresnel term is so important to present realistic look for materials.

The Fresnel term can be obtained as the solution of the Maxwell equations assuming an ideal planar surface. The formula of an arbitrary polarization can be expressed from two basic solutions, when the oscillation is parallel or perpendicular to the surface.

In the final result for these cases, the complex refraction index $n+kj$ also plays a crucial role:

$$F_p = \left| \frac{\cos\theta - (n+kj)\cos\theta'}{\cos\theta + (n+kj)\cos\theta'} \right|^2, \quad F_s = \left| \frac{\cos\theta' - (n+kj)\cos\theta}{\cos\theta' + (n+kj)\cos\theta} \right|^2$$

In these formulae, θ is the angle of incidence and θ' is the angle of refraction. Assuming that the light is not polarized, the ratio of the reflected and incident radiance can be expressed by the Pythagoras theorem: $F = \frac{F_s + F_p}{2}$.

Using Snell's law, the angle of refraction can be eliminated from the formula, thus the Fresnel term depends on three arguments: n and k are the real and imaginary parts of the refraction index, respectively, and θ is the angle of incidence for the given microfacet. Expressing the absolute value of the complex numbers, the following form of the Fresnel formula can be obtained [Glass95].

$$F_s = \frac{a^2 + b^2 - 2a\cos\theta + \cos^2\theta}{a^2 + b^2 + 2a\cos\theta + \cos^2\theta}$$

$$F_p = F_s \frac{a^2 + b^2 - 2a\sin\theta\tan\theta + \sin^2\theta\tan^2\theta}{a^2 + b^2 + 2a\sin\theta\tan\theta + \sin^2\theta\tan^2\theta}$$

where a and b are defined by the following equations:

$$2a^2 = \sqrt{(n^2 - k^2 - \sin^2\theta)^2 + 4n^2k^2} + (n^2 - k^2 - \sin^2\theta)$$

$$2b^2 = \sqrt{(n^2 - k^2 - \sin^2\theta)^2 + 4n^2k^2} - (n^2 - k^2 - \sin^2\theta)$$

The computation of the exact Fresnel is quite expensive even on the graphics hardware. In real time applications we need its approximation, which is much cheaper to evaluate, but is accurate enough not to destroy image quality [NV02]. The main objective of this paper is to propose such Fresnel approximations.

3. PREVIOUS WORK ON FRESNEL TERM APPROXIMATION

For many non-metallic materials, the extinction coefficient k is quite small, which allows us to ignore the imaginary part altogether. The assumption of the extinction coefficient being zero has also been made by Schlick, who has found the following simple rational approximation for the Fresnel term [Schl94]:

$$FSchlick(n, \cos\theta) = \frac{(n-1)^2 + 4n(1-\cos\theta)^5}{(n+1)^2}$$

This formula provides a fairly good approximation if the extinction coefficient is really zero (k=0), and n is in the range 1.4...2.2 (outside this range the relative error increases sharply, see Figure 5).

Now let us concentrate on metals or other materials having complex refraction index ($k > 0$) (Figure 1).

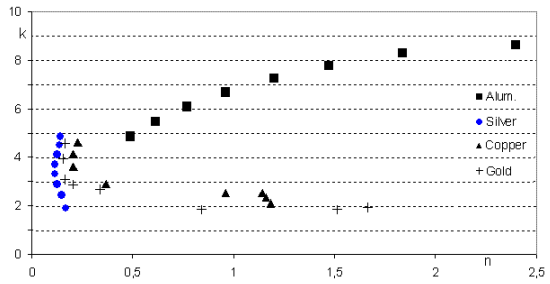


Figure 1. Complex refraction indices for different metals, for different wavelengths (λ=400..800 nm)

For complex refraction indices, the Fresnel term can have various characteristics (Figure 2). For small extinction coefficients, for example, the Fresnel term is a monotonous function of $\cos\theta$, with increasing values at $\cos\theta$=1 when k is increasing. Furthermore, for larger k values there exists a local minimum.

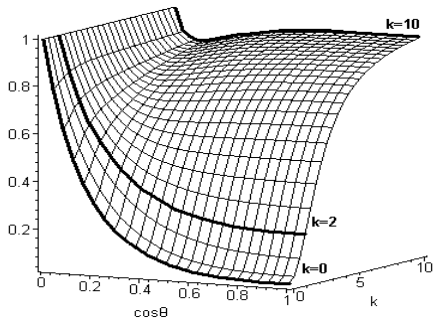


Figure 2. The Fresnel function (n=1.5, k=0..10)

If we apply Schlick's formula for metals, the result will be erroneous, with a significantly large error at $\cos\theta$=1 (Figure 3 and Figure 4).

In the followings we propose two improvements to significantly reduce the error of the original Schlick's approximation.

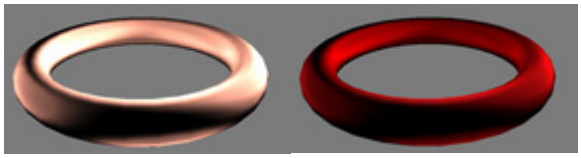


Figure 3. A copper ring rendered with the original Fresnel term and with Schlick's approximation

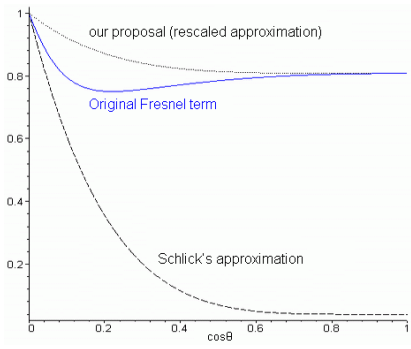


Figure 4. The Fresnel term, Schlick's formula and our new approximation for n=1.5, k=5

4. THE NEW FRESNEL TERM APPROXIMATION FOR METALS

Rescaling the Schlick's function

To reduce the error of the approximation we shall rescale Schlick's formula so that it will obey the value of the original function not only at $\cos\theta=0$, but at $\cos\theta=1$ as well (Figure 4).

To achieve this, we compute the exact and approximated function values at $\cos\theta=1$, as follows:

$$F_1 := 1 - Fresnel(n,k,\cos\theta = 1) = \frac{4n}{(n+1)^2+k^2}$$

$$S_1 := 1 - FSchlick(n,\cos\theta = 1) = 1 - \frac{(n-1)^2}{(n+1)^2} = \frac{4n}{(n+1)^2}$$

Thus the scaling factor is: $\frac{F_1}{S_1} = \frac{(n+1)^2}{(n+1)^2+k^2}$.

Now the modified (rescaled) Schlick's formula can be expressed as follows:

$$F^*(n,k,\cos\theta) := 1 - \frac{F_1}{S_1}\left(1 - FSchlick(n,\cos\theta)\right) =$$

$$= 1 - \frac{4n(1-(1-\cos\theta)^5)}{(n+1)^2+k^2} = \frac{(n-1)^2 + 4n(1-\cos\theta)^5 + k^2}{(n+1)^2+k^2}$$

The resulting formula is able to deal with complex refraction indices and is simple enough for practical applications:

$$F^*(n,k,\cos\theta) := \frac{(n-1)^2 + 4n(1-\cos\theta)^5 + k^2}{(n+1)^2+k^2}$$

After examining the relative error of the modified approximation (Figure 5) we can conclude that this simple modification enables us to extend the original Schlick's formula to complex refraction indices without significant increase of the relative error (compare error for k=0 and k>0). Note that for k=0 we get back Schlick's original formula.

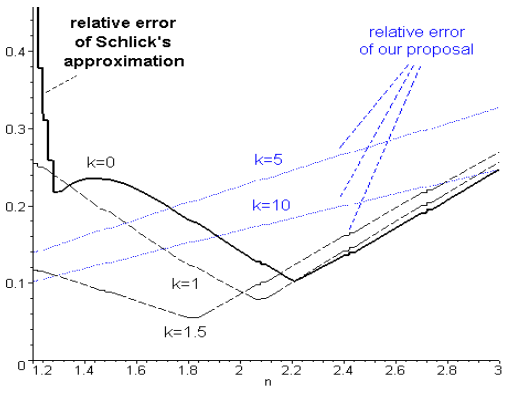


Figure 5. Relative error of the rescaled Schlick's formula for different *n* and *k* values

Error compensation

Since our approximation is unable to follow the local minimum of the original Fresnel function, metals with large (n, k) values (e.g. aluminum) may have a noticeable error. To improve our approximation, we shall compensate the error term – the difference between the original Fresnel function and the rescaled Schlick's formula – with a rational approximation that is simple enough to enable fast calculation and its shape is close to the shape of the error function (Figure 6). After examining the error term we chose the following approximation:

$$-ax(1-x)^{\alpha},$$

where $x=\cos\theta$. The derivative of this expression at x=0 and 1 equals to $-a$ and 0, respectively. The expression has a local minimum at $x = 1/(1+\alpha)$.

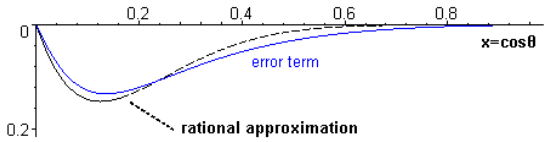


Figure 6. Error term for *n*=1.5, *k*=5 and a rational approximation with parameters *a*=3 and *α*=7

To determine the parameters a and α, we will have to set up two constraints.

Unfortunately, we cannot easily reproduce the local minimum of the error term. However, for most (n, k) values, the local minimum of the error term is located somewhere in the range of 0.1..0.15. (The only exceptions are those (n, k) pairs where both n and k are small, but in these cases – as we have seen earlier – the error term is negligible.) So the rational approximation should obey the value of the original error function at $x=\cos\theta=0.15$ (which is close to its local minimum).

Our second constraint will be the equality of the derivatives at x=0. For large k values – that is where error compensation becomes necessary – we can use the following approximation: $-a = -2n$, where value $-2n$ is a quite good approximation for the derivative of the original Fresnel term at x=0. (The derivative of the modified Schlick's formula is assumed to be zero.) As shown below, the relative error can be reduced below 5% even with this assumption, so there is no need to calculate the exact derivative. Note the dramatic improvement in case of aluminum.

relative error	rescaling only	approx. derivative	exact derivative
copper	4,9%	**4,1%**	1,9%
gold	5,4%	**5,0%**	5,2%
silver	9,0%	**5,1%**	4,4%
alu	17,5%	**2,6%**	2,0%

Values a and α should be regarded as derived material properties. When the CPU program instructs the GPU to use a different material model, it should compute these parameters and download them to the GPU as uniform variables.

According to our tests, the proposed simplifications do not reduce image quality, but reduce rendering times to *their half or third*. The following table summarizes the relative computation times:

Fresnel formula	100%
Schlick's formula	28%
Rescaled Schlick's formula	33%
Error compensation	54%

5. CONCLUSIONS

This paper proposed two new approximations of the Fresnel function. Unlike previous approaches, we did not assume that the imaginary part of the refraction index is negligible, thus our model can be applied for a wider range of materials, e.g. metals. Our two approximations differ in accuracy and cost of evaluation. The more accurate approximation (called "error compensation") is worth applying if there is a significant back lighting in the scene. In other cases the rescaled model provides satisfactory results.

6. REFERENCES

[Ashik00] Ashikhmin, M. and Premoze, S. and Shirley, P. A Microfacet-based BRDF Generator, SIGGRAPH'00, 2000.

[Cook81] Cook, R. and Torrance, K. A Reflectance Model for Computer Graphics, Computer Graphics, Vol. 15, Num. 3, 1981.

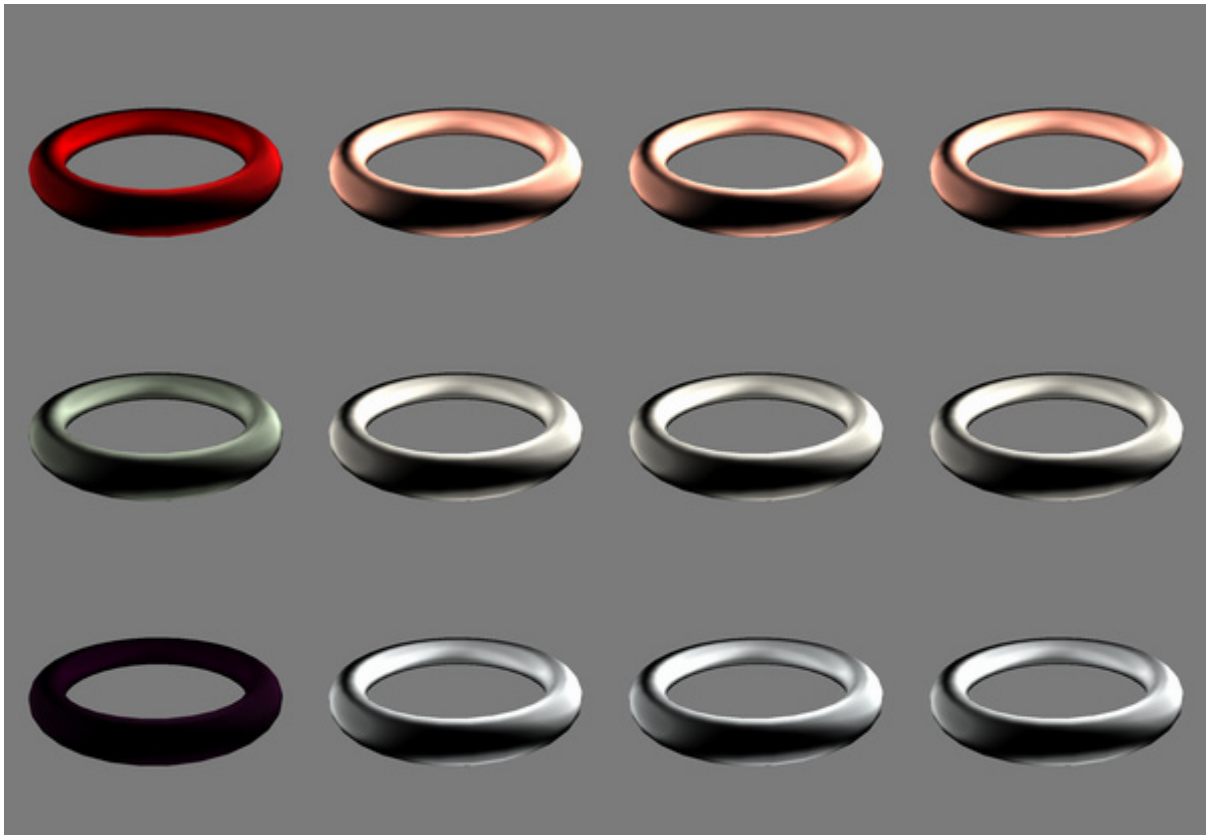
[Glass95] A.S.Glassner. Principles of Digital Image Synthesis. Vol. 2. Morgan Kaufmann Publishers, Inc, San Francisco, 1995. ISBN 1-55860-276-3.

[He91] He, X. and Torrance, K. and Sillion, F. and Greenberg, D. A Comprehensive Physical Model for Light Reflection, Computer Graphics, Vol. 25, Num. 4, 1991.

[Kele01] Cs. Kelemen and L. Szirmay-Kalos. A Microfacet Based Coupled Specular-Matte BRDF Model with Importance Sampling. Short Presentations, EUROGRAPHICS 2001.

[NV02] Fresnel Reflection. NVidia Technical Report, July 2002. (http://developer.nvidia.com/attach/6664)

[Schl94] Ch. Schlick. An inexpensive BRDF model for physically-based rendering. Proceeding of the International Conference Eurographics '94.



Schlick's model | Rescaled model | Compensated model | Exact Fresnel term

Figure 7. Copper, silver and aluminum rings rendered with different Fresnel approximations.

Snapshot: A rapid technique for driving a selective global illumination renderer

Peter Longhurst
University of Bristol
Merchant Venturers Building
Woodland Road
BS8 1UB, Bristol, UK
Pete.longhurst@bristol.ac.uk

Kurt Debattista
University of Bristol
Merchant Venturers Building
Woodland Road
BS8 1UB, Bristol, UK
debattis@cs.bris.ac.uk

Alan Chalmers
University of Bristol
Merchant Venturers Building
Woodland Road
BS8 1UB, Bristol, UK
alan@cs.bris.ac.uk

ABSTRACT

Even with modern graphics hardware, it is still not possible to achieve high fidelity global illumination renderings of complex scenes in real time. However, as these images are produced for human observers, we may exploit the fact that not everything is perceived when viewing the scene with our eyes. We are drawn to certain salient areas of an image. Taking this into account, it is possible to selectively render parts of an image at high quality and the rest of the scene at lower quality without the user being aware of this difference.

Methods exist for calculating which parts of an image are perceptually important, but generally they rely on having a fully rendered image to process. It is thus only possible to prioritise pixels to speed up the rendering of a frame once that frame has been rendered: an obvious catch. In pre-scripted animated sequences it is indeed possible to use rendered key frames to extract the necessary information, however, the cost of rendering such key frames could be significant and this is not appropriate for any interactive application. This paper presents a high speed OpenGL generated *"Snapshot"* of a frame to generate a saliency map to efficiently drive the selective global illumination rendering of an animated sequence.

Keywords

High Fidelity Graphics, Human Visual Perception, Selective Rendering, Interactive Rendering of Dynamic Scenes.

1. INTRODUCTION

The aim of realistic image synthesis is to produce a high fidelity reproduction that faithfully represents the real scene it is attempting to portray. Such full global illumination solutions can be computationally very costly. Recently, traditional rendering algorithms have been modified in order to selectively spend more time rendering perceptually important pixels at the highest quality, while the remainder of the image, which is not seen by the human viewer, can be rendered at a significantly lower quality [Cater03, Yee01]. Such selective renderers have been shown to significantly improve rendering times without the viewer being aware of the different qualities within the image. A major problem still remains however, and that is: how to rapidly identify the different qualities at which pixels should be rendered?

WSCG 2005 SHORT papers proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

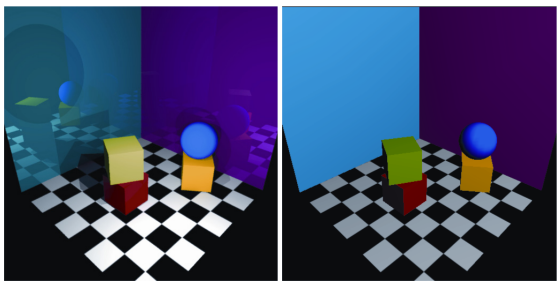


Figure 1: (a) Full global illumination image - 382 seconds (b) OpenGL Snapshot - 2 milliseconds.

The quality to which pixels in an image are to be rendered can be prioritised based on importance criteria. When considering ray tracing, the number of samples needed per pixel can be adjusted according to the saliency of that area of the scene. More rays

should be traced to achieve the desired quality in those salient areas where an individual's attention is likely to be focused. This measure requires knowing information on a per pixel basis, but until the scene is rendered this information is unknown.

An alternative to actually tracing the rays, is to develop methods to extract the necessary information from an interpretation of the scene geometry. OpenGL is a rapid approach that uses modern graphics hardware to draw 3D geometry.

Figure 1 shows a high speed OpenGL image of a scene, which we term a *Snapshot*, (achieved in 2ms) compared with full global illumination solution (382 seconds). This paper investigates how an indication of saliency can be achieved rapidly using such an OpenGL Snapshot.

2. PREVIOUS WORK

When viewing a scene, the human visual system will shift attention around the scene, selecting in turn the available visual information for localisation, identification and understanding of objects in the environment. During this process, more attention is given to salient locations, for example, a red apple in a green tree, and less attention to unimportant regions, so that detail in many parts of the scene can literally go unnoticed [Yarbus67]. Visual perception techniques are increasingly being used to improve the perceptual quality of rendered images, including [Bolin98, Luebke01, and Volevich00]

Previous selective rendering solutions using saliency have used the model of the human visual system proposed by Itti and Koch [Itti00, Itti98, and Koch85]. Some of these techniques have been developed for pre-composed animation sequences to determine areas of perceptual importance within key-framed sections, for example [Myskowski01].

Rendering time for animations has been reduced by saving computational effort on non-salient areas. This works well as salient areas for those frames between key-frames can be quickly interpolated, however, the key frames themselves have to be fully rendered. The drawback of this is that these methods cannot be used for interactive rendering scenarios.

Saliency was also used by Yee et al [Yee01] to accelerate animation renderings with global illumination. As with our approach, they use an initial OpenGL image and apply a model of visual attention to identify conspicuous regions. Yee et al. then construct, for each frame in the animation, a spatiotemporal error tolerance map called the Aleph map from spatiotemporal contrast sensitivity and a low-level saliency map. The Aleph map is then used as a guide to indicate where more rendering effort should be spent, significantly improving the computational efficiency during the animation. However, such a map takes several seconds to compute.

3. THE SNAPSHOT

We chose OpenGL as a basis for our Snapshot because it is well supported in hardware, fast and cross platform. It is designed to read model data from a Wavefront ".obj" file, an established format which many renderers can read [Alias].

Shadows and Reflections

Figure 1 shows how the simple Snapshot does not contain any of the shadow or reflection information which is present in the scene. Significant potential salient information is thus missing from this simple image. To overcome this, we have included simple shadowing and reflection calculations in Snapshot, as can be seen in Figure 2.

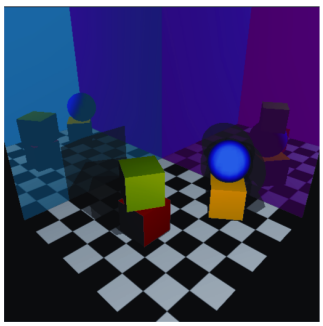


Figure 2: Snapshot with shadowing and reflections.

In OpenGL individual surfaces in a scene are drawn under an approximation of direct illumination. This shading is simply calculated on the vertex normals of a surface in relation to a lighting model, no account is taken for surface occlusions. In order to obtain shadows, manual calculations have to be undertaken to project occluding geometry onto surfaces. A standard method that we choose to do this involves observing the scene from the point of view of the light source. From this view surfaces to be shadowed can masked off, the rest of the scene can then be traced as shadow into this area [Kilgard99]. The drawback of this technique is that the scene needs to be drawn multiple times depending on the number of objects and lights present. Similarly, reflections require the camera to be moved to a projected position and re-rendering the scene on the mirror plane. To cut the total number of drawings down, we chose to render shadows and reflections solely on large planar surfaces where they will probably be most apparent.

Figure 3: Radiance saliency (left) vs. Snapshot saliency (right) with difference map (centre). (frame #125)

Reflected Light Sources

The biggest problem with the aforementioned approximations is that there is no account taken for indirect illumination. In a scene with many reflective surfaces or mirrors a significant contribution to the lighting of an object will come indirectly from reflections of light. To account partially for this reflected light, sources are added to the scene as additional light sources with reduced emission. The reflected position of each light source given each mirrored surface is calculated. These extra lights are then positioned with their emission components reduced by the reflectivity of the mirror. Each additional light will simply directly illuminate each surface in the scene. This extra contribution is proportionally greatest on surfaces which face none of the primary light sources.

4. SALIENCY COMPARISONS

In order to investigate the potential use of Snapshot in selective rendering we determined the saliency of the frames. For this we used the Itti and Koch method implemented in the iLab toolkit [Ilab]. We generated a saliency map for every Snapshot frame and for each Radiance rendered image. The absolute pixel difference was found and averaged across the image. This gave a measure of saliency difference between the images. Figure 3 shows the error for one frame of the animation. In these images the brightest areas represent the areas of greatest saliency. The Radiance image shows how, when properly calculated, indirect illumination adds salient information. Figure 4 shows how the error changed through the animation. The lower line on the graph demonstrates that adding shadows and reflections decreases this error. For the most part, the lines follow a similar path; however, for the complex part of animation it is clear that the Snapshot without shadows and reflections produces a significantly worse result.

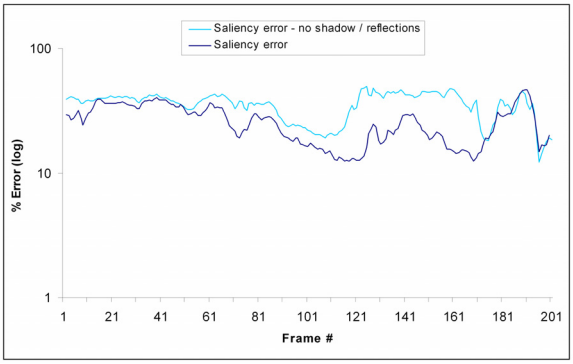


Figure 4: Saliency error during course of animation.

Table 1 shows numerically the saliency error for our two example frames. As previously shown in Figure 4, the error in saliency is always less when approximated shadows and reflections are added. In areas consisting of many such artefacts this difference is maximised, for the example complex frame the error almost halves. When the entire animation is considered this difference is not so great, but it is still significant. When there are no shadows or reflections the average error is 35%, this drops to 26% when these are added. A standard statistical analysis shows the difference of these means to be highly significant (the probability that the sets are then same < 0.001).

	Error
SF: no Shadows/Reflections/Texture	43%
SF: Textures	39%
SF: Textures + Shadows	37%
CF: no Shadows/Reflections/Texture	48%
CF: Textures	37%
CF: Textures + Shadows + Reflections	19%
Animation Average	26%
Animation Avg no shadows/reflections	35%

Table 1: Percentage error: Global illumination vs. Snapshot saliency.

Comparison of Saliency Map Prioritisation Based Rendering

To check the validity of using a saliency map based on our scene estimation to generate a full global illumination render we used our selective renderer, SharpEye, which allows us to manipulate the number of rays traced per pixel. For this experiment, we computed a reference image in which every pixel was dictated by the saliency map shooting a maximum of 9 rays per pixel. Further images were based on the generated saliency maps: from the full global illumination solution, the simple Snapshot and the more complex Snapshot.

The full global illumination solution took over 7 minutes to compute, whereas the selective rendered image using the saliency map generated from the full solution took only 2 minutes 11 seconds. Of particular interest is the selective rendering based on the saliency maps for the simple Snapshot took 2 minutes and 33 seconds while the Snapshot with shadows and reflections took 2 minutes 6 seconds, even faster than saliency map from the full solution. The VDP perceptual error (for the frame considered) for all selective renderings is less than 0.5% [Daly93].

5. CONCLUSIONS

Selectively rendering a global illumination computation can significantly reduce the time taken to render a scene without affecting the perceived quality of the resultant image. In the tests conducted in this paper, we have found that, in many cases, a simple OpenGL Snapshot does closely match an equivalent global illumination image in terms of saliency. The addition of approximate shadows and reflections to the Snapshot, although adding to the Snapshot computation time (from 2ms to 14ms), significantly increased the saliency correspondence between the Snapshot and the full global illumination solution. This Snapshot was then able to successfully drive the selective renderer, achieving a high perceptual fidelity between the selective rendered images and the full solutions.

Future work will use eye-tracking to further verify the possibility of using Snapshot as a means of driving a selective global illumination renderer. By examining these eyetracking results in conjunction with the saliency maps we hope to establish where the most significant flaws are in using OpenGL as a fast preliminary renderer. We will also consider more complicated scenes.

6. REFERENCES

[Alias] Alias wavefront: http://www.aw.sgi.com/.

[Bolin98] M. R. Bolin and G. W. Meyer. A perceptually based adaptive sampling algorithm. *Computer Graphics*, 32(Annual Conference Series):299-309, 1998.

[Cater03] K. Cater, A. Chalmers, and G. Ward. Detail to attention: Exploiting visual tasks for selective rendering. *Eurographics Rendering Symposium*, (Annual Conference Series), 270-280, 2003.

[Daly93] S. Daly. The visible differences predictor: An algorithm for the assessment of image fidelity. In A. B. Watson, editor, *Digital Images and Human Vision*, pages 179-206, Cambridge, MA, 1993. MIT Press.

[Haines86] E. A. Haines and D. P. Greenberg. The light buffer: a shadow testing accelerator. *IEEE Computer Graphics and Applications*, 6(9):6-16, 1986.

[Ilab] ilab neuromorphic vision c++ toolkit: http://ilab.usc.edu/toolkit/.

[Itti00] L. Itti and C. Koch. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision research*, 10(10-12):1489-1506, 2000.

[Itti98] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. In *Pattern Analysis and Machine Intelligence*, volume 20, pages 1254-1259, 1998.

[Kilgard99] M. Kilgard. Creating reflections and shadows using stencil buffers. In *GDC 99*, 1999.

[Koch85] C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. In *Human Neurobiology*, volume 4, pages 219-227, 1985.

[Lee85] M. E. Lee, R. A. Redner, and S. P. Uselton. Statistically optimized sampling for distributed ray tracing. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 61-68. ACM Press, 1985.

[Luebke01] D. Luebke and B. Hallen. Perceptually driven simplification for interactive rendering. *Rendering Techniques*, 2001.

[Myskowski01] K. Myskowski, T. Tawara, H. Akamine, and H. Seidel. Perception-guided global illumination solution for animation rendering. *SIGGRAPH 2001 Conference Proceedings*, pages 221-230, 2001.

[Volevich00] V. Volevich, K. Myszkowski, A. Khodulev, and E. Kopylov. Measuring and using the visual differences predictor to improve performance of progressive global illumination computation. *In Transactions on Graphics*, 19:122-161, 2000.

[Ward98] G. Ward and R. A. Shakespeare. *Rendering with Radiance*. Morgan Kaufmann Publishers, 1998.

[Weghorst84] H. Weghorst, G. Hooper, and D. P. Greenberg. Improved computational methods for ray tracing. In *ACM Transactions on Graphics*, pages 52-69. ACM Press, 1984.

[Yarbus67] A. Yarbus. Eye movements during perception of complex objects. In *Eye Movements and Vision*, pages 171-196, 1967.

[Yee01] H. Yee, S. Pattanaik, and D. P. Greenberg. Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. In *ACM Transactions on Graphics*, pages 39-65. ACM Press, 2001.

Automatic Visual Alignment Using Planar Regional Features and Stereo Vision

Forster, Carlos H. Q.
UNICAMP - Universidade Estadual de Campinas
FEEC, DCA, CP6101
13081-970, Campinas, SP Brazil
forster@dca.fee.unicamp.br

Tozzi, Clésio L.
UNICAMP - Universidade Estadual de Campinas
FEEC, DCA, CP6101
13081-970, Campinas, SP Brazil
clesio@dca.fee.unicamp.br

ABSTRACT

This paper addresses the determination of the rigid transformation between camera and object reference frames from a pair of intensity images and a known scene model. Two difficult parts of this problem that deserve particular attention are the matching between image and model features and the matching of image-features between stereo views. We propose the use of planar regions as features, what make both problems simpler. The former is handled by an invariant-based approach, for which a less complex base can be adopted, and the latter, by applying the epipolar constraint for inferior and superior bounds of region coordinates. The presented approach may be useful in many applications where camera-based tracking requires automatic initialization.

Keywords

Invariant-based matching, stereo vision, regional features, tracking, object recognition.

1. INTRODUCTION

In general terms, alignment is the determination of a transformation which can be used to relate and convert measurements between two reference frames. The relationship between camera and object reference frames is represented by a rigid or isometric transformation, composed of transformations of translation and rotation and represented by a six-parameter vector, representing position and orientation information, called pose. When camera intrinsic parameters are known, one can easily find the relative pose between camera and object provided that a set of correspondences between points from the camera image and from the object model are established. The difficulty arises when it is necessary to discover automatically the correspondences between these points.

Short paper proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

The objective of this paper is to propose an approach to visual alignment based on the design of a set of invariants for matching, a matching algorithm and complementary techniques. Matching based on invariants is recognized as a successful means for creating efficient and robust matching algorithms [Forsyth *et al.*, 1991].

Our proposal for automatic visual alignment consists in considering regional features as a source of information for the alignment and in obtaining these features from a binocular image acquisition system for which camera relative orientation and intrinsic parameters are known. The combination of stereo-vision and regional-features presents the following synergistic benefits:

(1) The matching between image- and model-features is simplified through the adoption of the rigid 3D transformation for the alignment model. It is then possible to avoid the inconveniences of the planar projective transformation in 2D, such as non-linearity, deformation and lack of simple invariants. In the adopted case, it is easier to find invariants due to the preservation of distances and angles and due to its linearity.

(2) Solution does not depend on region shape. As a result, symmetric regions are not problematic, a less strict segmentation procedure can be adopted and the contour of the convex hulls can be used instead of the original region contours.

(3) Stereo matching is simplified. We show that region-matching between 2 rectified views is very simple using the epipolar constraint in contrast to point-matching approaches. A straightforward algorithm can reconstruct features in 3D.

Possible applications for the proposed approach are robotic navigation, tangible interfaces and augmented reality. In these scenarios, it is necessary to discover the pose of known objects or scenes without user intervention, responding in interaction time and robust against occlusion and abrupt motion.

2. REVIEW OF USUAL APPROACHES

The usual approaches to automatic determination of correspondences which avoid high complexity are user intervention (the user marks the correspondences interactively), incremental solution (an image sequence with little variation between frames and known initial state are required) [Koller, 1993] [Simon and Berger, 1997] [Cornelis *et al*., 2000], predictive multiple-hypothesis solution (less sensitive to abrupt motion and clutter) [Fox *et al,*. 1999], the hybrid sensorial approach (additional non-optical sensors) [Azuma, 1999], artificial landmarks (for environments where objects can be tagged) [Bajura and Neumann, 1995], appearance approaches (use pixel neighbourhood information to compare images to a view-based model) [Se *et al.,* 2002], shape matching and search in the spaces of parameters or correspondences with the assumption of a fixed parametric form (usually approached by some form of clustering) [Haustler and Ritter, 1999] [Olson, 1994] [Wolfson and Rigoutsos, 1997].

3. ELEMENTS OF OUR APPROACH

We propose a simple 3D reconstruction technique for image-features that allows the extraction of feature attributes to form invariants. These invariants are then used to match model-features to image-features through a recognition algorithm.

Reconstruction

The 3D reconstruction of the contours based on a stereo image pair demands matching of features from the right-hand and the left-hand images and also finding homologous contour points.

As the relative orientation of the camera pair is known, a rectification step can be applied to polygon vertex coordinates in order to have the same *y*-coordinate for homologous points. With rectified coordinates, stereo matching of regions becomes a simple problem as each polygon representing a region is described by a pair of values corresponding to its maximal and minimal *y*-coordinate.

Homologous polygons must present the same pair of values except for their imprecision. Due to the additional imposed constraints, ambiguities are less frequent in the region matching case than when matching points. A hash based implementation of the stereo matching is illustrated in figure 1, the table is constructed with information from features from the left-hand image and it is indexed by the attributes from the right-hand image. In this example, feature F matches feature C as their set of attributes collides in the hash table.

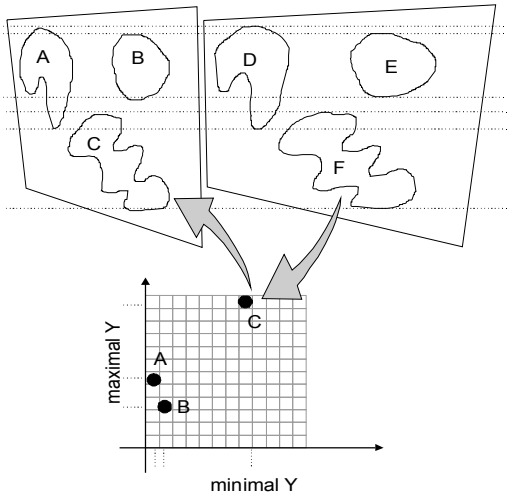


Figure 1. Stereo matching of regional features.

The intersection between a horizontal line and the contours of the convex hulls form two pairs of homologue points. The whole contour can then be reconstructed using a sequence of horizontal lines.

Base and invariants

A base is an ordered tuple of model-features associated to their corresponding image-features whose size is minimal, but enough to determine the geometric transform that maps model-features to image-features. A measure over feature tuples is considered invariant to a particular type of geometric transformation if it remains unchanged independently of the parameters of the transformation applied to the features. Aiming lower complexity, a compromise between search in correspondence- and pose-spaces is established: the search for correspondences can be restricted to tuples of image-features as large as the base.

From each 3D-reconstructed polygon, the barycentre coordinates and plane normal vector direction are extracted. A plane can be adjusted through the reconstructed vertices of a feature convex hull contour and its orientation can be estimated. The barycentre coordinates can then be computed by projecting the reconstructed points onto the adjusted plane and integrating along the contour according to Green's theorem.

It is necessary 2 features to form a base because the information provided by these attributes (barycentre and normal direction) allows a remaining degree of freedom for each feature. Defining a coordinate system with the origin in the barycentre of the first feature and the *x* axis oriented as its normal and the *y* axis oriented such that the barycentre of the second feature lie on the plane *xy,* The resulting *x* and *y* coordinates of the barycentre of the second feature and its normal direction defined in this new reference frame constitute a 4-value vector invariant to the rigid transformation.

Recognition model and algorithm

Three types of attributes are obtained from the observed geometry: (1) invariant attributes of individual features (colour, area); (2) attributes of individual features that depend on the transformation (barycentre and normal direction); (3) invariant attributes of feature-pairs (distance between barycentres, angle between planes etc.).

We develop a recognition algorithm using type (3) attributes considering the model illustrated in figure 2, where I_i and M_m are image- and model-features respectively and the Boolean C_{im} represents the correspondence between them.

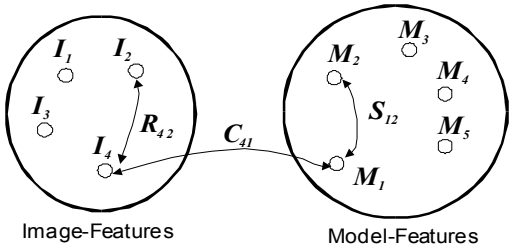


Figure 2. Recognition model

The proposed algorithm is based on abductive reasoning (diagnostics reasoning). If the measured attribute vector R_{ij} of the pair of image-features (I_i, I_j) is close enough to the attribute vector S_{mn} of model-features (M_m, M_n), we can raise the hypothesis C_{im}, i.e., I_i matches M_m. As more pairs of image-features that include I_i are analysed, more clues about I_i are collected, allowing reconsideration of the existing hypotheses. A voting scheme is then used to select the most reliable hypothesis to recognize the model-feature corresponding to I_i. An equivalent probabilistic approach is the estimation of the correspondences by maximum likelihood.

The equality $R_{ij} = S_{mn}$ is verified using a hash table whose granularity models the imprecision. The less dense are hash entries, the less strict or precise is the verification of the equality. Attributes are considered equal if they index the same cell of the hash table.

The hash table is built from the previously known geometric model. For each pair of model-features (M_m, M_n), with an attribute vector S_{mn}, the table index is computed based on the value of the attribute and a reference to the pair (*m*, *n*) is appended to the indexed cell. A description of the algorithms follows.

Building the hash table, off-line

For each model-feature *m*
  For each other model-feature *n*
    Measure the attributes *S(m,n);*
    Compute the index to the hash table given *S(m,n);*
    Append to the indexed cell in the table a reference to *(m,n).*

Recognition of image-features, on-line

For each image-feature *i*
  For each other image-feature *j*
    Measure the attributes *R(i,j);*
    Compute the index to the hash table given *R(i,j);*
    For each item in the indexed cell
      Find the identity of the pair *(m,n)* referenced in the item.
      Verify the compatibility of the remaining attributes of *(i,j)* and *(m,n);*
      If successful, cast a vote for *C(i,m)* and *C(j,n);*
For each image-feature *i*
  Find *X*, model-feature with most votes
  so that $C(i,X) \geq C(i,m), \forall m$;
  State that *i* matches *X*.

4. IMPLEMENTATION

The first steps of image-feature extraction are segmentation, labelling and tracing of region convex hull contours. A set of convex polygons representing image-features is returned. A simple segmentation strategy based on colour normalization and Euclidean distance in RGB-space is employed. Labelling is implemented as described in [Gonzalez and Woods, 1992]. The contours of features are traced placing a cursor in any point of the contour and examining each 2x2-pixel window to decide the direction the cursor must be moved to follow the contour. The collected contour coordinates are sorted by *y*-coordinate and the vertices not belonging to the convex hull are removed by a linear-time algorithm.

Pose is estimated by building and solving a system of linear equations for parameters from the information provided by the image-model feature matching. A least-squares technique is employed due to the availability of a more correspondences than the minimum required. The orthonormal property of the rotation matrix is fixed afterwards by factorisation.

We illustrate the proposed approach with an implemented case. A virtual model of the scene was created and a physical scene was constructed based

on this model. For a JPEG 1280x960-pixel image, the best 7 features ranked by maximal number of votes were correctly recognized and the resulting alignment is depicted in figure 3. We detected that 6 identified features were enough for a good alignment result in this experiment. Additional features contributed little to a better result because we lack a model for the dispersion of pose error and a means to evaluate the quality of individual pose measures.

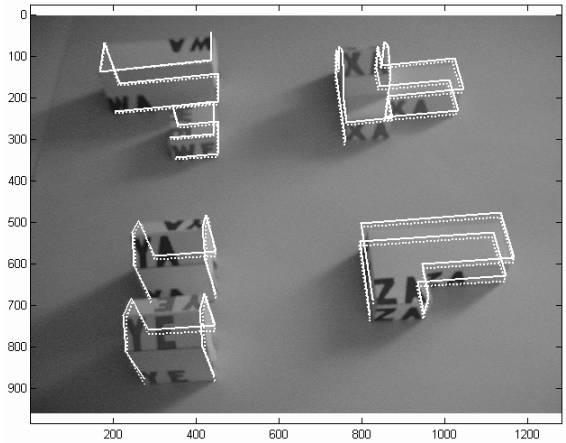


Figure 3 Resulting alignment. The continuous lines represent the computed alignment, while the dashed lines represent the nominal alignment.

5. Complexity Analysis

Let M be the number of model-features, I, the number of image-features and $O(H)$, the average complexity of a single access to the hash-table. Vote collection considers an average of $O(H)$ votes obtained from the hash-table for each of the $O(I^2)$ pairs of image-features, resulting a cost of $O(I^2H)$. Vote table analysis consists of a search in each row of the vote table for the column with greatest number of votes for which the cost is $O(IM)$. Total cost of the on-line phase is then $O(I^2H + IM)$. If the number of features is kept small, as in the expected usual case where each feature is represented by a considerable area of the image, the cost of sweeping the whole image for segmentation and labelling turns out to be the bottleneck.

6. CONCLUSION

We address in this paper the problem of automatic visual alignment from images. Our approach is comprehensive integrating several Computer Vision techniques around the assumptions of planar regional features and stereo vision. We show that under these assumptions, most of the involved components of the vision process become simple. The method is appropriate for images with regional features and designed to work in conditions of near perspective and partial occlusion. Being not incremental, it is designed to work independently of additional sensors and can be used to estimate initial state pose without user intervention. Regional features are typically found in cityscapes, traffic, office and home scenes.

7. ACKNOWLEDGMENTS

We thank Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) for financial support.

8. REFERENCES

Azuma, R. T., 1999. The Challenge of Making Augmented Reality Work Outdoors. In *Mixed Reality: Merging Real and Virtual Worlds*, Y. Ohta and H. Tamura, eds. Springer-Verlag, pp 379-390.

Bajura, M. and Neumann, U., 1995. Dynamic Registration Correction in Augmented Reality Systems, *IEEE VRAIS proc.*, pp 189-196.

Cornelis, M. V. K., Pollefeys, M. and Gool, L. V. Augmented reality from uncalibrated video sequences. *3D Structure from Images SMILE 2000.*

Forsyth, D.A., Mundy, J.L., Zisserman, A., Coelho, C., Heller, A.J. and Rothwell, C.A., 1991, Invariant descriptors for 3D object recognition and pose, *PAMI(13)*, no 10, pp 971-991.

Fox, D., Burgard, W., Dellaert, F. and Thrun, S., 1999. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. *Proc. of the 16th National Conference on Artificial Intelligence*.

Gonzalez, R.C. and Woods, R.E., 1992, *Digital Image Processing*, Addison-Wesley, Reading.

Haustler, G. and Ritter, D., 1999. Feature-Based Object Recognition and Localization in 3D-Space using a Single Video Image. *CVIU* 73(1) pp 64-81.

Koller, D. 1993. Moving Object Recognition and Classification based on Recursive Shape Parameter Estimation. In *Proc. of the 12th Israeli Conf. on Artificial Intelligence, Computer Vision, and Neural Networks*, pp 359-368, Tel-Aviv, Israel.

Olson, C. F., 1994. Time and Space Efficient Pose Clustering, IEEE *Conference on Computer Vision and Pattern Recognition*, pp 251-258.

Se, Lowe, Little 2002. Global Localization using Distinctive Visual Features. *Proc. of the Intl. Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, pp 226--231.

Simon, G. and Berger, M. O. 1997, *A two-stage robust statistical method for temporal registration from features of various type*. INRIA TR 3235.

Wolfson, H.J. and Rigoutsos, L., 1997, Geometric hashing: an overview, *CalSE(4)*, no 4, pp 10-21.

A Classification Scheme for Lens Techniques

Henning Griethe
University of Rostock
Chair of Business Informatics
18051, Rostock, Germany
Henning.Griethe@
wisofak.uni-rostock.de

Georg Fuchs
University of Rostock
Chair of Computer Graphics
18059, Rostock, Germany
Georg.Fuchs@
informatik.uni-rostock.de

Heidrun Schumann
University of Rostock
Chair of Computer Graphics
18059, Rostock, Germany
Heidrun.Schumann@
informatik.uni-rostock.de

Abstract

For the visualization of abstract information with spatial dependencies, the combination of icon representations with maps is widely accepted. However, with an increasing amount of data creating complete, yet not overloaded, visualizations becomes evermore difficult. Effective interaction methods are therefore needed to discover hidden information in these pictures. Lens techniques offer the potential to efficiently combine proven methods from cartography and information visualization. Such techniques are not yet exploited sufficiently, exemplified by how little effort has been made so far in even systematizing lenses used in different fields.

This paper introduces a common classification scheme that integrates known techniques and provides points of departure for new approaches. Derived from this work, a novel lens technique for the explorative analysis of maps is presented. The effectiveness of the proposed lens is demonstrated in an existing system for the visualization of health data.

Keywords

Visualization, Multivariate Data, Spatial-dependent Data, Cartography, Iconic Techniques, Lens Techniques.

1. Introduction

Data sources such as surveys or simulations generate vast amounts of raw data. Information visualization has turned out a wealth of methods for the efficient and comprehensive visualization of complex information spaces as depicted e.g. in [Tuf90]. However, most approaches give little or no regard to any (geo-)spatial dependencies. Applications such as multispectral satellite images and healthcare statistics are examples for geo-referenced, multivariate information.

Cartography is about effectively presenting large geographic data sets with different granularity on thematic maps. Tried and proven concepts for the creation of high-quality maps have lately being augmented with automated methods to create on-screen maps. However, visualizing data on maps is mostly still confined to low-dimensional, unstructured information spaces.

SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

Of course, ways to combine the expertise from both fields has been studied before (e.g. [Sku00, KrOm96]). Despite the sophistication of cartographic and information visualization techniques, it is often impossible to represent all relevant information in one image. The user must be able to interactively browse and explore the representation in order to gain satisfactory insight from the data.

One way to meet this requirement is to seamlessly integrating multiple levels of detail within the same representation, i.e. focus & context concepts with interactive repositioning of focus areas, as examined e.g. by [Kea98].

Viewing focus regions in such focus & context representations as lenses takes this concept one step further. Lenses can be stacked on top of each other to create combined effects (cf. [SFB94]). Thus, lenses are locally confined, chainable operators applied to the visualization pipeline.

Despite their usefulness, only little effort has been made to classify lenses in more detail. By systematizing the major principles of lens functionality, one gains the ability to adapt and combine already existing methods in new and beneficial ways. Therefore this paper will introduce

a common classification scheme that was the basis for the development of novel lens techniques.

The paper is structured as follows: In section 2, methods from cartography and information visualization that can be used for lens techniques will be presented shortly. Section 3 abstracts the characteristics of lenses and pictures them in a uniform classification scheme. A novel technique will be introduced in Section 4. Section 5 concludes with some results.

2. Basic concepts for lens design

Lens techniques typically realize some kind of optical distortion in a confined region (defined by its position and shape) of an image which allows the user to take a closer look at interesting details. As shown e.g. in [SFB94] there is a wide range of extending functionalities available. With geospatial data in mind, we will have a look on established concepts from Information Visualization and cartography to enhance lens techniques.

Focus & context applies specific transformation and magnification functions (see [LeAp94]) to control the magnification in the focus region as well as the miniaturization in the context region. This concept can be adapted as a lens technique so that the functions do not influence the representation outside of the lens region.[1]

The **overview & detail** concept can be used for lens techniques as well, e.g. by placing a detailed, semi-transparent layer above the overview layer. This differs from focus & context in that no distortions have to be applied.

Information hiding lenses work by filtering out superfluous data and thus emphasizing important information in a confined region. Such lenses can use different layers, selecting only a subset of layers to be shown (e.g. the 'Moving Filter' [SFB94]).

Even time-consuming cartographic methods can be used for lens techniques when restricting their application to a confined lens region only.

A **visual hierarchy** (see e.g. [HaGr94]) arises from the variation of visual variables like color, saturation, or sharpness (see [Ber74], [KrOm96]). For example, crisp picture elements are visually more prominent than blurred ones (cf. [KMH01]). Porting this general approach to lenses means that a given visual hierarchy is altered in the lens region, e.g. by moving objects matching certain criteria to the top of the (imposed or given) hierarchy.

[1] Note that focus & context, and thus lenses, are not confined to magnification effects. Keahey uses the notion of a 'detail axis' [Kea98], with the meaning of detail being domain-specific.

Generalization is an important concept in cartography to adapt a highly detailed base map to smaller scales. A 'generalization lens' addresses the presentation problem locally by graphic or semantic (concerning the data) simplification, classification, scaling, displacement, aggregation and subset selection of map features (cf. e.g. [HaGr94]).

Cartographic **labeling** uses sophisticated methods to shape objects and optimize their position on a map. However, good quality labeling for a large number of features is very hard to achieve in interactive times. Confining labeling to only a lens region reduces both the number of features as well as the general complexity of the task.

3. Development of a classification scheme for lens techniques on maps

In the following, the aspects contributing to the lens classification scheme are presented.

3.1. Fundamental type of lens

The first point of distinction for lens techniques is how they fundamentally function: either in a purely graphical way, or by operating on a semantic level. Spatial transformations (magnification, distortion), the variation of visual variables and the relocation of map objects are graphic techniques, while an increase or a reduction of information content as in information hiding or in generalization methods are techniques with semantic effects.

3.2. Integration in the visualization pipeline

The second criterion to classify a lens is the stage of the visualization pipeline modified by that lens. The principal possibilities are the realization as an additional filter operation, a modification of the mapping process, or the use of a different renderer for the lens region. The latter variant could be directly integrated into the visualization process, or be employed on the finished raster image (making that lens an independent, detached tool).

Obviously, which pipeline stage is modified is not completely independent from the first criterion, as any semantic change can only be achieved during the filtering and/or mapping stages of the pipeline.

3.3. Lens parameters

The third aspect of classification are the principal lens parameters: its position on the map, its shape and means for its adjustment (influences).

The most common way to position a lens is probably by user interaction (pointing with the mouse), but also data- or context-driven means are possible. Examples are moving the lens in reaction to changes to the data set (a new entry has been added or altered), or the automatic positioning of the lens in response to some user query, e.g. for a maximum

value. Likewise, the lens' shape is a parameter that can be changed as required. It can be a simple, fixed geometric form, e.g. a circle. Or it can be a more complex shape that is adjusted according to the current situation. Prime example would be a lens that always encloses complete map areas, and thus would have its shape defined by whichever map area the lens is positioned on.

Furthermore certain events, system requirements, or time may be used to generally parameterize the lens function aside from position and shape and hence should also be taken into account for classification.

3.4. Presentation with lens techniques

Leung & Apperley [LeAp94] distinguish distorted and undistorted views. Following their concept we also want to classify lenses by their distortion effects. For graphical lenses this means whether or not a straightforward optical distortion is applied within the lens area. With semantic lenses, one could distinguish if presented information will be gathered by simple selection (undistorted) or computed by appropriate functions (distorted).

Finally, the actual means by which the lens operates on the data is the last, if most diverse, classification aspect. It can be any applicable information visualization or cartographic methods, e.g. simple graphical magnification, creating visual hierarchies, or semantic generalizations (cf. Section 2). These criteria are universal enough to be useful for lens techniques in general and are therefore integrated into the classification scheme.

There are of course other distinctions possible, e.g. according to the kind of relevant events, the kind of used algorithms or data structures. Such criteria are quite special and go beyond a general systematic so that they were not regarded. Table 1 summarizes the mentioned criteria.

To demonstrate the classification of lens techniques by our scheme we will categorize an established approach: Rase's 'cartographic lens' [Ras97]. It is a *graphic* lens operating at the *rendering stage* of the visualization pipeline by repositioning points in the geometric space. The shape is a circle of *fixed* size, defined by a focal point and a given radius. The focal point can be moved around a map (*user-controlled*). Parameterizing influence comes from *user interaction*. By nature of the employed fisheye projection, the lens realizes a *distorting* (non-uniform) graphical *magnification* effect across its area.

To summarize, our categorization scheme allows the classification, evaluation and comparison of lens techniques. Moreover, on the basis of this scheme special lens types regarding the requirements of a given application can be developed.

Criterion	**Possibilities**				
Fundamental type	Semantic	Graphic			
Integration into the visualization pipeline	Filtering stage	Mapping stage	Rendering stage	None (image post-processing)	
Lens parameters					
Shape	Fixed	User-controlled	Data/context-driven		
Position	User-controlled	Data/context-driven			
Influences	User interaction	Events	System requirements	Time	Other
View type	Distorted	Undistorted			
Method/Operator	Various (visual hierarchies, magnification, ….)				

Table 1: Categorization scheme for lens techniques (adapted from [Gri04])

4. A new lens technique for LandVis$_t$

LandVis$_t$ is a system for the visualization of health information. For a German federal state health insurance data (the number of people who reported sick in a period) are displayed. LandVis$_t$ uses icons on an administrative map which is organized in three resolution levels: state, districts and zip-code areas. One of these icons, the maximum icon, encodes in a 'clock hand' at which time most reports occurred. Placed on a map these icons give an impression on the spread of a disease (see [Tom02]).

Unfortunately, at higher map resolution levels the representation becomes cluttered as a growing num-ber of icons has to be placed. The integrated Carto-graphic Lens [Ras97] can declutter the representa-tion locally, however only within limits. Thus, more sophisticated lens techniques are needed. As an example we will now introduce a selected technique from [Gri04] that operates on the filtering stage of the visualization pipeline: the Aggregate-Lens.

Influencing the visualization at the filtering stage allows the most complex adaptations. The general

idea of the Aggregate-Lens is to aggregate any given data within the lens region into a compact overview representation. Examples are the calculation of averages and sums or the determination of extreme values, outliers and other characteristic properties. Such a lens dissolves local clutter and allows the user to gain a flexible and quick impression on the main characteristics of a region.

As an example specific to LandVis$_t$, the Aggregate-Lens was implemented in a way that it summarizes the maximum icons within its boundaries. It displays a single modified maximum icon that shows the average peak time over all map areas at which a certain disease reached its maximum spread in the lens region. These map areas are selected upon inclusion of the bounding box centers in a movable selection square and are highlighted accordingly (see Figure 1 left).

As supplemental information to the average peak time, the Aggregate-Lens displays the overall time interval of all peak times as a circle segment around the clock face. Furthermore the standard deviation is encoded in the color of that circle segment using a blue-red color scale (low to high deviation). Last, the most extreme outlier is shown as a yellow clock hand in the according map area (Figure 1 left).

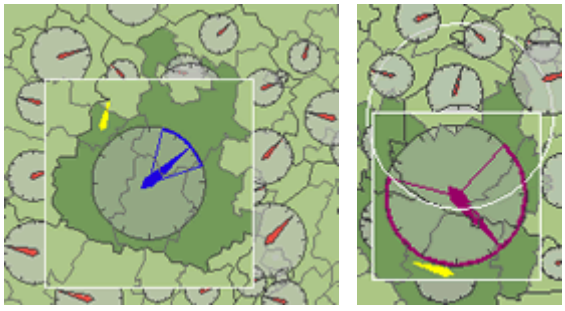


Figure 1: Aggregate-Lens (left); example for its combination with a Cartographic Lens (right)

Classification: The Aggregate-Lens is a *semantic* lens operating on the *filtering stage*. Its position is solely *user-controlled* (with a mouse). While the user can specify the size of the selection rectangle, the form of the lens is determined by geographic conditions, i.e. it is *data-driven*. Parameterizing influence originates from *user interaction*. The presented information is modified by a function, so the Aggregate-Lens is a semantic *distorting* lens. The content of the lens is displayed by means of *modified icon encoding*.

5. Conclusions

An effective visualization requires sophisticated interaction methods. Within the considered context of geo-spatial multivariate data, lens techniques are a suitable choice.

Despite its application capabilities these techniques are rarely systematized in literature. This paper therefore proposed a common classification scheme for arbitrary lens techniques, based on a review of applicable techniques from both information visualization and cartography.

As an example, the Aggregate-Lens was proposed that declutters the representation by displaying statistical moments of a number of data sets. An implementation example specifically for health data using icon encoding was presented.

Another aspect of lens techniques is its possible combination, as shown in Figure 1 (right). Using the proposed classification, expedient combinations can be found, as examined in more detail in [Gri04].

6. References

[Ber74] Jacques Bertin. *Graphische Semiologie: Diagramme, Netze, Karten*. de Gruyter, 1974.

[Gri04] H. Griethe. *Einsatz von Linsentechniken für Ikonen über interaktiven Kartendarstellungen*. Master Thesis, Chair of Computer Graphics, University of Rostock, 2004.

[HaGr94] G. Hake and D. Grünreich. *Kartographie*. de Gruyter Berlin, 7. Auflage, 1994.

[Kea98] T. A. Keahey. *The Generalized Detail-in-Context Problem*. In Proceedings of the IEEE Symposium on Information Visualization, IEEE Visualization, 1998.

[KMH01] R. Kosara, S. Miksch and H. Hauser. *Semantic Depth of Field*. In Proceedings of InfoVis 2001, San Diego, USA, October 2001.

[KrOm96] M. J. Kraak, F. J. Omeling. *Cartography, Visualization of spatial data*. 1996.

[LeAp94] Y. K. Leung and M. D. Apperley. *A Review and Taxonomy of Distortion-Oriented Presentation Techniques*. ACM Transactions on Human-Computer Interaction, Vol. 1:2, 1994.

[Ras97] Wolf-Dieter Rase. *Fischauge-Projektionen als kartographische Lupen*. Salzburger Geographische Materialien, Heft 26, 1997.

[SFB94] M. C. Stone, K. Fishkin and E. A. Bier. *The movable filter as a user interface tool*. In Proceedings of the ACM Conference on Computer Human Interaction, 1994.

[Sku00] André Skupin. *From Metaphor to Method: Cartographic Perspectives on Information Visualization*. Department of Geography, University of New Orleans, 2000.

[Tom03] C. Tominski et al. *Visualisierung zeitlicher Verläufe über geographischen Karten*. In Kartographische Schriften (Band 7), Deutsche Gesellschaft für Kartographie e.V., 2003.

[Tuf90] Edward R. Tufte. *Envisioning Information*. Graphics Press, Connecticut, 1990.

A Roller - Fast Sampling-Based Texture Synthesis Algorithm

Michal Haindl
Institute of Information Theory and Automation
Academy of Sciences of the Czech Republic
CZ18208, Prague, Czech Republic
haindl@utia.cas.cz

Martin Hatka
Faculty of Nuclear Sciences
and Physical Engineering
Czech Technical University
CZ11519 Prague, Czech Republic
hadis@email.cz

ABSTRACT

This paper describes a method for synthesizing natural textures that realistically matches given colour texture appearance. The novel texture synthesis method, which we call the roller, is based on the overlapping tiling and subsequent minimum error boundary cut. An optimal double toroidal patch is seamlessly repeated during the synthesis step. While the method allows only moderate texture compression it is extremely fast due to separation of the analytical step of the algorithm from the texture synthesis part, universal, and easily implementable in a graphical hardware for purpose of real-time colour texture rendering.

Keywords
Virtual Reality Models, Texture Synthesis.

1. INTRODUCTION

Virtual or augmented reality systems (VR) require object surfaces covered with realistic nature-like colour textures to enhance realism in virtual scenes. To make virtual worlds realistic detailed scene models must be built. Satisfactory models require not only complex 3D shapes accorded with the captured scene, but also life-like colour and texture. This will increase significantly the realism of the synthetic scene generated. Textures provide useful cues to a subject navigating in such a VR environment, and they also aid in the accurate detailed reconstruction of the environment.

WSCG SHORT papers proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

The purpose of a synthetic texture is to reproduce a given digitized texture image so that both natural and synthetic texture will be indiscernible. However modelling of a natural texture is a very challenging and difficult task, due to unlimited variety of possible surfaces, illumination and viewing conditions simultaneously with the strong discriminative functionality of the human visual system. The related texture modelling approaches may be divided primarily into intelligent sampling and model-based-analysis and synthesis, but no ideal method for texture synthesis exists. Each of existing approaches has its advantages and also limitations.

Model-based colour texture synthesis [Bes74], [Kas81],[Ben98], [Hai91a], [Hai00], [Zhu00],[Gri03], [Hai04] requires non-standard multi-dimensional (3D for static colour textures or even 7D for bidirectional texture function) models. If a 3D texture space can be factorized then these data can be modelled using a set of less-dimensional 2D random field models, otherwise it is necessary to use some 3D random field model. Among such possible models the Gaussian Markov random fields are advantageous not only because they do not suffer with some problems of alternative options (see [Hai91b], [Hai91a],

[Hai00], [Hai00] for details) but they are also relatively easy to synthesize and still flexible enough to imitate a large set of natural and artificial textures. Unfortunately real data space can be decorrelated only approximately, hence the independent spectral component modelling approach suffers with some loss of image information. Alternative full 3D models allow unrestricted spatial-spectral correlation modelling, but its main drawback is large amount of parameters to be estimated and in the case of Markov models also the necessity to estimate all these parameters simultaneously. Model-based methods are mostly too difficult to be implemented in modern graphical card processors.

Intelligent sampling approaches [DeB97], [Efr99], [Efr01], [Hee95], [Xu00] rely on sophisticated sampling from real texture measurements. Given a randomly selected starting block of texture in the image, they propagate out from it selecting new texture blocks. For each new block in the image, all neighboring blocks that have already been generated are checked and the example image (or images) is searched for similar textures. The k best such matches are found and then randomly choosen the corresponding new texture patch from among them. The methods [Efr01],[Efr99], [Wei01] all vary in how the blocks are represented, how similarity is determined, and how the search is performed. Intelligent sampling approaches are based on some sort of original small texture sampling and the best of them produce very realistic synthetic textures, usually better than the model-based methods. However these methods require to store original texture sample, often produce visible seams, they are mostly computationally demanding, they cannot generate textures unseen by the algorithm, and they cannot even approach the large compression ratio of the model-based methods.

The rest of the paper is organised as follows. The following section describes a simple sampling approach based on the repetition of a double toroidal tile carved from the original texture measurement. Results and conclusions are reported in the last section.

2. DOUBLE TOROIDAL TILE

The double toroidal tile (see Fig.1) is limited by the selected minimal rectangle to be inscribed in from the original texture measurement. The texture tile is assumed to be indexed on the regular two-dimensional toroidal lattice. The optimal lattice searched by the algorithm allows for seamless repetition in both horizontal and vertical directions, respectively.

Let us define the overlap error for a pixel r as follows:

$$\psi_r^h \quad = \quad \left(Y_r - Y_{r+[N-h,0]}\right)^2 \quad \forall r \in I_h \quad ,$$

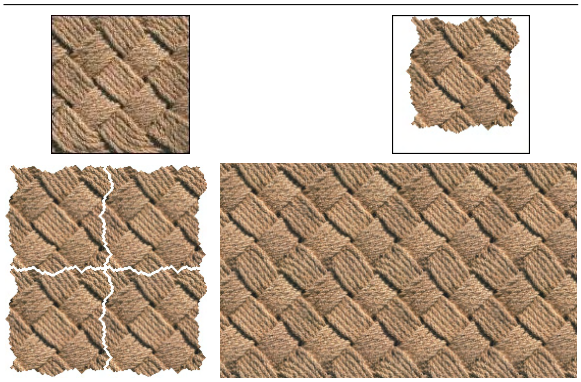


Figure 1. The roller principle - upper row input texture and toroidal tile, bottom row texture generation and the result, respectively.

$$\psi_r^v \quad = \quad \left(Y_r - Y_{r+[0,M-v]}\right)^2 \quad \forall r \in I_v \quad ,$$

where Y_r denotes a multispectral pixel indexed on the $N \times M$ underlying lattice. The multiindex r has two components $r = [r_1, r_2]$, the first component is row and the second one column index, respectively. The index sets I_h, I_v are defined

$$\begin{aligned} I_h &= (1, \ldots, h) \times (1, \ldots, M) \quad , \\ I_v &= (1, \ldots, N) \times (1, \ldots, v) \quad , \end{aligned}$$

h and v lie in preselected intervals $h \in \langle h_{\min}; h_{\max} \rangle$ and $v \in \langle v_{\min}; v_{\max} \rangle$. The optimal horizontal and vertical overlaps are found from the following two relations for $\zeta \in \{h, v\}$:

$$\zeta^* = \min_{\zeta} \left\{ \frac{1}{\zeta} \sum_{\forall r \in I_\zeta} \psi_r^\zeta \right\} \quad .$$

Optimal Cut

The optimal cuts for both the horizontal and vertical edge is searched using the dynamic programming method. Alternatively we can use some other suboptimal search such as the A^* algorithm if necessary to speed up also the analytical part of the method. However for most applications the fast synthesis is prerequisite while the computation time for separately and only once solved analytical part is of no importance. Both optimal cuts have to minimize the overall path error

$$\begin{aligned} \Psi_r^{h^*} &= \psi_r^{h^*} + \min \left\{ \Psi_{r-[1,1]}^{h^*}, \Psi_{r-[0,1]}^{h^*}, \Psi_{r+[1,-1]}^{h^*} \right\} \quad , \\ \Psi_r^{v^*} &= \psi_r^{v^*} + \min \left\{ \Psi_{r-[1,1]}^{v^*}, \Psi_{r-[1,0]}^{v^*}, \Psi_{r+[-1,1]}^{v^*} \right\} \quad . \end{aligned}$$

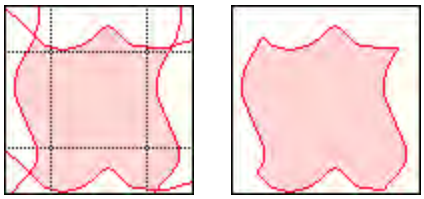


Figure 2. The optimal tile cuts in both directions.

The combination of both optimal vertical and horizontal cuts creates the toroidal tile as is demonstrated on the Fig.2.

Synthesis

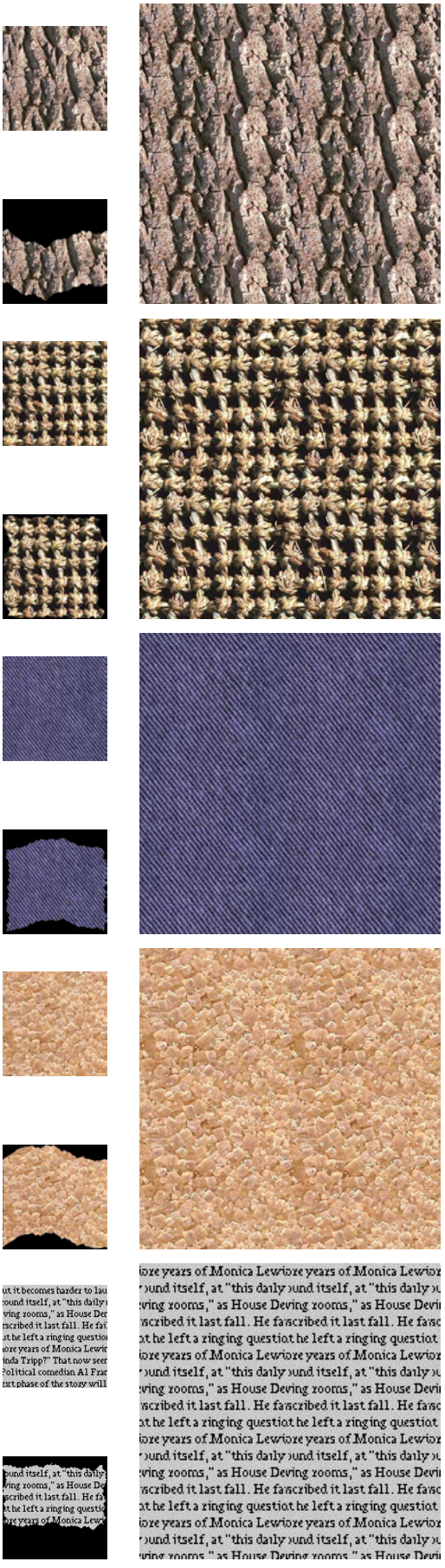
The synthesis of any required colour texture size is simple repetition of the created double toroidal tile in both directions until the required texture is generated. There is no computation involved in this step hence it can be easily implemented in real time or inside the graphical card processing unit.
The complete roller algorithm is as follows:

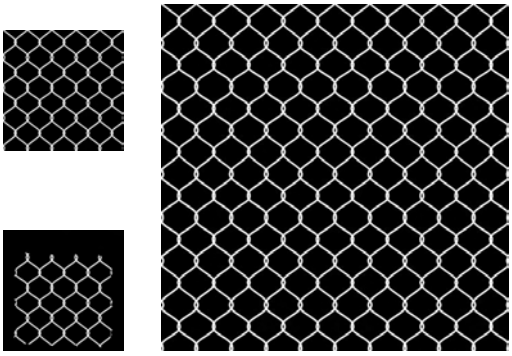
- Analysis
    1. Find the optimal horizontal and vertical overlaps h^*, v^*.
    2. Search for optimal horizontal and vertical cuts starting from $I_h \cap I_v$.
    3. Create the double toroidal texture tile.
- Synthesis

The range of horizontal and vertical overlapping intervals are the only parameters specified by the user. The analytical part is completely separated from the synthesis. The most time consuming part of the analysis is optimal cuts search whose time requirement is proportional to $T \propto v^2hN - 2v^2h^2 + Mvh^2$. The optimal overlap search time is negligible and the synthesis step contains no computations.

3. RESULTS AND CONCLUSIONS

We have tested the algorithm on several hundred colour and grayscale textures from the VisTex database [Pic95], Noctua database, Brodatz textures [Bro66] and mainly from our extensive Prague texture database, which currently contains over 500 colour textures. Tested textures were either natural such as bark, wood, plants, water, etc., or man-made knitwear, upholstery, brick wall, textiles, food products and many others. Some results (bark, rattan, jeans cloth, sugar, text and wire mesh) are demonstrated in the following images.





Resulting textures are mostly surprisingly good for such a very simple algorithm. For example our results on the text texture (the second from the bottom) are indistinguishable (see [Efr01]) from results on the same texture using much more complicated and slower image quilting algorithm [Efr01]. Obviously there is no optimal texture modelling method and also the presented method fails on some textures. However on most of our failure examples also some alternative intelligent sampling methods failed. The test results of our algorithm on our extensive natural texture collection are encouraging. The presented method is extremely fast, very simple and easily implementable even in the graphical processing unit. The method offers moderate compression ratio for transmission or storing texture information while it has negligible computation complexity. The roller method can be used for easy and fast seamless synthesis of any required texture size for many natural or man made textures. The method's extension for bidirectional texture functions is straightforward.

ACKNOWLEDGMENTS

This research was supported by the EC projects no. IST-2001-34744 RealReflect, FP6-507752 MUSCLE and grants No.A2075302, 1ET400750407 of the Grant Agency of the Academy of Sciences CR.

References

[Ben98] BENNETT, J., AND KHOTANZAD, A. 1998. Multispectral random field models for synthesis and analysis of color images. *IEEE Trans. on Pattern Analysis and Machine Intelligence 20*, 3 (March), 327–332.

[Bes74] BESAG, J. 1974. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, B-36*, 2 (February), 192–236.

[Bro66] BRODATZ, P. 1966. *Textures: A Photographic Album for Artists and Designers*. Dover Publications.

[DeB97] DE BONET, J. 1997. Multiresolution sampling procedure for analysis and synthesis of textured images. In *Proc. SIGGRAPH 97*, ACM, 361–368.

[Efr01] EFROS, A. A., AND FREEMAN, W. T. 2001. Image quilting for texture synthesis and transfer. In *SIGGRAPH 2001, Computer Graphics Proceedings*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., 341–346.

[Efr99] EFROS, A. A., AND LEUNG, T. K. 1999. Texture synthesis by non-parametric sampling. In *Proc. Int. Conf. on Computer Vision (2)*, 1033–1038.

[Gri03] GRIM, J., AND HAINDL, M. 2003. Texture modelling by discrete distribution mixtures. *Computational Statistics Data Analysis 41*, 3-4 (Jan.), 603–615.

[Hai00] HAINDL, M., AND HAVLÍČEK, V. 2000. A multiresolution causal colour texture model. In *Advances in Pattern Recognition, Lecture Notes in Computer Science 1876*, F. J. Ferri, J. M. Inesta, and P. Pudil, Eds. Springer-Verlag, Berlin, August, ch. 1, 114 –122.

[Hai02b] HAINDL, M., AND HAVLÍČEK, V. 2002. A multiscale colour texture model. In *Proceedings of the 16th IAPR International Conference on Pattern Recognition*, IEEE Press, Quebec City, R. Kasturi, D. Laurendeau, and C. Suen, Eds., vol. I, 255 – 258.

[Hai04] HAINDL, M., GRIM, J., SOMOL, P., PUDIL, P., AND KUDO, M. 2004. A gaussian mixture-based colour texture model. In *Proc. 17th IAPR Int. Conf. on Pattern Recognition*, IEEE Press, Los Alamitos, J. Kittler, M. Petrou, and M. Nixon, Eds., vol. 3, 177–180.

[Hai91a] HAINDL, M. 1991. Texture synthesis. *CWI Quarterly 4*, 4 (December), 305–331.

[Hai91b] HAINDL, M. 1991. Texture synthesis. Tech. Rep. CS-R9139, Centrum voor Wiskunde en Informatica, Amsterdam, The Netherlands.

[Hai00] HAINDL, M. 2000. Texture modelling. In *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, International Institute of Informatics and Systemics, Orlando, USA, B. Sanchez, J. M. Pineda, J. Wolfmann, Z. Bellahse, and F. Ferri, Eds., vol. VII, 634–639.

[Hai03] HAINDL, M. FILIP, J. 2003. Fast btf texture modelling. In *Texture 2003*, IEEE Computer Society.

[Hee95] HEEGER, D., AND BERGEN, J. 1995. Pyramid based texture analysis/synthesis. In *Proc. SIGGRAPH 95*, ACM, 229–238.

[Kas81] KASHYAP, R. 1981. Analysis and synthesis of image patterns by spatial interaction models. In *Progress in Pattern Recognition 1*, Elsevier, North-Holland, L. Kanal and A.Rosenfeld, Eds.

[Pic95] PICKARD, R., GRASZYK, C., MANN, S., WACHMAN, J., PICKARD, L., AND CAMPBELL, L. 1995. Vistex database. Tech.rep., MIT Media Laboratory.

[Wei01] WEI, L., AND LEVOY, M. 2001. Texture synthesis over arbitrary manifold surfaces. In *SIGGRAPH 20001, Computer Graphics Proceedings*, ACM Press / ACM SIGGRAPH / Addison Wesley Longman.

[Xu00] XU, Y., GUO, B., AND SHUM, H. 2000. Chaos mosaic: Fast and memory efficient texture synthesis. Tech. Rep. MSR-TR-2000-32, Redmont.

[Zhu00] ZHU, S., LIU, X., AND WU, Y. 2000. Exploring texture ensembles by efficient markov chain monte carlo - toward a "trichromacy" theory of texture. *IEEE Trans. on Pattern Analysis and Machine Intelligence 22*, 6 (June), 554–569.

A New Sub-Pixel Detector for X-Corners in Camera Calibration Targets

Dazhi Chen Guangjun Zhang

School of Instruments Science and Photoelectric Engineering
Beijing University of Aeronautics and Astronautics
Xueyuan Road 37
100083, Beijing, China

cdzbuaa@hotmail.com zgj310@eyou.com

ABSTRACT

X-corner patterns are most widely used in camera calibration. In this paper, we propose a new sub-pixel detector for X-corners, which is much simpler than the traditional sub-pixel detection algorithm. In this new algorithm, the pixel position of X-corner is firstly detected by a new operator. Then a second order Taylor polynomial describing the local intensity profile around the corner is educed. The sub-pixel position of X-corner can be determined directly by calculating the saddle point of this intensity profile. Neither preliminary intensity interpolation nor quadratic fitting of the intensity surface is necessary, which greatly reduces the computation load of the detection process. Furthermore, computer simulation results indicate that our new algorithm is slightly more accurate than the traditional algorithm.

Keywords

X-corners, sub-pixel detection, camera calibration

1. INTRODUCTION

Detection of target points is an important task in camera calibration, which will greatly affect the accuracy and stability of 3D machine vision. In recent years, many patterns have been used in camera calibration, such as squares [Zha00a], circles [Hei00a] and crosses [Dev00a] etc. Checkerboards with black-and-white squares are most widely used because the easy sub-pixel detection algorithm for X-corners with high precision [Luc00a]. Traditional algorithm for detecting X-corners first estimates their pixel locations by standard corner detectors (such as Harris [Har01a] and Noble [Nob00a]), then the sub-pixel positions can be determined by fitting quadratic functions to the local intensity profile around the corners and computing their extremal points. The main shortcoming of this algorithm is that the fitting of local intensity surface complicate the detection process. In 2001, Lucchese proposed a new simplified algorithm finding the extremal points by a morphological shrinking operation on the local intensity profile [Luc00a]. This algorithm requires a preliminary interpolation of intensity over the 2×2-pixel neighborhood of the detected corners, which means that shortening the interpolation interval will improve the precision of detection, but the computation load will increase at the same time.

In this paper, we propose a new sub-pixel detector for X-corners. We firstly use a new operator based on Hessian matrix to find the pixel position of X-corner. Then a second order Taylor polynomial describing the local intensity profile around the corner is educed. The sub-pixel position of the corner can be determined directly by calculating the saddle point of this profile. Our new algorithm does not need surface fitting or intensity interpolation, which can simplify the computation greatly.

This paper is organized as follows. Section 2 reviews briefly the traditional sub-pixel algorithm for detecting X-corners. Section 3 describes our new sub-pixel detector in detail. Section 4 provides the computer simulation results. The contrast of accuracy and robustness between the new algorithm and the

WSCG SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

traditional algorithm is used to validate our new algorithm.

2. TRADITIONAL ALGORITHM

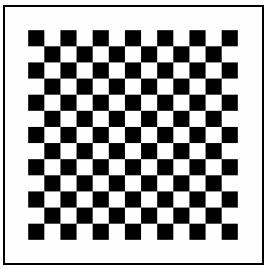


Figure 1: X-corners pattern

Fig. 1 shows a planar target with X-corners widely used in camera calibration. Traditional algorithm for detecting X-corners first finds their pixel positions by Harris detector based on a Hessian matrix looking for the auto-correlation matrix:

$$\mathbf{M} = \begin{bmatrix} \left(\frac{\partial \mathrm{I}}{\partial x}\right)^2 \otimes w & \left(\frac{\partial \mathrm{I}}{\partial x}\cdot\frac{\partial \mathrm{I}}{\partial y}\right)\otimes w \\ \left(\frac{\partial \mathrm{I}}{\partial x}\cdot\frac{\partial \mathrm{I}}{\partial y}\right)\otimes w & \left(\frac{\partial \mathrm{I}}{\partial y}\right)^2 \otimes w \end{bmatrix},$$

where w is a Gauss smoothing operator. Harris corner detector is expressed as

$$R = \det(\mathbf{M}) - \lambda(\mathrm{trace}(\mathbf{M}))^2 .$$

The X-corner is just the local peak point of R. Considering the local intensity around one X-corner in the image which has been smoothed by a Gauss low pass filter, we can see that the 3D shape of the intensity profile is just like a saddle (see Fig. 2c). The saddle point of this surface is just the X-corner to be detected. For each X-corner, a quadratic fitting of the local intensity profile can be obtained. The function can be expressed as

$$\mathrm{F}(x,y) = ax^2 + bxy + cy^2 + dx + ey + f \qquad (1)$$

This function turns out to be a hyperbolic, so the position of the saddle point can be determined by calculating the intersection of the two lines as follows:

$$\begin{cases} 2ax + by + d = 0 \\ bx + 2cy + e = 0 \end{cases}.$$

In general, this traditional algorithm allows for accuracies of the order of a few hundredths of a pixel [Luc00a], which can satisfy most applications of 3D machine vision. But the preliminary interpolation of intensity and the latter surface fitting aggravate the computation load of this algorithm. In section 3, we can see that our new algorithm is more simple and practical for X-corner sub-pixel detection.

3. NEW SUB-PIXEL DETECTOR

Model for X-Corner Intensity Profile

The model for the intensity of an ideal X-corner (see Fig. 2a) can be expressed as

$$f(x,y) = \begin{cases} 0, & xy \le 0 \\ 1, & xy > 0 \end{cases}.$$

Because of the effect of noises, the X-corners in practical target image are not consonant with this ideal model, so preliminary low pass filter is necessary in image processing, which can reduce the effect of noises and smooth the intensity profile around the corners. We select a standard Gauss operator here:

$$\mathrm{g}(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} .$$

Then the model for the intensity profile of an X-corner smoothed by Gauss operator is given by

$$\mathrm{r}(x,y) = \mathrm{g}(x,y) \otimes \mathrm{f}(x,y) = \Phi(x)\Phi(y) + \Phi(-x)\Phi(-y) \qquad (2)$$

where

$$\Phi(x) = \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{x} e^{-\frac{t^2}{2\sigma^2}} dt .$$

Operator for X-Corner Detection

Before calculating the sub-pixel location of X-corner, we need an operator to determine its pixel position. Harris detector is a universal operator for detecting all kinds of corners. But for X-corner, we can use a new shape operator deduced by the eigenvalues of Hessian matrix of the image function. Haralick has discussed that the eigenvalues of Hessian matrix correspond to the minimum and maximum second directional derivatives of a surface, and their associated eigenvectors are the directions in which the second directional derivative is extremized and are orthogonal to each other [Har00a]. Considering the neighborhood of a pixel in an image function as a surface patch, Hessian matrix is expressed as

$$\mathbf{H} = \begin{bmatrix} r_{xx} & r_{xy} \\ r_{xy} & r_{yy} \end{bmatrix},$$

where r_{xx}, r_{xy}, r_{yy} are the second order partial derivatives of the image function $\mathrm{r}(x,y)$. The two eigenvalues of Hessian matrix are

$$\lambda_1 = \frac{1}{2}\left(r_{xx} + r_{yy} + D\right),\ \lambda_2 = \frac{1}{2}\left(r_{xx} + r_{yy} - D\right).$$

Corresponding normalized eigenvectors are

$$\mathbf{n}_1 = \left[\sqrt{\frac{1}{2}\left(1 - \frac{f_{yy} - f_{xx}}{D}\right)} \quad \sqrt{\frac{1}{2}\left(1 + \frac{f_{yy} - f_{xx}}{D}\right)}\right]^T,$$

$$\mathbf{n}_2 = \left[\sqrt{\frac{1}{2}\left(1 + \frac{f_{yy} - f_{xx}}{D}\right)} \quad -\sqrt{\frac{1}{2}\left(1 - \frac{f_{yy} - f_{xx}}{D}\right)}\right]^T,$$

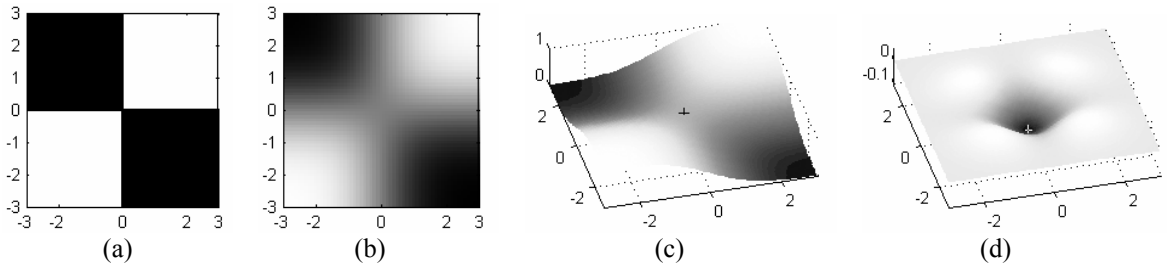


Figure 2: (a) An ideal X-corner consists of four neighboring black-and-white squares. (b) Smoothed X-corner. (c) 3D intensity profile around X-corner. The saddle point corresponding to the X-corner is marked with a '+'. (d) 3D profile of S around X-corner. The negative extremum point corresponding to the X-corner is marked with a '+'.

where

$$D = \sqrt{(r_{xx} - r_{yy})^2 + 4r_{xy}^2} .$$

For an X-corner point as Eq. 2, the maximum eigenvalue λ_1 of corresponding Hessian matrix is positive and the minimum eigenvalue λ_2 is negative, so we can get a new simple operator to detect the pixel position of an X-corner:

$$S = \lambda_1 \cdot \lambda_2 = r_{xx} r_{yy} - r_{xy}^2 .$$

Fig. 2d shows the 3D shape of S around an X-corner. The corner to be detected is the local negative extremum point of S. Based on this constraint, we can determine the pixel position (x_0, y_0) of an X-corner.

Sub-Pixel Detection

It is obvious that the real sub-pixel position of the X-corner must locate in the vicinity of (x_0, y_0). We suppose that its real position is $(x_0 + s, y_0 + t)$, where $(s,t) \in [-0.5,0.5] \times [-0.5,0.5]$. For each X-corner, we can use a second Taylor polynomial to describe the local intensity profile around it. Hence,

$$\begin{aligned} \mathrm{r}(x_0 + s, y_0 + t) &= r + \begin{pmatrix} s & t \end{pmatrix} \begin{pmatrix} r_x \\ r_y \end{pmatrix} \\ &+ \frac{1}{2} \begin{pmatrix} s & t \end{pmatrix} \begin{pmatrix} r_{xx} & r_{xy} \\ r_{xy} & r_{yy} \end{pmatrix} \begin{pmatrix} s \\ t \end{pmatrix} \end{aligned} \quad (3)$$

where r is the value of the image function $\mathrm{r}(x, y)$ at (x_0, y_0), r_x, r_y are the first order partial derivatives of $\mathrm{r}(x, y)$ at (x_0, y_0), r_{xx}, r_{xy}, r_{yy} are the second order partial derivatives of $\mathrm{r}(x, y)$ at (x_0, y_0).

As confirmed above, an X-corner is just the saddle point of the intensity profile. We can set the first order derivatives of Eq. 3 along s and t to zero:

$$\begin{cases} r_{xx} s + r_{xy} t + r_x = 0 \\ r_{xy} s + r_{yy} t + r_y = 0 \end{cases} .$$

Then we can get the sub-pixel position $(x_0 + s, y_0 + t)$ of the X-corner to be detected, where

$$s = \frac{r_y r_{xy} - r_x r_{yy}}{r_{xx} r_{yy} - r_{xy}^2} , \quad t = \frac{r_x r_{xy} - r_y r_{xx}}{r_{xx} r_{yy} - r_{xy}^2} .$$

Detection of X-corners in Discrete Space

For discrete image, only one modification has to be made, which is the implement of convolution in discrete space. Here we select standard Gauss kernel as the convolution mask. It is given by

$$\mathrm{g}(m,n) = \frac{1}{2\pi\sigma^2} e^{-\frac{m^2+n^2}{2\sigma^2}} .$$

The dimension of the mask is $(2N+1) \times (2N+1)$, where N is given by 4[σ]. Then the partial derivatives of discrete image can be obtained by convolving the original image with corresponding differential Gauss kernels.

4. EXPERIMENTAL RESULTS

The proposed algorithm has been tested on computer simulated data. We built a planar target image with 144 X-corners as shown in Fig. 1. The size of each square is 32×32 pixels, and the overall size of this image is 512×512 pixels. In order to simulate the real camera calibration, we get a second image derived from this original image through a projective operation as follows:

$$\mathbf{x}_2 = \mathbf{M}\mathbf{x}_1 ,$$

where

$$\mathbf{M} = \begin{bmatrix} 0.918 & 0.180 & -25.037 \\ -0.109 & 0.997 & 28.723 \\ 0 & 0 & 0.923 \end{bmatrix} .$$

Moreover, zero-mean white Gaussian noises are added to the resulting image to mimic the practical situation. Fig. 3 shows the resulting images. The standard deviation σ of the Gauss smoothing operator is set as 3, and the size of the kernel is 25×25. For each noise level, we perform 100 independent trials, and the results shown are the average. The contrast of the estimated errors between the traditional algorithm and the new algorithm is

summarized in Table 1, which indicates that our new algorithm is slightly more accurate and robust.

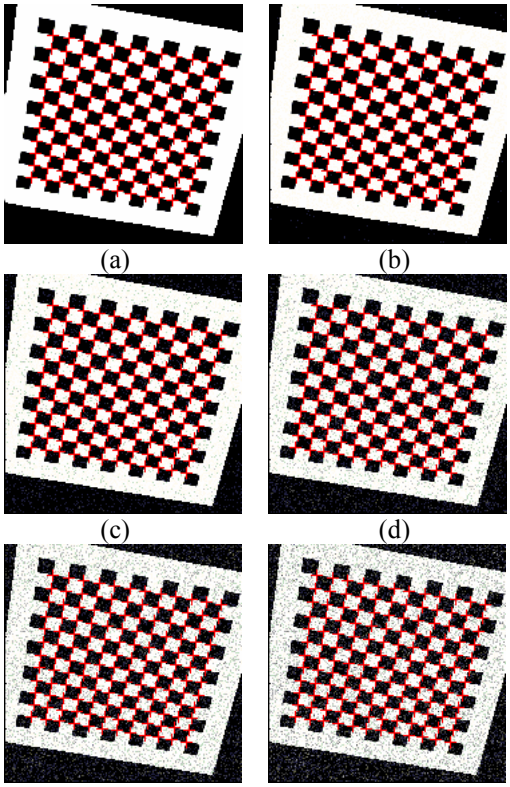


Figure 3: Images of the simulated target, where X-corners are marked as '+'. (a) Noise level $\sigma_n = 0$; (b) $\sigma_n = 0.04$; (c) $\sigma_n = 0.08$; (d) $\sigma_n = 0.12$; (e) $\sigma_n = 0.16$; (f) $\sigma_n = 0.20$.

Noise Level (σ_n)	Estimated Errors (σ_s)	
	Traditional Algorithm	New Algorithm
0	0.0088	0.0086
0.04	0.0331	0.0327
0.08	0.0644	0.0638
0.12	0.0958	0.0949
0.16	0.1279	0.1268
0.20	0.1598	0.1585

Table 1: The estimated errors of the detection for X-corners by two different algorithms: σ_s is defined as

$$\sigma_s = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left\|\hat{x}_k - x_k\right\|^2},$$

where x_k is the true position of X-corner, and $\hat{x}_k$ is the calculated position.

5. CONCLUSION

In this paper, we have proposed a new sub-pixel detector for X-corners in camera calibration targets. This algorithm consists of a new X-corner operator, followed by a second order Taylor polynomial describing the local intensity profile around the X-corner. The sub-pixel position of X-corner can be determined directly by calculating the saddle point of this polynomial. Neither intensity interpolation nor surface fitting is necessary, which simplifies the detection process greatly. Compared with the traditional method, our new algorithm is slightly more accurate and robust.

6. ACKNOWLEDGMENTS

This work was supported by a National Science Fund for Distinguished Young Scholars (50125518).

7. REFERENCES

[Dev00a] Devy, M., Garric, V., Orteu, J. J., Camera Calibration from Multiple Views of a 2D Object, Using a Global Nonlinear Minimization Method, Proc. of International Conference on Intelligent Robots and Systems, Grenoble, France, Sep. 1997, pp. 1583-1589

[Har00a] Haralick, R. M., Watson, L. T., Laffey, T. J., The Topographic Primal Sketch, The International Journal of Robotics Research, Vol. 2, No. 1, 1983, pp. 195-217

[Har01a] Harris, C., Stephens, M., A Combined Corner and Edge Detector, 4th Alvey Vision Conference, Manchester, UK, Aug. 1988, pp. 147-151

[Hei00a] Heikkila, J., Geometric Camera Calibration Using Circular Control Points, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 10, Oct. 2000, pp. 1066 – 1077

[Luc00a] Lucchese, L., Mitra, S. K., Using Saddle Points for Subpixel Feature Detection in Camera Calibration Targets, Proc. of the 2002 Asia Pacific Conference on Circuits and Systems, Singapore, Dec. 2002, pp. 191-195

[Nob00a] Noble, J. A., Finding Corners, Image and Vision Computing, Vol. 6, No. 2, 1988, pp. 121-128

[Suk00a] Sukanya, P., Takamatsu, R., Sato, M., A New Operator for Image Structure Analysis, Proc. of International Conference on Image Processing, Lausanne, Switzerland, Sep. 1996, pp. 615-618

[Zha00a] Zhang, Z., A Flexible New Technique for Camera Calibration, Technical Report, Microsoft Research, Mar. 1999

Towards Invariant Interest Point Detection of an Object

Md. Saiful Islam, Andrzej Sluzek, Zhu Lin
School of Computer Engineering
Nanyang Technological University
Singapore 639798.
Email: saiful@pmail.ntu.edu.sg, assluzek@ntu.edu.sg, danielin@pmail.ntu.edu.sg

ABSTRACT

Detection of some interest points on an object is useful for many applications, such as local shape description of the object, recognition of the object in clutter environment etc. The same object present in different images can have some geometric and photometric transformations with respect to one another. The detection method should be robust to all these transformations. We describe relative scale Harris method for interest point detection. This method is robust to linear geometric transformations. A threshold selection method is also described for invariance to intensity change, partial occlusion and clattered environment. Unlike multi-scale methods our method is fast enough to be suitable for real time applications.

Keywords: interest point, relative scale, invariance, repeatability

1. INTRODUCTION

A reference object and a test image are given. The reference object present in the test image can have some geometric and photometric transformation, with respect to the given reference object. Moreover, the surrounding of the object of interest could be cluttered with other objects and object itself could be partial occluded. In all these conditions, we like to detect similar sets of interest points on the reference object and on the object of interest in the test image. Harris detector [Har88] is a classical work on interest point detection. Later, it was improved by Schmid et al [Sch00] for better repeatability rate of interest points, in presence of relative rotation. These detectors fail when there is large scale change for the object of interest, present in the image. To alleviate the problem Mikolajczyk et al [Mik01], [Mik04] investigated scale-space interest point detector, using so called Harris-Laplacian method. But this method could be computationally quite expensive for real time applications.

In this paper, we introduce relative scale Harris method for scale invariant interest point detection. It is assumed that the relative scales of the same object presents in different images are known *a prior*. For intensity invariant interest points, a heuristic based method of threshold selection is proposed. So, our detection method is robust for different kind of geometric (translation, rotation and scale change) and photometric transformations (intensity scaling and intensity shift). This method is also robust for cluttered environment and partial occlusion. Unlike multi-scale interest point detection methods [Mik01], [Mik04], our method is computationally quite fast. The proposed method of interest point detection could be quite useful for many applications such as local shape description, object recognition in cluttered environment etc.

Our detection method is described in Section 2. Section 3 presents some experimental results to show the effectiveness of the method. Section 4 concludes the paper with future works.

Conference proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. METHOD

In this section, our goal is to detect two similar sets of points on the same object presents in two images, irrespective of different kinds of transformations. Section 2.1 presents relative scale Harris method for scale and rotation invariant interest point detection. Section 2.2 gives a threshold selection method for intensity invariant interest points. Section 2.3 deals with cluttered environment and partial occlusion.

2.1 Relative scale Harris method

The detection method depends on the **prior knowledge about relative scales** of the object, present in the images. We assign an arbitrary reference scale σ_R to the given reference object. A point (x', y') on the object present in the test image is related to a point (x, y) of the reference object by linear geometric transformations as given bellow:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a \\ b \end{bmatrix}$$

where s is arbitrary scaling factor, θ is arbitrary rotation and (a, b) is arbitrary translation. The **relative scale σ_I** of the object in the test image is a linear function of the scaling factor s.

$$\sigma_I = cs\sigma_R$$

where c is a constant. The value of c depends on the reference scale σ_R and the range of scale of consideration $[\sigma_{min} - \sigma_{max}]$, where $\sigma_{min} < \sigma_R < \sigma_{max}$.

In original Harris method [Har88], the image is first differentiated in two perpendicular directions and then integrated by a circular Gaussian window. In improved Harris version [Sch00], a 1D Gaussian kernel is convolved with the image for differentiation. Here, we use the relative scale of the object σ_I as the variance of Gaussian for Harris integration. The variance of Gaussian for Harris differentiation is $\sigma_D = k\sigma_I$, where k is a constant. The scale normalized auto-correlation matrix of Harris detector [Mik04], [Mik01] at a point $X = (x, y)$ of the image I is given by

$$N(X, \sigma_I) = \sigma_D^2 g(\sigma_I) \otimes \begin{bmatrix} I_x^2(X, \sigma_D) & I_x I_y(X, \sigma_D) \\ I_x I_y(X, \sigma_D) & I_y^2(X, \sigma_D) \end{bmatrix}$$

here, $g(\sigma_I)$ which is the circular Gaussian integration window at the scale σ_I is given by

$$g(\sigma_I) = \frac{1}{2\pi\sigma_I^2} e^{-\frac{x^2+y^2}{2\sigma_I^2}}$$

$I_x(X, \sigma_D)$ and $I_y(X, \sigma_D)$ are the partial derivatives of the given image in x and y direction respectively and can be found by convolving the image with the 1D Gaussian kernel.

$$I_x(X, \sigma_D) = h(\sigma_D) \otimes I(X)$$

$$I_y(X, \sigma_D) = (h(\sigma_D))^T \otimes I(X)$$

where $h(\sigma_D)$ is the 1D Gaussian first derivative kernel at the scale of σ_I

$$h(\sigma_D) = -\frac{x}{\sigma_D^3\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma_D^2}}$$

The measure of **corner response** at the point X and scale σ_I is

$$R(X, \sigma_I) = \det(N(X, \sigma_I)) - \lambda \mathrm{tr}^2(N(X, \sigma_I))$$

where λ is a constant. $R(X, \sigma_I)$ is positive in corner region and a point is selected as a **corner point** if it is the local maximum of the measure of corner responses [Har88], i.e. point X is a corner point if

$$R(X, \sigma_I) > 0 \text{ and } R(X, \sigma_I) > R(X_w, \sigma_I) \; \forall \; X_w \in W$$

where W is the 8-neighborhoods of the point X.

A corner point could be selected as an **interest point** if

$$R(X, \sigma_I) > T_c$$

where T_c is a constant threshold.

Experimental results for relative scale interest point detection are shown in Section 3.

2.2 Threshold selection for intensity invariant detector

In this section, we investigate the effect of intensity change on the detection method. The change in intensity may happen for different reasons such as different brightness of light source, direction of incident, and change in camera aperture etc. In this paper we only considered the effect of uniform intensity changes, i.e. intensity scaling and intensity shift.

$$I'(x, y) = sI(x, y) + c$$

where s and c are intensity scaling and shift parameters respectively.

The measure of corner response $R'(X, \sigma_I)$, at the point X and scale σ_I for the image I', is independent of intensity shift but the intensity scaling affects the response. It can be easily shown that the change in corner response is not linear for intensity scaling i.e.

$$R'(X, \sigma_I) = s^2 \det(N(X, \sigma_I)) - s^4 \lambda \mathrm{tr}^2(N(X, \sigma_I))$$

Here, we propose a heuristic based method to select the threshold for intensity invariant interest points. The threshold T_v' is selected based on the **normalized-corner response**. Consider, $\Lambda = \{X_i\}$ is the set of corner points for $I'(x, y)$ and $R'(X_i, \sigma_I)$ is the corner response at point X_i. The points in the set Λ are sorted in descending order i.e. $R'(X_i, \sigma_I) >= R'(X_{i+1}, \sigma_I)$ for $i = 1 \ldots. |\Lambda| - 1$; where $|\Lambda|$ is number of corner points. We get the normalized-corner response $Q(X_i, \sigma_I)$, normalizing $R'(X_i, \sigma_I)$ by $max(R'(X_i, \sigma_I)) = R'(X_1, \sigma_I)$ for $i = 1 \ldots. |\Lambda|$.

$$Q(X_i, \sigma_I) = \frac{R'(X_i, \sigma_I)}{R'(X_1, \sigma_I)}$$

Plotting the normalized-corner response $Q(X_i, \sigma_I)$ against the detected corner points we get **Q-Λ** histogram. From experiments it can be seen that Q-Λ histograms always have almost similar shape, irrespective of the objects contained in the image. Figure 1(a) and 1(b) show two images having

translation, rotation, scale change and intensity change, with respect to one another. Figure 1(c) shows the Q-Λ histogram for the image 1(b). X-axis of the histogram shows the sequence of detected corner points in the sorted set Λ and the Y-axis shows the corresponding $Q(X_i, \sigma_l)$. Now we select a threshold T_v' for $Q(X_i, \sigma_l)$, where T_v' is sufficiently large (for example 0.01 - 0.04) to discard most points on the portion of the curve parallel to X-axis (Figure 1(c)). So, a corner point is an interest point if $Q(X_i, \sigma_l) > T_v'$.

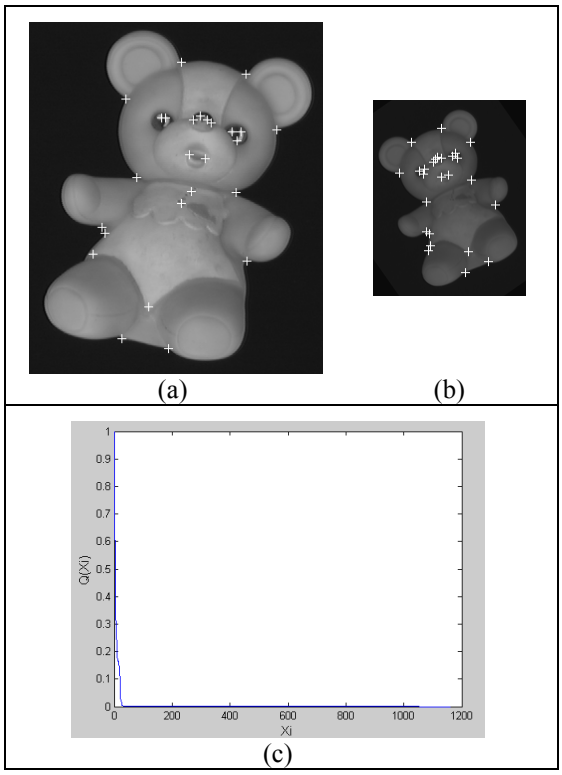


Figure 1: Threshold selection method a) The reference object and detected interest points. This image is borrowed from *RSORT* [Sel99]. b) Modified image (reference object is rotated (35^0), scale changed (50%), intensity scaled (0.5) and intensity shifted (+10)). c) Q-Λ histogram for the image in (b).

2.3 Cluttered environment and partial occlusion

Now, we should consider two other important factors for interest point detection: the given test image may contain other objects and the object of interest may be partially occluded (Figure 2). The threshold selection method is greatly affected by these two factors. In the case of cluttered environment, stronger corner points may come from other objects, affecting the normalization process. For Partial occlusion some of the strongest corner points may be absent due to the occlusion, again affecting the normalization process. In both cases our solution is to reduce the threshold, allowing more interest points. For large number of detected corner points due to the cluttering, we want to reduce the threshold slightly to ensure the higher repeatability rate. A particular choice for the final threshold T_v, for invariant interest point detection could be

$$T_v = \frac{T_v'}{\ln(|\Lambda|)}$$

where $|\Lambda|$ is the number of detected corner points in the set Λ.

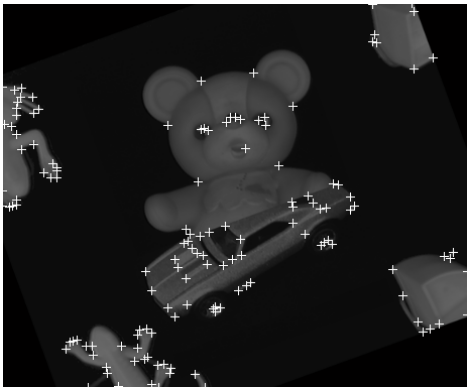


Figure 2: Detected interest points for partial occlusion and cluttered environment. Original images are taken form *RSORT* [Sel99] and then manipulated.

3. EXPERIMENTAL RESULTS

In order to evaluate our interest point detector we have used the repeatability rate criterion proposed by Schmid et al [Sch00]. Here, we can define the repeatability rate as the ratio of number of corresponding interest points pairs and the total number of points detected on the given reference object, with consideration of a localization error of ε. Let $\Lambda^{(r)} = \{X_i^{(r)}\}$ and $\Lambda^{(t)} = \{X_j^{(t)}\}$ be the sets of points detected on reference object and on the same object in the test image respectively. If $C(X_i^{(r)}, X_j^{(t)})$ is the number of corresponding points pairs then the repeatability rate is

$$\boldsymbol{rr} = \frac{C(X_i^{(r)}, X_j^{(t)})}{|\Lambda^{(r)}|} \times 100\%$$

where $|\Lambda^{(r)}|$ be the number of points in the set $\Lambda^{(r)}$.

We carried out experiments on several objects from two data sets: *RSORT* [Sel99] and *COIL-20* [Mur95]. The four objects used for the experiments are shown in Figure 3. The results are summarized in Table 1. One of the main objectives of this work is to detect scale invariant interest points. We changed the scale of each of the objects from 150% to 50% with a simultaneous change in rotation (35^0), intensity scaling by a factor 0.6, and intensity shift +10. We

computed average repeatability rate (**avg-*rr*)** for all four objects. Figure 4 shows the avg-*rr* for scale change. On the average, we got 86% avg-*rr*, which is much better than the improved Harris detector [Sch00] and better than Harris-Laplacian method [Mik01], [Mik04]. Our relative scale method is as fast as single step Harris method that is much faster than the scale-space method. We computed avg-*rr* for change of rotation angle (0^0 -180^0) and change of intensity (intensity scaling factor was changed from 1.6 to 0.4). We also carried out experiments for partial occlusion and cluttered environment as shown in Figure 2. Here, we have two objects of interest (bear and car) present in the image. Both of them have further scale, rotation and intensity change with respect to the given reference objects. The repeatability rate for the bear is 85% and car is 70% and avg-*rr* for these two objects is 77.5%.

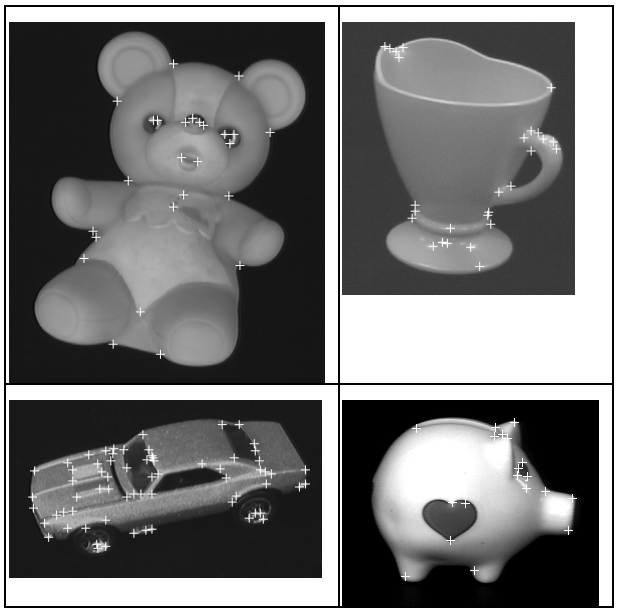


Figure 3: Four reference objects used for experiment. Detected interest points are also shown. Images are taken from *RSORT* [Sel99] and *COIL-20* [Mur95] dataset.

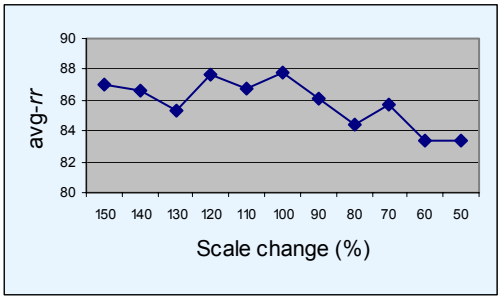


Figure 4: Average Repeatability rate (avg-*rr*) for scale change (150% - 50%) for the four objects in Figure 3.

Transform-ation /cond-ition type	**Range of change**	**Additional simul-taneous transfor-mation/condition**	**Average and range of avg-*rr* (%)**
Rotation	0^0 - 180^0	Scale 80% Intensity scaling 1.2 Intensity shift -5	86.86 [83.3 - 91.6]
Scale	150%-50%	Rotation 35^0 Intensity scaling 0.6 Intensity shift +10	85.84 [83.3 - 87.8]
Intensity scaling	1.6 – 0.4	Scale 120% Rotation 25^0	83.5 [83.5 - 83.5]
Cluttered & Partial Occlusion	Figure 2	Scale 80% Rotation 20^0 Intensity scaling 0.5	77.5 bear- 85% car – 70%

Table 1: Results for different transformations and conditions.

4. DISCUSSION

In this paper, we introduced the relative scale interest point detection method for scale invariant interest points. A heuristic based method was proposed to select threshold for invariance to intensity change, cluttered environment and partial occlusion. Our detection method gives better performance than the existing methods in term of repeatability rate and computing time. The method could be useful for many applications. For the relative scale method we assumed that the scale of the object is known in advance. Our method only deals with linear geometric transformations. Consideration of affine scale change is important for real-time applications. As future work we like to find invariant descriptors for the detected interest points. Such descriptors could be helpful for point to point matching.

5. REFERENCES

[Har88] Harris, C. and Stephens, M. A combined corner and edge detector. 4th Alvey Vision Conference, Manchester, UK, 1988.

[Mik04] Mikolajczyk, K. and Schmid, C. Scale & affine invariant interest point detectors, International Journal of Computer Vision, vol. 60, no. 1, pp. 63-86, 2004.

[Mik01] Mikolajczyk, K. and Schmid, C. Indexing based on scale invariant interest points. ICCV01, pp. 525-531, 2001.

[Mur95] Murase, H. and Nayar, S. Visual learning and recognition of 3D objects from appearance. International Journal of Computer Vision, vol. 14, no. 1, pp. 5-24, Jan 1995.

[Sch00] Schmid, C., Mohr, R. and Bauckhage, C. Evaluation of interest point detectors. International Journal of Computer Vision, vol. 37, no. 2, pp. 151-172, 2000.

[Sel99] Selinger, A. and Nelson, R.C. A perceptual grouping hierarchy for appearance-based 3D object recognition. Computer Vision and Image Understanding, vol. 76, no. 1, pp. 83-92, 1999.

Image-Based Point Rendering and Its Application to Color Editing Tool

Hiroaki Kawata
Keio University, SFC
5322 Endo, Fujisawa-city,
Kanagawa, 252-8520, Japan.
t02282hk@sfc.keio.ac.jp

Alexandre Gouaillard
CREATIS, INSA de Lyon
69621 Villeurbanne, France
alexandre.gouaillard@insa-lyon.fr

Masahiko Morita
Keio University, SFC
masahiko@sfc.keio.ac.jp

Kenji Kohiyama
Keio University, SFC
kohiyama@sfc.keio.ac.jp

Takashi Kanai
Keio University, SFC
kanai@sfc.keio.ac.jp

ABSTRACT

Advances in 3D scanning technologies have enabled the practical creation of hundreds of millions of points. In this paper, we describe a novel image-based point rendering algorithm only using points. Most of previous point rendering algorithms has to prepare normal vectors in advance to establish shading effects. Our algorithm is based on image processing and can calculate normal vectors on the fly in each frame using principal component analysis. Also, our algorithm is familiar with various other image processing algorithms. As an example we demonstrate an interactive color-editing tool for points.

Keywords

Point-Based Rendering, Image Processing, Principal Component Analysis, Color Editing.

1. INTRODUCTION

In recent years, a large number of scanned points can be acquired thanks to the development of scanning technology via 3D range image scanners. On the other hand, geometric processing techniques such as surface reconstruction from range images [Hoppe92] are needed to utilize points in the various Computer Graphics (CG) applications. Surface reconstruction is a laborious process and often requires a try-and-error task to make polygonal surfaces from such points suitable for practical use.

Point-based rendering has been a focus of constant attention to address the above issue. There are several advantages compared to the polygon-based rendering: The image quality of point-based rendering can now be approximately the same as polygon-based rendering whereas the data representation of points keeps compact because faces are not needed to render.

Short paper proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

Point-based rendering was firstly introduced by Levoy and Whitted [Levoy85]. In order to achieve the rendering quality as same as polygon-based rendering, a method to fill holes between points is required. One type of approaches for filling holes is to define a rectangle, a circle or an ellipsoid called as *splat* [Rusin00] or *surfel* [Pfist00, Zwick01] for each point. The other type of approaches based on image processing [Gross98, Kawat04a] has been also proposed.

On being related to the last advantage, there are several painting or surface editing approaches. [Agraw95] proposed 3D painting for scanned objects; however the algorithm is used for meshes. [Zwick02] developed an application for editing point set surfaces. In this application, color editing and surface editing for point set surface can be applied at interactive frame rate. [Adams04] proposed another painting method for point set surfaces. In this approach a virtual brush for painting, which is also defined by a point set, is used.

In this paper, we propose a color editing tool by using point based processing technique based on

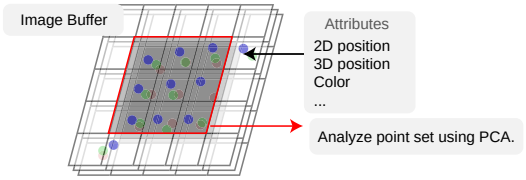


Figure 1. Image-Buffer and affecting range of PCA.

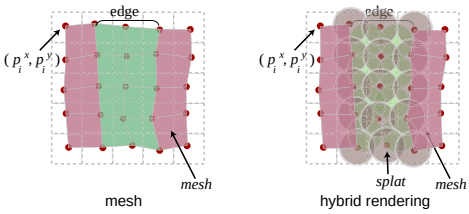


Figure 2. Hybrid rendering.

image processing. Our approach can render the image by **only** using position information of points without preparing normal vectors in advance. Our approach is called as Image-Based Point Rendering (IBPR) [Kawat04a].

The paper is organized as follows, firstly in Section 2, we describe an IBPR algorithm used in color editing tool on our approach. In Section 3, we propose color editing tools as an application of our approach. In Section 4, we show results of our approach. In Section 5, we conclude this paper and discuss about our approach.

2. IMAGE-BASED POINT RENDERING

In this section, we describe a brief overview of Image-Based Point Rendering (IBPR) algorithm described in [Kawat04a]. The input data of IBPR is only 3D positions $p_i = (x, y, z)\ (i = 1 \ldots n)$ of points. Also, color information $c_i = (r, g, b, a)$ can be attached in each point if needed. In IBPR an image buffer is provided to render points. We set the resolution of such an image buffer smaller than screen resolution (frame buffer). This approach is similar to the pull-push algorithm proposed by Grossman and Dally [Gross98]. Whereas the pull-push algorithm uses several multi-resolution images for filling holes, we use only one low-resolution image with arranging its resolution.

In the rendering process, we store each 2D coordinates $p_i = (p_i^x, p_i^y)$, color information c_i and a normal vector $n_i = (n_i^x, n_i^y, n_i^z)$ to a corresponding pixel of an image buffer. A normal vector is computed using stored 2D positions p_i in each frame (see [Kawat04a]). We determine the resolution of an image buffer (w_i, h_i) as follows:

$$w_i = w_s / s, h_i = h_s / s, \quad (1)$$

$$w = \frac{zn \cdot zf}{zf - zn} \cdot \omega + 1, \quad (2)$$

$$s = \frac{\sigma}{\tan(fov/2)} \cdot \frac{1}{w} \cdot \lambda.$$

where w_s, h_s are width and height of a frame buffer respectively. s is determined by the resolution of a point set and the distance from a view position to a point ω and a field of view fov. σ is the resolution of point set, λ is width or height, and zn, zf are the distance from a view position to a near plane and to a far plane respectively.

After all information of points is stored in an image buffer, we compute a normal vector and create a rectangle grid mesh from each pixel and its neighbor pixels only which points are stored. We use these meshes for shading which allow filling holes. Finally, we magnify an image buffer to the size of an original frame buffer for the actual rendering.

We describe a method to compute normal vectors for each pixel of an image buffer using principal component analysis (PCA). We use PCA here for only computing principal directions. In PCA computation process, we need to search a set of points for each pixel and its neighbor pixels. In our approach, we can search neighbor points by referring neighbor pixels of an image buffer efficiently. Figure 1 shows the range of pixels of an image-buffer needed to search neighbor points (multiple points can be stored for each element; these points are used in up-sampling process described later in Section 3).

In original IBPR [Kawat04a], holes are filled using a lower resolution image buffer. However, this process itself causes a blurring effect on a resulting image. We apply a hybrid rendering which both mesh and splat are used as drawing primitives. We describe the technique in Figure 2.

The details of the computation of normal vectors using PCA and hybrid rendering are also described in [Kawat04b].

3. COLOR EDITING

In this section, we propose a color-editing method based on IBPR with some extensions. In our approach, we can use general 2D image processing algorithms, because IBPR is much familiar with these algorithms. Since our approach uses an image buffer, general 2D image processing techniques can be applicable. In this paper, we focus on a stamp tool (Figure 3) for points. In a stamp tool, one region of an image is copied to another region. This tool can be used for the removal of noises of range images.

Figure 3. Stamp tool.

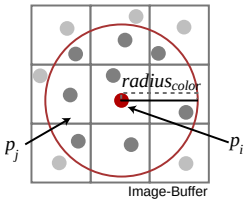


Figure 4. Color transformation.

Editing Process

In IBPR, a stamp tool is executed after all point information is stored to an image buffer (described in Section 2). We use this information for color editing directly. Next, we describe the process.

First, the user defines the source region of the image on screen space. 2D positions p_i and colors c_i included in the specified region are stored to a temporary buffer.

Second, the user defines the destination region of the image on screen space. In this region color information is updated according to the temporary buffer. Figure 4 shows such an updating process. We determine a color of p_i by using the distance between p_i and its neighbor points p_j. We also calculate a weight for each neighbor point by using a distance from p_i. A color is calculated by a weighted sum of colors in neighbor points. This calculation is done for setting a higher priority to a closer point.

We also define the region of interest $radius_{color}$. It is also used for computing a weight. The weight is computed as follows,

$$\omega_j = \frac{|p_j - p_i|}{radius_{color}}. \quad (3)$$

Then, a color c_i of p_i is determined as follows,

$$c_i = \frac{\sum_{j=1}^{n} c_j \omega_j}{\sum_{j=1}^{n} \omega_j}, \quad (4)$$

where value of $radius_{color}$ is defined by the user. This process gives a simple way to find neighbor points without any special data structure of point sets.

Up-Sampling

In the above process, if the density of points on the destination region is not enough, the blurring of a resulting image can occur. To address this problem, we apply up-sampling to increase the quality of a resulting image.

Up-sampling is used in most researches about point-based painting. For example in [Adams04],

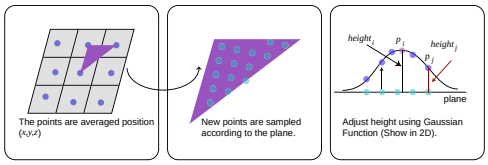


Figure 5. Up sampling process.

Normal | Up sampled | Source image

Figure 6. Result of up sampling.

dynamic up-sampling scheme is applied for less density regions. We use a simple method to improve the quality of destination regions in a stamp tool. In this method neighbor point regions are used to calculate new sampling points. Such regions are given by an image buffer. Our up-sampling method is described as follows:

1. We define a sampling plane based on neighbor points (Figure 5 left). The plane is defined by three positions which are taken from an image buffer.
2. Next, we calculate an interval of new sample points. Based on this interval, new points are sampled on this plane (Figure 5 middle).
3. We adjust a height from the plane of each new sampled point according to source 3D points (the positions of original 3D points) (Figure 5 right). Source 3D points are stored to an image buffer in advance. We also compute a distance $height_i$ from the plane to each source 3D point. We use a Gaussian function to determine the weight of source 3D point. A position p'_j of new sampled points are determined as follows:

$$r(d) = e^{-d^2/h^2}, \quad p'_j = n \cdot \{r(|p_i - p_j| / range) \cdot height_i\}, \quad (5)$$

where d is a distance from a source 3D point. h is a user-specified parameter and is set to 0.2. $range$ is also a parameter and is set according to an interval of up sampling. A direction n is a normal vector of the plane.

Figure 6 shows the comparison of resulting images between up-sampling and the original approach (up-sampling is not used). It can be seen that the pasted image is clearly displayed compared to the original approach.

4. RESULTS

In this section, we show the results of our approach. We evaluate our approach on Pentium 4

(a) (b)

Figure 7. The results of stamp tool.

3.2 GHz CPU and 1GB RAM PC. For the experiment we use an original range image of Stanford Bunny (362,272 points). Figure 3 shows the result of our point-based rendering with attached color attribute and the right upper of Figure 3 show a magnified image of the result with our stamp tool. The rendering time is 1.11 seconds.

Next, we show the results of color editing tool. We used Beetle range images with color information (559,327 points) for the experiment. We edit the regions of noises appeared in range images by pasting colors of other regions. Figure 7 (a) shows rendering results before editing operations with different view directions. Figure 7 (b) shows the results after editing operations. On the left upper of each figure, a magnified image to a modified region is shown. From these results, it can be seen that the regions of noises are corrected and are smoothly rendered.

5. CONCLUDING REMARKS

In this paper, we have described IBPR and its application to color editing tool. Our approach for point-based rendering is familiar with various 2D image processing algorithms. As one of examples, we have shown that a stamping tool can be efficiently implemented on our point-based rendering scheme. However, the rendering time is slow, but we can imagine that an image-processing part of our algorithms can be easily ported to a fragment program of GPU, which dramatically improves the rendering speed.

We have also demonstrated a stamp tool as an application to our IBPR. We think that we can implement other 2D image processing tools and also 3D geometry editing tools in the near future work.

ACKNOWLEDGMENTS

Stanford Bunny range images are courtesy of Stanford University Computer Graphics Laboratory.

REFERENCES

[Agraw95] M. Agrawala, A. C. Beers and M. Levoy. 3D painting on scanned surfaces. In Proc. 1995 Symposium on Interactive 3D Graphics, pp. 145-150, 1995.

[Adams04] B. Adams, M. Wicke, P. Dutré, M. Gross, M. Pauly and M. Teschner. Interactive 3D painting on point-sampled objects. In Proc. Eurographics Symposium on Point Based Graphics 2004, pp. 57-66, 2004.

[Gross98] J. Grossman and W. J. Dally. Point sample rendering. In Proc. Eurographics Workshop on Rendering 98, pp.181-192, 1998.

[Hoppe92] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In Proc. SIGGRAPH 92, pp. 71–78, 1992.

[Kawat04a] H. Kawata and T. Kanai. Image-based point rendering for multiple range images. In Proc. 2nd International Conference on Information Technology & Applications (ICITA 2004), pp. 478–483, 2004

[Kawat04b] H. Kawata, A. Gouaillard and T. Kanai. Point-based painterly rendering. In Proc. 2nd International Conference on Cyber Worlds 2004, 2004.

[Levoy85] M. Levoy and T. Whitted. The use of points as a display primitive. Technical Report 85-022, Computer Science Department, University of North Carolina at Chapel Hill, 1985.

[Pfist00] H. Pfister, M. Zwicker, J. van Baar and M. Gross. Surfels: Surface elements as rendering primitives, Proc. ACM SIGGRAPH 2000, pp.335-342, 2000.

[Rusin00] S. Rusinkiewicz and M. Levoy. Qsplat: A multiresolution point rendering system for large meshes. In Proc. ACM SIGGRAPH 2000, pp.343-352, 2000.

[Zwick01] M. Zwicker, H. Pfister, J. van Baar and M. Gross. Surface splatting. In Proc. ACM SIGGRAPH 2001, pp. 371-378, 2001.

[Zwick02] M. Zwicker, M. Pauly, O. Knoll and M. Gross. Pointshop3D: an interactive system for point-based surface editing. In Proc. ACM SIGGRAPH 2002, pp.322-329, 2002.

Occlusion Culling with Statistically Optimized Occlusion Queries

Vít Kovalčík
Faculty of Informatics
Masaryk University
Botanická 68a
602 00 Brno,
Czech Republic
xkovalc@fi.muni.cz

Jiří Sochor
Faculty of Informatics
Masaryk University
Botanická 68a
602 00 Brno,
Czech Republic
sochor@fi.muni.cz

ABSTRACT

This paper presents an effective algorithm for occlusion culling using hardware occlusion queries. The number of queries is optimized according to the results of the queries from several preceding frames. Parts of the scene which are found to be unoccluded in recent frames, are tested less often thus reducing the number of queries performed per frame. The algorithm is applicable to any kind of scene, including scenes with moving objects. The algorithm utilizes a tree structure containing objects in the scene.

Keywords

Visibility, real-time rendering, occlusion culling, occlusion query

1. INTRODUCTION

The number of details in virtual environments is still increasing and requires the use of "clever" algorithms for displaying a scene. Simple brute force approaches to rendering complex scenes, do not achieve interactive frame rates. Therefore algorithms performing occlusion culling have to be used. Such algorithms are able to detect objects, which are occluded by another object(s) from a user's point of view, and quickly discard these hidden objects from further processing.

There are many methods for performing occlusion culling (for more details see survey [Coh03a]). In recent years hardware based occlusion queries have started to be used. The query allows the programmer to indirectly access the Z-buffer and test if an object is visible or if it is shielded by already rendered objects. The tested area is usually just a bounding box of a fully detailed object. Based on the results of a query the application can decide whether or not to render a full object.

WSCG 2005 SHORT papers proceedings,
ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

Despite the simplicity of the occlusion query function, it is not trivial to use it correctly to gain a significant performance boost. Several algorithms for using occlusion queries have been developed.

One of the first was [Hil02a]. The scene is divided into a grid and each cell in the grid contains list of objects that are intersecting it. When rendering a frame, the grid is processed by layers in front to back order. For each cell the visibility of its bounding box is queried and in cases where the box is visible, objects in the cell's list are rendered, unless they had not been previously rendered because they intersected another already processed cell.

Another approach was described in [Hey01a]. In contrast with the previous method, this algorithm works in screen-space. The screen is divided into a regular low-resolution grid in which each cell remembers whether the relevant part of the screen is occluded. When rendering, objects in the scene are processed in front-to-back order and each object is tested against the occlusion grid. Because the occlusion state of the cells in the grid is updated only when it is necessary, this method is called "Lazy occlusion culling" (for more details see [Hey01a]).

Our method uses a scheme similar to one recently described in [Bit04a]. The whole scene is organized in a tree structure. During rendering the nodes are traversed and their visibility is tested using occlusion queries on their bounding boxes. The contents of a

node are rendered only where such a node is found visible.

In order to reduce the number of queries, we are using heuristics to detect nodes, which are probably visible and the algorithm renders such nodes without issuing a query.

2. OCCLUSION QUERY

Occlusion query is a hardware function present in modern graphic cards. The principle is simple: After a part of the scene is rendered onto screen and to the Z-buffer, there is a complex object to be rendered. Instead of rendering it, the displaying algorithm may choose to test whether the object is actually visible. This test is performed by retrieving the bounding box of the object and applying the occlusion query on it. The occlusion query returns the number of pixels that would have been visible, if the box had been rendered. This is done by comparing the box with stored Z-buffer values. If the number of possibly visible pixels is equal to zero, the bounding box is hidden by previously rendered object(s) and it is not necessary to render the complex object.

Unfortunately, the use of the occlusion query function is not that simple. Due to the buffering of data sent to a graphic card it often happens that previous parts of the scene have not been rendered at the time when a query is issued. However, occlusion queries might be processed asynchronously. It is possible to start a query, then render some object and use the result of the query later when it is available. Furthermore, it is not necessary to wait with the next query until the previous one is finished – the queries may run simultaneously. The processes are illustrated in Figure 1:

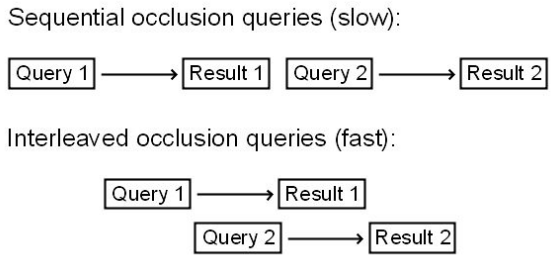


Figure 1: Illustration of simple and advanced use of the occlusion queries

3. ALGORITHM

Overview

Our algorithm requires the scene to be organized in a hierarchical tree structure. In our experiments we utilize an axis-aligned BSP tree, but octree, kD-tree or other similar structures could also be used. Each object in the scene is placed in exactly one node, that encom-passes the object fully and as tightly as possible.

When rendering a frame, the algorithm sets up a queue which holds nodes to be processed. Initially it contains the root node only. The queue is processed in natural order and for each node the algorithm decides whether the objects in a node will be discarded, queried for visibility or rendered without using an occlusion query.

The first case is straightforward. If the node is discarded, for example because of frustum culling, it is removed from the queue and the algorithm moves to the next node in the queue.

The second case is slightly more complicated. For some nodes, the algorithm may decide that an occlusion query is not necessary (the decision process will be described later). Objects which are stored in such a node are immediately rendered and the node is removed from the queue. Its descendants are placed in the queue at the position of the deleted node. The newly inserted nodes are sorted in front-to-back order. The algorithm then continues with the first descendant.

The third case is the most complex. If there is not enough information about the results of recent occlusion queries, it is difficult to predict, whether objects in a node should be rendered or not. At this point, the query is issued. The result of the query will be available after some time. It is possible to wait for the query to finish, but it would be a waste of time that could be used for processing another node. Hence the algorithm starts to process the next node in the queue instead of waiting for the result.

When a query finishes, depending on the result the node may be either skipped or the objects in the node are rendered and the node in the queue is replaced by its descendants. Because newly inserted nodes precede the currently processed node, the algorithm has to sometimes return and pass through the queue again. It can stop processing the queue at any time and return to the beginning of the queue, usually after the number of queries exceeds some threshold (about 20) and there is high probability that the first queries are already finished. It would be possible to stop processing the queue and return to the exact time when the result of the first query is available, but that would require additional checking of the status of the query, which in itself is time consuming.

The actual implementation uses two queues – one is the main query described above and the other is the queue with nodes with the occlusion query issued and not finished yet. Here is the overview in pseudocode:

```
queue.insert (root);
while (!queue.empty) {
  while (!queue.empty &&
  !query_queue.FirstNode.AnswerReady) {

    act_node = queue.FirstNode;
    action = CalcNodeAction (act_node);
```

```
    if (QUERY == action) {
      act_node.IssueQuery;
      query_queue.Add (act_node);
    } else if (RENDER == action) {
      RenderNode (act_node);
    }
    query.DeleteFirstNode ();
  }

  queue.SetPointerToStart;
  while (!query_queue.empty) {
    visible_pixels =
      query_queue.FirstNode.GetResult;
    SaveStatistics
      (query_queue.FirstNode,
      visible_pixels);
    if (visible_pixels > 0) {
      RenderNode
        (query_queue.FirstNode);
      queue.AddChildrenBeforePointer
        (query_queue.FirstNode);
    }
    query_queue.DeleteFirstNode;
  }
}
```

The *CalcNodeAction* function is crucial for the algorithm. It takes a node as a parameter and returns the value, which informs the rest of the algorithm, what actions should be taken for the given node. The actions are:

- *RENDER*. Objects in the node will be rendered without issuing a query.
- *SKIP*. The node is invisible, the objects in the node will not be rendered.
- *QUERY*. Occlusion query will be issued to determine if the node is visible or not.

Here is a pseudocode for a simple version of the *CalcNodeAction* function. This version does not utilize any results of the preceding occlusion queries.

```
if (FrustumCulled (node))
  return SKIP;
else if (ViewerIsInside (node))
  return RENDER;
else
  return QUERY;
```

Optimizations

The *CalcNodeFunction* can make an estimation (based on the results of recent occlusion queries) and change a return value from *QUERY* to *RENDER*. This estimation has to be done carefully, otherwise we could end up with rendering many objects, which are actually occluded. On the other hand, we do not want to use many occlusion queries as it may severely reduce performance.

The algorithm stores the results of several recent occlusion queries for every node and uses them to determine whether to initiate an occlusion query or not. The more times the node was found visible, the less often the query will be issued to check if it is still visible.

Pseudocode for the optimized *CalcNodeAction* function follows:

```
if (FrustumCulled (node))
  return SKIP;
else if (ViewerIsInside (node))
  return RENDER;
else {
  if (StatisticsTooOld (node))
    return QUERY;

  if (OCCLUDED ==
    LastQueryResult (node))
    return QUERY;

  occ_num = GetNumOfUnoccludedResults
    (node);
  not_query_time = last_query_time +
    BASE_TIME * (2 - 0.5^occ_num);

  if (actTime > not_query_time)
    return QUERY;

  return RENDER;
}
```

The BASE_TIME constant depends on the speed of viewer's movement and also on the type of scene. For higher speeds we select a lower number. In our tests, this constant was equal to 2/3 of second, which is a good compromise between forgetting the statistics too early and predicting the occlusion incorrectly because of too old information.

4. RESULTS

All tests were performed on a computer with Intel Pentium 4/2.0 GHz processor, 512 MB of RAM and ATI Radeon 9700 with 128 MB of memory.

Three scenes were tested. The first one (Figure 2) was the power plant model containing nearly 2 million triangles. It is a smaller version of the UNC's power plant model. Unfortunately, it was not possible to render the full model with its 12 million triangles in interactive frame rate with this configuration.

The second scene (Figure 3) was a computer-generated library with shelves containing nearly 40,000 books with a total of over 10 million triangles. The shelves does not have back sides, so the books were the main occluders.

The third scene (Figure 4) consists of 5,000 randomly placed teapots that are made of 32 million triangles.

Three different rendering algorithms were used:

- No occlusion culling. This algorithm uses only view-frustum culling, it does not use any kind of occlusion culling.
- Simple occlusion culling. The algorithm starts with the root of the scene hierarchy and traverses

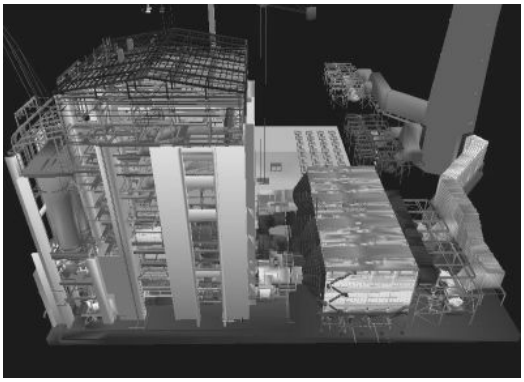


Figure 2: The power plant model (first test scene)

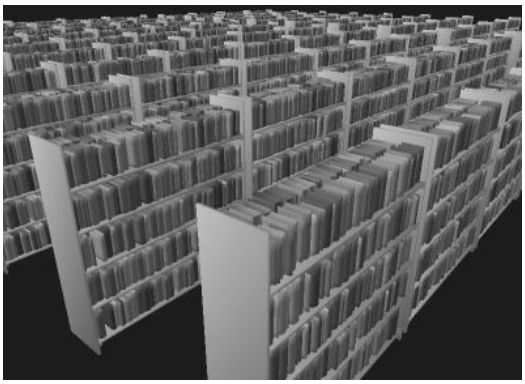


Figure 3: The library (second test scene)

the hierarchical structure in a top-down manner to the leaves. Before rendering a node, the occlusion query is utilized to get the visibility of the node's bounding box. If the box is invisible, the node and it's descendants are not going to be rendered.

– Statistical occlusion culling. This is the algorithm described in the previous section.

For each scene we run these algorithms to render a fly-through containing several hundred frames and we measured the total rendering time. The result are shown in Table 1:

Scene	No OC	Simple OC	Stat. OC
Power plant	43	37	**22**
Library	14	7	**4**
Teapots	33	11	**11**

Table 1. Time (in seconds) to fly through several scenes using three different rendering algorithms.

The occlusion culling algorithm with statistically controlled occlusion queries gives the best results in most cases. However, sometimes it may be slower than "Simple OC" because the statistics of a recent occlusion may give a false prediction and unnecessarily render many objects. But the statistics are used only for a brief interval, so "Stat. OC" is usually only a little slower in these problematical cases.

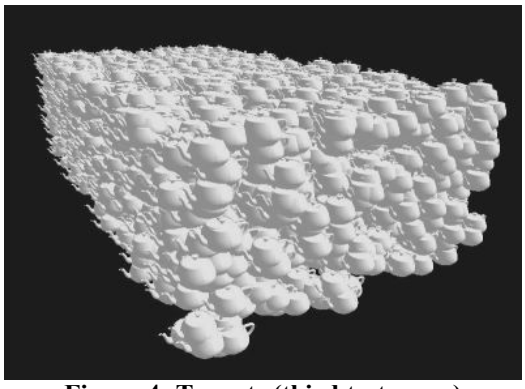


Figure 4: Teapots (third test scene)

5.CONCLUSION AND FUTURE WORK

We have described a new occlusion culling algorithm, which is able to render scenes up to four times faster than algorithms using view-frustum culling only. It can operate on any type of scene, including a scene with moving objects.

There are many directions for future work. The algorithm can be improved by better ordering of the queries, or by improving prediction function based on the recent statistics.

6. ACKNOWLEDGMENTS

This work was supported by grant VZ MSM 143300003.

7. REFERENCES

[Bit04a]Bittner, J., Wimmer, M., Piringer, H., Purgathofer, W. Coherent Hierarchical Culling: Hardware Occlusion Queries Made Useful. Computer Graphics Forum (Proc. Eurographics 2004) 23(3):615-624, 2004.

[Coh03a] Cohen-Or, D. Chrysanthou, Y., Silva, C.T., Durand, F. A survey of visibility for walkthrough applications. IEEE Transactions on Visualisation and Computer Graphics 9, 3. 2003.

[Cor02a] Corrêa, W.T., Klosowski, J.T., Silva, C. T. Fast and Simple Occlusion Culling. Game Programming Gems 3, Charles River Media, 2002.

[Hey01a] Hey, H., Tobler, R.F., Purgathofer, W. Real-Time Occlusion Culling with a Lazy Occlusion Grid. Technical Report TR-186-2-01-02, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 2001.

[Hil02a] Hillesland, K., Salomon, B., Lastra, A., Manocha, D.. Fast and simple occlusion culling using hardware-based depth queries. Technical Report TR02-039, Department of Computer Science, University of North Carolina, 2002.

Evaluation of Modern Dynamic Programming Algorithms for Real-time Active Stereo Systems

J-D. Nahmias d.nahmias@cs.ucl.ac.uk
A. Steed a.steed@cs.ucl.ac.uk
B. Buxton b.buxton@cs.ucl.ac.uk
Department of Computer Science, University College London
Gower Street, London WC1E 6BT, UK

ABSTRACT

Stereo Vision has been an active research field that has produced a variety of different algorithms. Unfortunately most of the algorithms that produce superior results rely on non-linear optimization techniques that are very computationally expensive, and therefore not feasible to use for real-time applications such as tele-immersion. This paper will examine a number of real-time stereo algorithms based on dynamic programming (DP) used in conjunction with structured light in order to improve the quality and facilitate the correspondence search. We will examine some of the early DP algorithms as well as the more recent work produced by [Criminisi et al] for the purpose of Gaze manipulation for teleconferencing in the context of 3d reconstruction. We present adaptation of [Birchfield et al] DP algorithm to work with structured light. Additional we look at spatial-temporal support region for computing matching costs.

Keywords: Dynamic Programming, Stereo, Tele-immersion, Real-Time

1. INTRODUCTION

Tele-immersion relies on creating the illusion that its users are sharing a collaborative virtual space. This is a difficult task to achieve but, with recent advances in the fields of real-time 3d graphics, networking and computer vision, this has become a feasible goal and very promising results are starting to surface [Mull04]. To maintain the illusion of tele-immersion, users should perceive each other in 3d. This is not limited to viewing each other in stereo since motion parallax should be preserved (i.e. when a user moves the display should update itself appropriately rendering images from the new point of view).

Ideally, one would like to capture the 3d information in real-time, thereby removing the requirement to model deformations and not compromising behavioral realism while maintaining an accurate structural realism. Stereo vision has been a very active field of research for a few decades and has produced good results as well as a huge wealth of literature on the subject. For an excellent overview of modern stereo reconstruction techniques readers are referred to [Scha02].

This paper will focus on examining Dynamic Programming algorithms with structured light for stereo reconstruction owing to their relatively low computational cost when compared to other stereo reconstruction algorithms based on non-linear optimization algorithms. The traditional DP algorithms [Cox96] will be presented as well as more recent variations. Their relative performance will be examined in a structured and unstructured light context. [Zhang03] proposed a framework for space-time stereo that utilizes temporal coherence in-order to further improve stereo reconstruction with structured light. The algorithms presented in this paper will similarly be extended into the space-time domain.

Short paper proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. DP STEREO FORMULATION

This section will cover the standard DP stereo formulation [Cox96] as well the variations due to [Birch98] and [Crim03].

Traditional DP Stereo

Given a pair of rectified images $I_l(x)$ and $I_r(y)$ representing the left and right images at the xth and yth pixels respectively for a given scanline, it can be shown from [Forsyth02] that the depth of a given pixel is inversely proportional to its disparity $(x-y)$. The problem is therefore one of

correspondence. Using the *uniqueness* and *monotonic ordering* constraint DP algorithms will solve the disparity by minimizing a cumulative cost function *C(l, r)* defined as follows:

$$C(l,r) = \min \begin{cases} C(l-1,r) + OccCost \\ C(l-1,r-1) + M(l,r) \\ C(l,r-1) + OccCost \end{cases}$$

Here, *OccCost* is a parameter of the system and defines a penalty for occlusions and *M(l,r)* is a cost function that defines the dissimilarity between two pixels *l* and *r* of the left and right scanline respectively. It is quite common for *M(l,r)* to be the Sum of Squared Difference or SSD defined as:

$$M(x,y) = \sum (I_l(x) - I_r(y))^2$$

The recurrence in *C(l, r)* defines the allowed moves in the forward pass of the DP algorithm, namely: one horizontal occluded move, one diagonal matched move, and one vertical occluded move. After initialization of the cost matrix the DP algorithm iterates through each cell within the constraint network calculating *C(l, r)* and storing a backwards link to the previous cell containing the minimum cost. Once the cost matrix has been calculated the second stage of the algorithm is a backwards pass that follows all the stored links to produce the minimum cost path and therefore the disparity for that scanline. This is repeated for each scanline in the pair of images and a disparity map is produced.

Birchfield's DP Algorithm

The [Birch98] algorithm differs from the traditional DP algorithm in a few key ways, summarized as follows. Firstly, the cumulative cost function and the dissimilarity measure are changed and defined as follows:

$$C(l,r) = \min \begin{cases} C(l-1,r) + OccCost \\ C(l-1,r-1) + M(l,r) + Match\,\mathrm{Re}\,ward \\ C(l,r-1) + OccCost \end{cases}$$

where:

$$M(x_i, y_i) = \max\{0, I_L(x_i) - I_{\max}, I_{\min} - I_L(x_i)\}$$

$$I_{\min} = \min\left(I_R^-, I_R^+, I_R(y_i)\right), I_{\max} = \max\left(I_R^-, I_R^+, I_R(y_i)\right)$$

$$I_R^- = \frac{1}{2}\left(I_R(y_i) + I_R(y_i - 1)\right), I_R^+ = \frac{1}{2}\left(I_R(y_i) + I_R(y_i + 1)\right)$$

The dissimilarity function *M(x,y)* measures how well the intensity at *x* fits the linearly interpolated region around *y*.

Another change is the addition of a constraint that intensity variation accompanies depth discontinuities. An intensity variation is defined as any set of three pixels whose min and max levels vary more than four. This threshold is very low and is intended to prevent the algorithm from making poor choices in regions of the image that do not contain much information. It also specifies on which side the depth discontinuity must lie with respect to the intensity variation and also requires occlusions to be accompanied by the intensity variation on the appropriate side.

The cost matrix is also computed in a more efficient manner by computing the minimal cumulative cost of reaching neighbouring cells through the particular cell currently being evaluated. The computational cost is equivalent to that of the traditional DP algorithm [Cox96]. However, it permits the algorithm to prune the cost matrix and further reduce the number of cells that need to be evaluated and thereby reducing the runtime to $O(n\Delta \log \Delta)$.

Once the cost matrix has been calculated, the initial estimates of the disparities are further refined by post processing steps. Firstly outliers are removed by a mode filter. Then the disparities regions are grown in the x and y axis based on reliability, the stopping criteria is intensity variation. This paper will show in section 4 that these post-processing steps can cause problems when used in conjunction with structured light.

Criminisi's DP Algorithm

In [Crim03] a new DP algorithm is proposed with the motivation of creating a depth map in order be used in conjunction with an IBR technique that morphs two images to create a new image from a different view point. This paper evaluates this algorithm from the point of view of 3d reconstruction. The algorithm uses a three-plane graph, a left occluded plane L, a matched plane M and a right occluded plane R (see Figure 1).

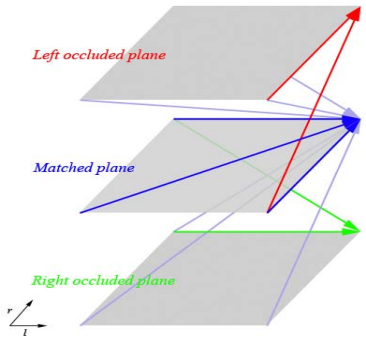


Figure 1 Adapted from [Crim03] showing the 13 allowed moves of the algorithm

This model allows a total of thirteen moves in the DP and has the advantage of allowing a much finer grain control of penalty costs. The following defines the cumulative cost functions $C_L(l,r), C_M(l,r), C_R(l,r)$ for each plane L, M and R respectively:

$$C_L(l,r) = \min\begin{cases} C_L(l,r-1) + \alpha \\ C_M(l,r-1) + \beta \end{cases}$$

$$C_M(l,r) = M(l,r) + \min\begin{cases} C_M(l-1,r) \\ C_L(l-1,r) + \beta \\ C_R(l-1,r) + \beta \\ C_M(l,r-1) \\ C_L(l,r-1) + \beta \\ C_R(l,r-1) + \beta \\ C_M(l-1,r-1) \\ C_L(l-1,r-1) + \beta \\ C_R(l-1,r-1) + \beta \end{cases}$$

$$C_R(l,r) = \min\begin{cases} C_R(l-1,r) + \alpha \\ C_M(l-1,r) + \beta \end{cases}$$

Here, α is the cost of moving within an occluded plane and β the cost of making a transition between planes. *M(l,r)* is a windowed normalized cross-correlation: $M(l,r) = (1 - M'(l,r))/2$ where

$$M'(l,r) = \frac{\sum (I_L - \bar{I}_L)(I_R - \bar{I}_R)}{\sqrt{\sum (I_L - \bar{I}_L)^2 \sum (I_R - \bar{I}_R)^2}}.$$

In [Crim03] the dissimilarity matrices are stacked across all the scanlines and Gaussian smoothed with a kernel orthogonal to both left and right scanlines.

3. IMPLEMENTATIONS

This section will cover the various implementations as well as certain variations that had to be made in order for the algorithms to work with structured light. A possible framework for space-time stereo using structured light described in [Zhang03] motivated the extension of the implemented algorithms into the space-time domain. However the space-time support windows were not sheared and skewed as described in [Zhang03]. The results of these various implementations will be presented in section 4.

The [OpenCV] implementation was used for the [Birch98] DP algorithm and modified while the [Crim03] DP algorithm was implemented from scratch without using Gaussian smoothing of the dissimilarity matrix. Support for SSD dissimilarity measure over spatial and temporal windows was added to the [OpenCV] implementation. The structured light patterns broke some of the assumptions made in the post-processing steps of

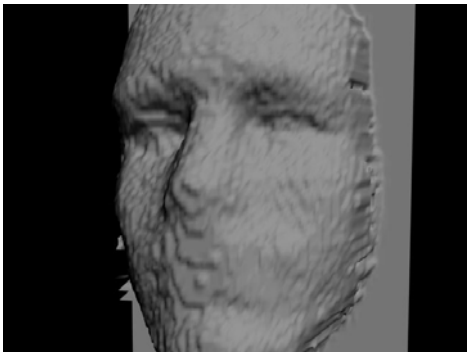


Figure 2 3d Reconstruction based on obtained disparity map using [Crim03] algorithm and space-time window as well as regular triangulation

[Birch98] described in section 2. The structured light pattern removes intensity variations along the y-axis and therefore one of stopping criteria for the region growing of the disparities along the y axis is violated. These were subsequently removed with the one exception of the mode filter.

The structured light pattern was generated as described in [Zhang03] using a 16bit Reflected Gray Code [Bit76] that is subsequently shuffled to produce high frequency change in both the spatial and temporal domains. The patterns are then smoothed and projected using an Infocus DLP projector at a resolution of 800x600 and are cycled through at the same frame-rate as the cameras performing capture.

The images were captured using two Balser A602fc cameras at a resolution of 640x480 with a quantization depth of 8bits. The camera calibration and image rectification was performed using the [Boug01] Matlab toolkit.

Finally the best disparity map was used to create 3d surface using two different triangulation algorithms. Figure 2 shows the 3d surface created using a regular triangulation.

4. EXPERIMENTS AND RESULTS

A polystyrene head was used for the experiments. The [Birch98] and [Crim03] algorithm were tested against the following dataset

- A pair of naturally lit images using spatial window.
- A pair of images lit with a structured light pattern using a spatial window
- A set of images lit with a time varying structured light pattern using a symmetrical spatio-temporal window of size 5x5x8.

The [Birch98] algorithm was tested using both SSD and interpolated cost functions, while the [Crim03] was tested using the cross-correlation cost function.

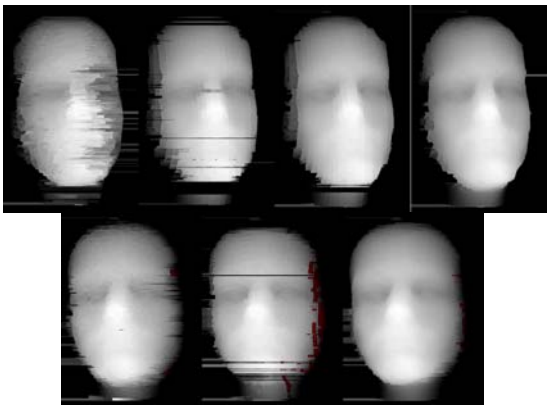


Figure 3 top left to right: [Birch98] algorithm, without light; with light; with light SSD 5x5; with light and a SSD 5x5x8 spatio-temporal, bottom left to right [Crim03], without light, with light 5x5, with light 5x5x8

All results were evaluated qualitatively by examining the disparity maps. However most of the differences in quality are quite evident on visual inspection of the disparity maps.

Figure 3 illustrates all the results. Using structure light improves the algorithm significantly, and, again, using a spatio-temporal window improves it even further. The results become interesting when comparing the [Birch98] algorithm against the [Crim03] algorithm under different conditions. Without structured light, the [Crim03] algorithm produces vastly superior results in comparison to the [Birch98] algorithm. However, when structured light is used with a spatial window the differences become much less pronounced. When comparing the performances of the algorithms with spatio-temporal windows the differences become even more subtle. Under these conditions, the performance criteria need to be clearly defined and some quantitative measures need to be applied.

From these results, one can clearly state that using the given structured light pattern improves each algorithm provided. One can make a similar statement with regards to using a spatio-temporal window with the structured light. However because the spatio-temporal window is not sheared or skewed it implicitly assumes a static scene. The statement that using a symmetric spatio-temporal window leads to an improved performance thus cannot, without further investigation, be made with regards to dynamic scenes.

5. CONCLUSION & FUTURE WORK

This paper has shown that structured light can be a powerful tool for the improvement of the DP algorithms described in Section 2. The paper has also highlighted some of the benefits in using temporal information when evaluating dissimilarities between pixels for static scenes. However, it has also left some questions unanswered, such as whether the benefits of the [Crim03] algorithm in the unstructured case are due to the fact it uses a normalized cross-correlation dissimilarity function or may they be mainly attributed to its three plane DP graph approach. Further work still needs to be carried out to evaluate these algorithms with dynamic scenes and properly investigate whether or not the [Crim03] algorithm is superior to the modified [Birch98] algorithm when used with structured light and with a spatio-temporal SSD dissimilarity function.

ACKNOWLEDGEMENTS

The first author was supported by funding associated with the EPSRC funded Interdisciplinary Research Collaboration (IRC) project EQUATOR (EPSRC Grant GR/N15986/01)

REFERENCES

[Bit76] J. R. Bitner, G. Ehrlich, and E. M. Reingold, "Efficient generation of the binary reflected gray code and its applications," *Communications of the ACM*, vol. 19, no. 9, pp. 517–521, 1976.

[Birch98] S. Birchfield and C. Tomasi, Depth Discontinuities by Pixel-to-Pixel Stereo, *Proceedings of the Sixth IEEE International Conference on Computer Vision (ICCV), Mumbai, India, pages 1073-1080, January 1998*]

[Boug01] J.-Y. Bouguet. Camera Calibration Toolbox for Matlab.http://www.vision.caltech.edu/bouguetj/calib doc/index.html, 2001.

[Cox96] I.J. Cox, S.L. Hingorani, and S.B. Rao. A maximum likelihood stereo algorithm. *Computer vision and image understanding*, 63(3):542–567, 1996.

[Crim03] A.Criminisi, J. Shotton, A. Blake, P.H.S. Torr. Gaze-Manipulation for One-to-one Teleconferencing. *Int. Conf. on Computer Vision, ICCV*, Nice, France, 2003

[Forsyth02] D. Forsyth, J. Ponce: Computer Vision: A Modern Approach. *Prentice Hall ISBN 0130851981*

[Mull04] J. Mulligan, N. Kelshikar, X. Zampoulis, K. Daniilidis, Stereo-based Environment Scanning for Immersive Telepresence, submitted to the *IEEE Transactions on Circuits and Systems for Video Technology Volume 14, Issue 3, March 2004, pages 304-320.*

[OpenCV] Open Computer Vision Library http://sourceforge.net/projects/opencvlibrary/

[Scha02] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. on Computer Vision*, 47(1):7–42, 2002.

[Zhang03] L. Zhang, B. Curless, and S.M. Seitz. Spacetime stereo: Shape recovery of dynamic scenes. In *Conference on Computer Vision and Pattern Recognition*, 2003.

GEOMETRICAL ACCURACY OF BAYER PATTERN IMAGES

Roland Perko and Philipp Fürnstahl
Graz University of Technology
Inffeldgasse 16, A–8010 Graz
Austria

Joachim Bauer and Andreas Klaus
VRVis Research Center
Inffeldgasse 16, A–8010 Graz
Austria

ABSTRACT

Modern digital still cameras sample the color spectrum using a color filter array coated to the CCD array such that each pixel samples only one color channel. The result is a mosaic of color samples which is used to reconstruct the full color image by taking the information of the pixels' neighborhood. This process is called demosaicking. While standard literature evaluates the performance of these reconstruction algorithms by comparison of a ground-truth image with a reconstructed Bayer pattern image in terms of grayscale comparison, this work gives an evaluation concept to asses the geometrical accuracy of the resulting color images. Only if no geometrical distortions are created during the demosaicking process, it is allowed to use such images for metric calculations, e.g. 3D reconstruction or arbitrary metrical photogrammetric processing.

Keywords

Bayer pattern demosaicking, Geometrical accuracy, Evaluation

1 INTRODUCTION

Commercially available digital still cameras are based on a single CCD sensor overlaid by a color filter array (CFA) which gives the possibility to capture a color image with only one CCD sensor. Several types of color mosaics have been implemented in the past, whereat the most common CFA is called the Bayer pattern [Bayer76] and is shown in figure 1. The scheme results in 25% red and blue and 50% green coverage of the array. A real example of an image captured by a CCD sensor that is equipped with a Bayer pattern filter is shown in figure 2. It results in an image mosaic of three colors, where the missing color pixels have to be interpolated to get a complete full RGB color image. This reconstruction is called *demosaicking*.

The paper is structured as follows. First, the state

Conference proceedings ISBN 80-903100-7-9
WSCG2005, January 31-February 4, 2005
Plzen, Czech Republic.

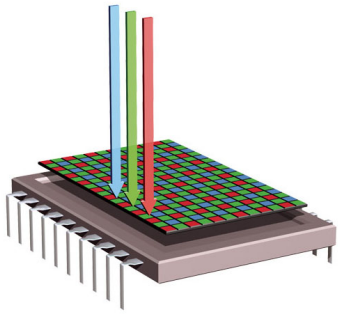
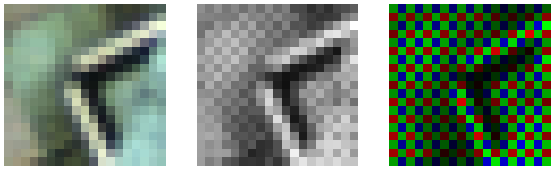


Figure 1: Concept of color acquisition using Bayer pattern (image from © 2003 Foveon, Inc. Used with permission). Digital sensor equipped with a color filter array, where every pixel only records one color instead of three.



(a) (b) (c)

Figure 2: Principle of Bayer pattern images. (a) Small detail of a conventional color image with 19×19 pixels. (b) This image is captured by a single CCD sensor equipped with a Bayer pattern filter. The mosaic is strongly visible. (c) Image (b) where each sample is plotted with the color from the Bayer pattern.

of the art of Bayer pattern demosaicking is presented in section 2. Next, the standard accuracy evaluation method is described in section 3. In section 4 the novel geometrical accuracy evaluation method and results are given. Finally, concluding remarks are made in section 5.

2 STATE OF THE ART

The Bayer color filter was patented in 1976 by [Bayer76]. To interpolate color values at each pixel, Bayer proposed simple bilinear interpolation. At the beginning of the development of digital still cameras, [Cok87] suggested to use a constant hue-based interpolation, since pixel artifacts in the demosaicking process are caused in sudden jumps in hue. [Freem88] then proposed to use a median-based interpolation of the color channels to avoid color fringes. In 1993 Hibbard filed a patent to adaptively interpolate a full color image by using an edge-based technique. The patent was approved two years later [Hibba95]. Meanwhile, [Laroc94] got their edge-based method approved, which can be seen as an extension to Hibbard's approach. [Hamil97] used the concepts of both edge-based methods and created a combination and extension of these approaches. [Chang99] proposed a simple, however promising method using a bigger local neighborhood to define the gradients. The difficulty of Bayer pattern demosaicking is still a hot topic in the computer vision community, e.g. see [Malva04]. A good survey is found in the review article [Raman02].

3 EVALUATION

The standard evaluation concept for Bayer pattern demosaicking methods is to start with a color image I_{ref}. This image is converted to a Bayer pattern image I_{BP}, that is then reconstructed to a full color image I_{res} by using a demosaicking method. This image can now be compared with the reference image I_{ref}. Figure 3 illustrates the evaluation setup.

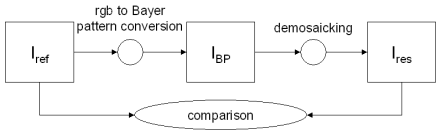


Figure 3: Concept of Bayer pattern demosaicking evaluation setup.

Several metrics are defined, to allow a quantitative comparison. The root mean square error (RMSE) counts the gray value differences of the images in DN and is defined as follows

$$RMSE(I_{\text{ref}}, I_{\text{res}}) = \sqrt{\frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} (I_{\text{ref}}(i) - I_{\text{res}}(i))^2} \tag{1}$$

where $\mathcal{N}$ is the neighborhood containing all pixels of the image and $i = (x, y)$ is the index for one single pixel. The RMSE is calculated for each color channel and the total RMSE is defined by the sum of the single RMSE values. [Raman02] suggests to use the RMSE in the L*a*b* color model, which has the advantage that color differences in this color model match with human perception. This error metric is defined as

$$RMSE_{\text{L}^*\text{a}^*\text{b}^*}(I_{\text{ref}}, I_{\text{res}}) = \sqrt{\frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} (\Delta L^*(i))^2 + (\Delta a^*(i))^2 + (\Delta b^*(i))^2} \tag{2}$$

where $\Delta L^*(i) = L^*_{I_{\text{ref}}}(i) - L^*_{I_{\text{res}}}(i)$, $\Delta a^*(i) = a^*_{I_{\text{ref}}}(i) - a^*_{I_{\text{res}}}(i)$ and $\Delta b^*(i) = b^*_{I_{\text{ref}}}(i) - b^*_{I_{\text{res}}}(i)$.

All these error metrics are defined globally for the whole image. Therefore, a small total RMSE may not be directly related to a good demosaicking results. In homogenous image areas the RMSE is near to zero, whereas it is larger near edges. Therefore, an additional error metric is introduced, which calculates the RMSE per color channel only in the neighborhood of Canny edges [Perko04]. In the evaluation this error metric is called RMSE at edges.
The results of the discussed demosaicking methods are given in table 1 for an aerial image.

Approach	RMSE					RMSE at edges		
	red	green	blue	total	L*a*b*	red	green	blue
nearest neighbor	7.0	5.4	6.1	18.5	9.9	22.8	17.1	19.5
bilinear	3.8	2.5	3.0	9.3	4.7	11.5	7.4	8.8
Cok logarithmic	2.6	2.5	2.1	7.2	4.1	7.6	7.4	5.8
Cok linear	2.5	2.5	2.1	7.1	4.0	7.2	7.4	5.7
Hibbard	2.5	2.4	2.2	7.1	4.0	7.0	6.7	5.9
Laroche	2.2	2.1	2.0	6.3	4.2	6.1	5.9	5.5
Hamilton	2.1	1.4	1.7	5.2	3.5	5.9	3.5	4.5
Chang	1.9	1.3	1.5	4.7	3.3	5.2	3.5	3.9

Table 1: Demosaicking results for an aerial image with 1500×480 pixel. A part of this image is shown in figure 2 (a). Given are RMSE in DN for all color channels and total RMSE, the RMSE at edges in DN for all color channels and the RMSE in L*a*b* color space.

In figure 4, the color fringes occurring at the edges are visible, above all for the simple demosaicking approaches (a) and (b). Visually the methods by Hamilton and Chang perform best. The numerical evaluation is not very surprising: As

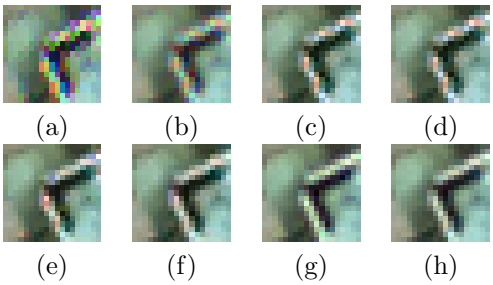


Figure 4: Demosaicking results for different methods. A detail with 19×19 pixel is used. (a) Nearest neighbor (b) Bilinear (c) Cok logarithmic (d) Cok linear (e) Hibbard (f) Laroche (g) Hamilton (h) Chang. Simple methods like (a) and (b) produce color fringes at edges, whereas edge-base approaches (e)-(h) converges more and more to the original image.

expected the nearest neighbor methods perform worst, followed by the bilinear and constant hue-based interpolation. The edge-based approaches outperform the non-adaptive ones and give better results according to their complexity. The algorithm of Chang gives the best results.
Of course the green color channel is reconstructed with the smallest error, since already 50% of the green data is available in the mosaic. Also worth to mention is that, as expected, the RMSE at edges is significantly higher (about a factor of 3) than over the whole channel.

4 GEOMETRICAL ACCURACY

For metric digital cameras the geometrical accuracy of the resulting images is essential. The higher the frequencies in an image, the more artifacts will occur after demosaicking. Since non-linear interpolation is used in the edge-based approaches, the question is, whether the geometry is changed by this procedure. The motivation for this evaluation comes from the observation sketched in figure 5. Strong image artifacts occur caused by non-linear interpolation. Therefore, several algorithms based on the image geometry are performed to answer this question. In this evaluation, the following test setup is used: A color image I_{RGB} is converted to grayscale using standard YIQ color model yielding to the reference grayscale image I_{ref}. On the other side, the color image I_{RGB} is converted to a Bayer pattern image I_{BP}, which is then reconstructed to the full color image $I_{\mathrm{RGB}'}$ and also converted to grayscale, resulting in I_{res}. If the demosaicking process does

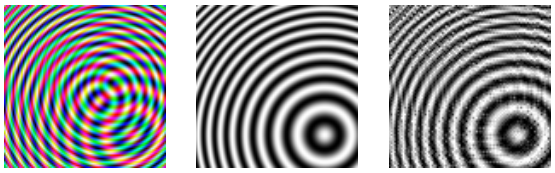


Figure 5: Hypothesis that the demosaicking process may introduce geometrical distortions. (a) Synthetic color image containing high frequencies. Each color channel contains concentric circles with varying center and frequencies. (b) Red color channel of the input image. (c) Red color channel of reconstructed Bayer pattern image using method by Hamilton.

not change the geometry, the grayscale images I_{ref} and I_{res} should now have a very similar geometry. This concept is illustrated in figure 6. Two tests are performed to determine the geometrical aspects:

(i) Subpixel matching: According to this concept two images are generated where one of them is subpixel translated by a given subpixel shift (e.g. $(-0.3, -0.2)$ pixel). These resulting images are then matched using the algorithm by [Gleas90]. Now, if there are differences in the geometry, the image pair based on the Bayer pattern should give a bigger error than the pair based on the input image.

(ii) Subpixel corner detection: The image pairs are generated as in the subpixel matching test. Then subpixel Harris corners are extracted in both images and matched using nearest neighbor assignment.

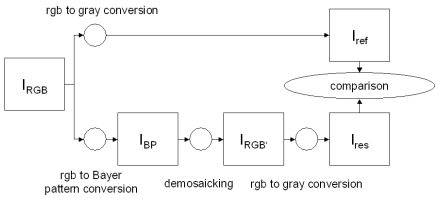


Figure 6: Concept of geometrical Bayer pattern demosaicking evaluation setup.

Both tests are performed on ten different images and the average results are chosen. Figure 7 shows the mean and the standard deviation of the errors in x and y coordinates for the subpixel matching test and figure 8 for the corner detection test. The errors are given in pixel for the original image and for eight demosaicking methods. As expected the nearest neighbor method introduces the largest errors and should not be used. The seven other methods give errors comparable to the original image, however the method by Hibbard

performs worse. Standard methods like Hamilton and Chang perform very good by producing no additional error in comparison to the original image.

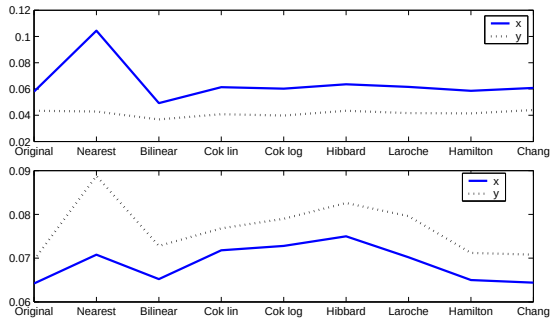


Figure 7: Results of the subpixel matching test in pixel. The mean error values are shown on the top and the standard deviation on the bottom for the original image and for eight Bayer Pattern demosaicking methods.

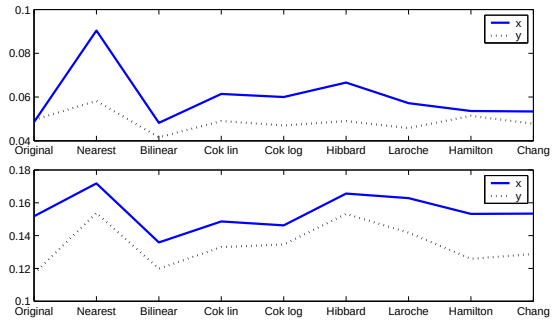


Figure 8: Results of the subpixel corner detection test in pixels. The mean error values are shown on the top and the standard deviation on the bottom for the original image and for eight Bayer Pattern demosaicking methods.

5 CONCLUSION

This paper describes the difficulty of reconstructing the missing color samples of a Bayer pattern image. The presented results lead to two basic conclusions: First, there are algorithms for demosaicking which give very good results and are computationally not very expensive, namely the algorithms by Hamilton and Chang. Second, the demosaicking process does not create geometrical distortions, so that e.g. stereo matching produces the same results as on true color images. Both aspects are very important for metric computer vision, consequently a camera equipped with a Bayer pattern filter for color image sensing is a useful approach.

References

[Bayer76] Bryce E. Bayer. Color imaging array. U.S. Patent 3,971,065, July 1976.

[Chang99] Ed Chang, Shifun Cheung, and Davis Pan. Color filter array recovery using a threshold-based variable number of gradients. *Proceedings of SPIE (San Jose, CA)*, 3650:36–43, January 1999.

[Cok87] D.R. Cok. Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal. U.S. Patent 4,642,678, 1987.

[Freem88] W. T. Freeman. Median filter for reconstructing missing color samples. U.S. Patent 4,774,565, February 1988.

[Gleas90] Shaun S. Gleason, Martin A. Hunt, and W. Bruce Jatko. Subpixel measurement of image features based on paraboloid surface fit. *Proc. Machine Vision Systems Integration in Industry, SPIE, Boston MA*, 1386, November 1990.

[Hamil97] J.F. Hamilton and Jr.J.E. Adams. Adaptive color plane interpolation in single sensor color electronic camera. U.S. Patent 5,629,734, 1997.

[Hibba95] R. H. Hibbard. Apparatus and method for adaptively interpolating a full color image utilizing luminance gradients. U.S. Patent 5,382,976, 1995.

[Laroc94] C. A. Laroche and M.A. Prescott. Apparatus and method for adaptively interpolating a full color image utilizing chrominance gradients. U.S. Patent 5,373,322, 1994.

[Malva04] Henrique Malvar, Li-wei He, and Ross Cutler. High-quality linear interpolation for demosaicing of Bayer-patterned color images. *IEEE International Conference on Speech, Acoustics, and Signal Processing*, 2004.

[Perko04] Roland Perko. *Computer Vision for Large Format Digital Aerial Cameras*. PhD thesis, Graz, University of Technology, October 2004.

[Raman02] Rajeev Ramanath, W. Snyder, G. Bilbro, and W. Sander. Demosaicking methods for bayer color arrays. *Journal of Electronic Imaging*, 11(3), July 2002.

Photographic Depth of Field Blur Rendering

Pichard Cyril
DUBOI
221 bis Bld Jean Jaurès
92100, Boulogne, France
cyril@duboi.com

Michelin Sylvain
SISAR Team
University of Marne-la-Vallée
6, cours du Danube
77700, Serris, France
michelin@univ-mlv.fr

Tubach Olivier
DUBOI
221 bis Bld Jean Jaurès
92100, Boulogne, France
tubach@duboi.com

ABSTRACT

The purpose of our study is to introduce a new method for depth of field blur simulation used for compositing real and synthetic images in a realistic way. Existing methods do not simulate accurately the depth of field blur produced by photographic equipment. The problem is twofold: photometric and geometric. Existing methods based on convolution do not produce a realistic photometric rendering compared to real photograph and the photographic blur contains patterns associated with the diaphragm shape, neither handled by existing methods. Our method, based on image processing, addresses these two issues by following the photographic image creation process. It can be used on synthetic or real images, essentially for special effects compositing.

Keywords

Depth of field blur, Image compositing, Image processing, Special effects, Boke

1. INTRODUCTION

Depth of field is an inherent phenomenon of optical systems. Cinema special effects intensively mix and compose images from different sources: computer graphic images, photographs, matte-painting, stock shot and real shot. The goal is to obtain the maximum of realism in the composite. One aspect of realism is the depth of field blurs matching in composited images, so frequently the depth of field blur has to be simulated on sharp images. The depth of field blur has some visual characteristics that existing methods do not handle. Photographers know that defocused lights tend to take on the characteristic shape of the camera's aperture. This phenomenon is not simulated by any existing method. Moreover, the lighter values tend to swallow dark values on defocused images, and methods based on convolution do not deal with this issue.

After introducing previous work, there follows an explanation as to how we simulate the lens, and a presentation of a method that takes into account visual photographic defocus.

Conference proceedings ISBN 80-903100-7-9
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. PREVIOUS WORK

Image processing

Potmesil and *al.* [Pot81] were the first to present a method able to generate depth of field blur. This method is based on convolution with a varying kernel size dependant on the point depth on the pixel. Kernel matrices are computed using characteristics of the confusion circles created by a thin lens. Shinia [Shi94] improved Potmesil's method by adding ray-tracing to compute masked pixels.

Computer graphics

Cook et *al.* [Coo84] describe an anti-aliasing ray-tracing method that can be used to simulate depth of field blur and motion blur. The idea here is to use anti-aliasing rays to simulate light beams. This method is named distributed ray-tracing. Kolb, Mitchel and Hanrahan [Kol95] describe a camera model based on physical properties of lenses. Their principle is to follow light rays through every optical component of the lens from film to scene. Interactive depth of field blur [Hae90] [Nei99] can be achieved with OpenGL using an accumulation buffer. This method is based on accumulating different images taken from near viewpoints. There are no physical properties behind this method, so the simulation is not accurate. Heidrich, Slussallek, Seidel [Hei97] extend this principle to improve the quality of the rendering by computing the characteristics of the various accumulated views.

3. LENS SIMULATION

A good review on lenses can be found in [Ray02]. To simulate the lens, we use the thin lens model

described by Potmesil [Pot81]. We first add a diaphragm shape to this model.

Diaphragm

The diaphragm is an aperture of varying size generally placed inside the lens. Each aperture diaphragm has a specific shape due to its conception. Diaphragms are generally made with blades' leaves that form an iris. The more blades there are the more circular the shape is. The side effect is the modeling of light beams' shapes, which affects the form of the light stains projected on the film. To simulate the form of the aperture, we use a Bezier curve. These kind of curves can handle every form of existing diaphragm. Rounded corners of diaphragms can be simulated accurately.

4. FILM SIMULATION

Physical phenomenon

Physically, the film records exposure values (in lux/s-1). The film is chemically processed and generally scanned to obtain a digital image. The final image I is in pixel values, with no physical unit. There is a non-linear mapping between pixel values and exposure values [Deb97].

On the film, each disk of light mixes with the others and produces the image, blurry or not. Physically, we can consider that a sort of convolution is done with exposure values. Methods like Potmesil's use convolution of pixel values. We can notice in Figure 2 that the convolution of pixel values does not produce the same visual result like defocused image does in Figure 3. The convolution of pixel values appears darker. Our idea is to simulate this phenomenon using inverse mapping of the film. We want to mix pixel values in a different space value, ideally in the exposure space. Unfortunately the conversion curves between exposure values and pixel values are unknown and impractical to obtain for most of the source images we use in compositing environment: film, photographs, matte-painting, stock-shots... Moreover, some exposure values cannot be recorded on the film because they are too high, for instance, light peaks. Therefore, they are value-saturated and we cannot obtain their exposure information without high dynamic range information [Deb97]. We propose here a simple method to simulate this conversion without using conversion curves and high dynamic range information.

Simple algorithm for film rendering

Our first idea is based on the fact that the bright values tend to swallow darker values, as with dilation. This phenomenon shows the importance of the apparent brightness. The brighter the pixel is, the bigger the effect. In order to characterise the brightness of a pixel $I(I_r, I_g, I_b)$ we use the digital luminance commonly defined as follows:

$$L(I) = 0.3 \times I_R + 0.59 \times I_G + 0.11 \times I_B$$

Figure 1. Original image

Figure 2. Gaussian blur

Figure 3. Defocused image

Note that this luminance does not have any physical property. The conversion we propose is simple. We apply a conversion function F to the initial pixel value. This function F is used to control the swallowing of the dark values and the boosting of light values. Simple conversion functions can be used for F like gamma function, or look-up table handled by Bezier curves. Processed pixels values are then pre-multiplied by the digital luminance of the source pixel. Then, an optical effect can be simulated, as occurs in exposure values: motion blur,

depth of field blur, flare effects, etc. There follows an explanation of how to make realistic depth of field blur using this conversion. Note that we have to store the pixel luminance contribution during operation, therefore, we use a special channel. To convert back in pixel values, we have to divide by the luminance contribution, and apply the inverse function F^{-1} (see Fig.4).

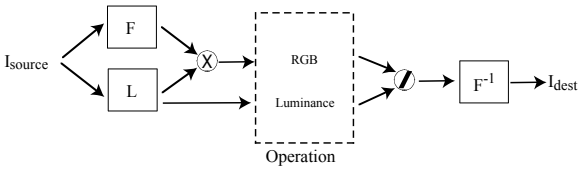


Figure 4. Film rendering principle

5. DOF BLUR ALGORITHM

Principle

Our algorithm is a mixture between Potmesil's method and OpenGL methods [Hae90][Hei97]. Like Potmesil's method we use an array of matrices and the method is based on image processing. As with OpenGL methods, we use the principle of the accumulation buffer. The algorithm works in 3 stages:

- Matrices creations
- Accumulation of light stains
- Final rendering of accumulation buffer into a frame buffer.

Mask matrices creation

The mask matrices M_i are made by rasterizing the Bezier diaphragm shapes. Figure 5 shows different rasterized masks matrices. For each depth, we have to create a mask matrix. The size of the mask calculated in pixels depends on the image format size and the size of the stain. For the current application we only store 256 matrices of different sizes, with the same shape, each associated with an alpha value. A curve is used to convert pixel depth to alpha values. Computer artists can manipulate this curve. To fill the mask matrices, we do not use intensity distribution as with Potmesil's method; we only use black or white values. This choice has been made for several reasons: light distribution for circular shapes is known but for unknown shapes the light distribution is undefined.

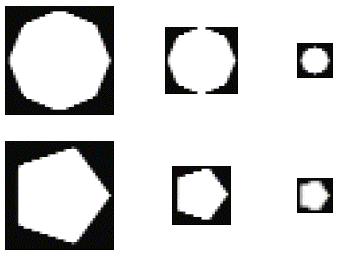


Figure 5. Rasterized Bezier diaphragms

Light distribution depends on light waves, which are difficult to obtain with pixel values. Moreover, the use of intensity distribution would increase computing time and the visual benefit would be nominal when taking the time into account.

First stage

Our algorithm follows the photographic image creation process. The processing consists of accumulating each light stain produced by each theoretical point of the scene. Theoretical points are represented by the source image pixels. This accumulation is done in an accumulation buffer (see Fig.6). The algorithm works as follows:

1. For each pixel $I(p')$ of the source image
    1. Compute luminance $L(I)$,
    2. Compute value $I_F = F(I_{source})$,
    3. Multiply L with I_F: $I_{premult} = L(I) \times I_F$
    4. Accumulate the resulting value $I_{premult}$ in the accumulation buffer through the appropriate mask M_i.
    5. Also accumulate luminance contribution L through the mask in a special channel.

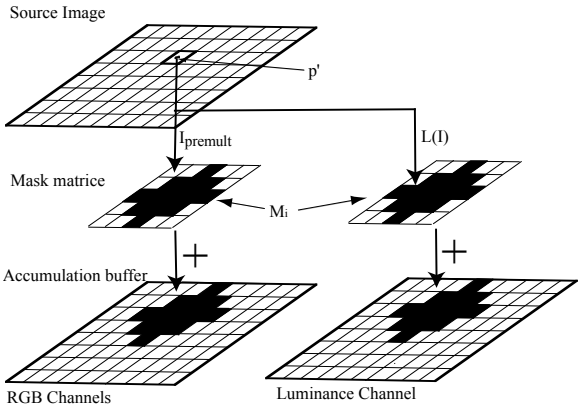


Figure 6. Accumulation principle

Second stage

The second stage is necessary to return to pixel values. This can be seen as the "chemical process". When the accumulation of confusion stain values is finished, the result is copied back from the accumulation buffer into the frame buffer, dividing by the contribution of luminance stored in the special channel.

6. RESULTS

Figure 7 shows the real defocused image of the scene and Figure 8 the original image processed with our algorithm. We can see that the method produces the same kind of visual effect. But we have to correct the colour of the original image to obtain similar values and to digitally remove the bridge fences in order to obtain this image. We predict the result would be

better in terms of colour and visual effects if we were to use high dynamic range images.

Figure 7. Defocused image

Figure 8. Defocus simulated

The second issue is that we generally do not have an image depth map. So for Figure 8 we used a constant value of mask. We can notice that the effect seems bigger at the bottom of Figure 8 than in Figure 7. This is because of the constant depth mask used.

For large size of defocus, we have to use multi-resolution techniques because computing time can be too important: it is possible to reach half an hour for a 3000x2000 16bit/channel image on a 1Ghz Pentium 4.

7. CONCLUSIONS

Our method can produce visual characteristics of photographic defocus by simulating diaphragm shape and film response. It can be used on real and synthetic images to produce a photo-realistic depth of field blur. This method tends to produce satisfactory results in terms of user satisfaction and has been intensively used on several French movies: “Immortel” by director Enki Bilal, “Un long dimanche de fiançailles” by director Jean Pierre Jeunet, and many others.

Nevertheless, we have made many assumptions: the shape does not turn with the depth, distribution of intensity is constant, the size of the image does not vary with focusing, etc. None of these assumptions provide enough visual improvement compared to the time increase. In any case, this method is included in our image compositing software DUTRUC, so computer artists can pre-process original images in order to obtain a satisfactory result.

8. REFERENCES

[Coo84] Cook, R., and Porter, T., and Carpenter, L. Distributed Ray Tracing. In SIGGRAPH 84 conf.proc., Computer Graphics, pp.137-144, 1984.

[Deb97] Debevec, P.E., and Malik, J. Recovering High dynamic range radiance maps from photographs. In Computer Graphics and Interactive Techniques conf.proc., ACM Press, pp.369-378, 1997.

[Hae90] Haeberly, P., and Akeley, K. The accumulation buffer : Hardware support for high-quality rendering. In SIGGRAPH 90 conf.proc., Computer Graphics, ACM Press, pp.309-318, 1990.

[Hei97] Heidrich, W., and Slusallek, P., and Seidel, H. An image-based model for realistic lens systems in interactive computer graphics. Graphic Interface 97, pp.68-75, 1997.

[Kol95] Kolb, C., and Mitchell, D., and Hanrahan, P. A realistic camera model for computer graphics. In SIGGRAPH 95 conf.proc., Computer Graphics, ACM Press, pp.317-324, 1995.

[Nei99] Neider, J., and Davis, T., and Woo, M. OpenGL Programming Guide : the official guide to learning OpenGL, version 1.2. Addison-Wesley, 1999.

[Pot81] Potmesil, M., and Chakravarty, I. A lens aperture camera model for synthetic image generation. In SIGGRAPH 81 conf.proc., Computer Graphics, ACM Press, pp.297-305, 1981.

[Ray02] Ray, S.F., Applied Photographic Optics, third edition. Focal Press, 2002.

[Shi94] Shinia, M. Post-filtering for depth of field simulation with ray distribution buffer. In Graphic Interface 94 conf.proc., Canadian Computer Communications Society, pp.39-66, 1994.

Nailing and Pinning: Adding Constraints to Inverse Kinematics

Mardé Greeff, Jörg Haber, Hans-Peter Seidel
Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken, Germany
{greeff, haberj, hpseidel}@mpi-sb.mpg.de

ABSTRACT

Inverse kinematics is commonly applied to compute the resulting movement of an avatar for a prescribed target pose. The motion path computed by inverse kinematics, however, often differs from the expected or desired result due to an underconstrained parameter space of the degrees-of-freedom of all joints. In such cases, it is necessary to introduce additional constraints, for instance by locking a joint's position and / or rotation. We present a method to fix a joint in terms of position and explain how to incorporate these constraints into the inverse kinematics solution.

Keywords: animation, inverse kinematics, positional / rotational constraints

1 INTRODUCTION

Many Computer Graphics applications, such as virtual environments, computer games, and interactive stories, feature animated characters, for instance humans and animals. To create animated sequences, the animator should be able to position and move all parts of the character. In many animation systems, a *skeleton* (or *articulated figure*) of the character is used to specify positions and motion. Such skeletons consist of rigid links (denoted as segments in H-Anim [HAWG] terminology) connected by joints. Usually, the articulated figure has a hierarchical structure, where each joint has its own coordinate system and is positioned relatively to the coordinate system of its parent. An articulated figure can often be divided into kinematic chains (limbs) where each chain has one end that is free to move, called the end-effector. To obtain a specific configuration of the articulated figure, each joint needs to be set to the correct rotation angle to obtain the required position. Specifying the configuration of the figure by rotating each joint one by one down the hierarchy is denoted as *forward kinematics* (FK). In contrast, when the position and orientation of a specific end-effector is given, the rotation angles of all joints further up the hierarchy can be computed using *inverse kinematics* (IK).

We present a method to pin a joint to a position in space and explain how to incorporate these constraints into the inverse kinematics solution.

SHORT papers proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2 RELATED WORK

The Resolved motion-rate method introduced by Whitney [Whi69], is one of the methods that are frequently used to solve the IK problem. Many extentions have been proposed, such as the pseudo-inverse method [MK85], the Jacobian transpose method [Wel93] and the selectively damped least squares method [BK03]. Two approaches making use of the resolved motion-method are the weighting strategy and the task-priority approach. With the weighting strategy, such as in [BMW87], when tasks get into conflict, the algorithm will distribute the residual error among the tasks according to their weight. Therefore no task is exactly satisfied unless one task's weight is higly dominant with respect to other weights. With the task-priority approach, conflicts are dealt with directly at differential level. When all goals cannot be satisfied simultaneously, the task with the highest priority reaches its goal while the residual error of the other tasks are minimized [BB98]. We use the task-priority approach.

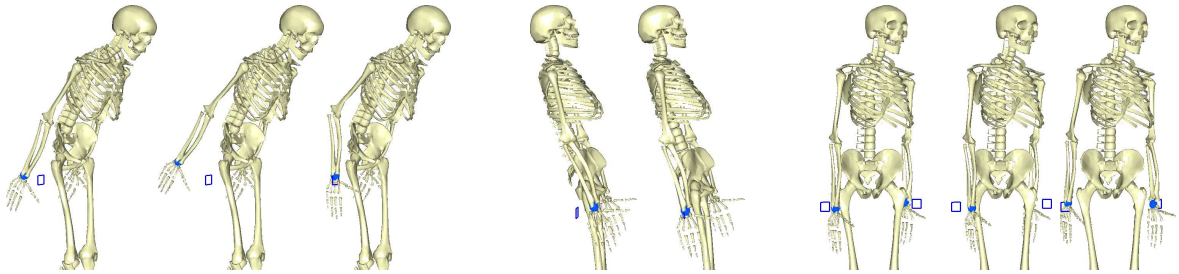


Figure 1: Constraints for keeping wrist pinned when spine is rotated. Left three images: leaning forwards moving pinned wrist; the shoulder first moves the pinned wrist away from the pinned position; the elbow moves the pinned wrist back to its pinned position. Middle two images: Moving the pinned wrist back to its position when leaning backwards. Right three images: twisting moving wrist from pinned position; the shoulder first moves the pinned wrist away from the pinned position; the shoulder moves the pinned wrist back to its pinned position

Constraints Rotational constraints, where the joint's rotation about an axis is restricted within joint angle limits, are addressed [MF98] [BT97]. Badler *et al.* [BMW87] addressed positional constraints by describing how to position an articulated figure with a weighting strategy. However, joint angle limits and rotational constraints are not considered.

3 OUR APPROACH

Inverse Kinematics Method

In our system we use the task-priority algorithm with damped least-squares. We implemented the recursive algorithm, including linear equality and inequality constraints that are satisfied after each iteration step, as discussed in [BB98].

Dealing with Positional Constraints

To efficiently check whether a joint caused a pinned joint to move from its position, each joint J_i is assigned a chain K_i $(i = 1, \ldots, n)$ that is used in the pinning algorithm (see Table 1). A chain K_i is a part of the hierarchy of the articulated figure that contains the joint J_i. For a hierarchy similar to the H-Anim specification [HAWG], the chain K_i usually starts at the joint J_i and proceeds down the hierarchy to the leave nodes. For instance, the chain corresponding to the right shoulder joint would start at the shoulder, and proceed via elbow and wrist to all finger joints of the right hand. When checking whether a joint J_i caused another pinned joint to move, only joints in the chain K_i are tested. This provides the flexibility to allow a joint to be pinned for rotations by a specified chain, but not pinned for rotations by the other joints outside the chain.

The pseudo-code of our algorithm for handling pinned joints is listed in Table 1.

```
1.  for each simulation step
2.    changed = FALSE
3.    compute q̇ , q
4.    ∀ joints i = 1, ..., n
5.      θ_curr = current rotation of J_i
6.      θ_new = θ_curr + q̇_i
7.      ∀ joints j in chain K_i
8.        if (J_j is pinned)
9.          pos_curr = current position of J_j
10.         set rotation of J_i to θ_new
11.         pos_new = current position of J_j
12.         if (pos_curr ≠ pos_new)
13.           ∀ joints p = i + 1, ..., n
14.             φ_curr = current rotation of joint J_p
15.             φ_new = φ_curr + q̇_p
16.             set rotation of J_p to φ_new
17.         pos_new = current position of J_j
18.         if (pos_curr ≠ pos_new)
19.           set rotation of J_i to θ_curr
20.           changed = TRUE
21.         ∀ joints t = i + 1, ..., n
22.         set rotation of J_t to q_t
23.       set rotation of J_i to θ_curr
24.     if (NOT changed)
25.       set rotation of J_i to θ_new
```

Table 1: Our algorithm.

4 ADDITIONAL CONSTRAINTS

In this section, we discuss special constraints that can be used for the arm of a human-like articulated figure when positional constraints are added to the IK problem.

Reachable Space

In many applications where a reaching task is applied, it should be tested whether the goal is within the reachable space of the hand, to determine whether the spine should remain fixed or should be allowed to rotate in order to obtain a natural pose. When the shoulder position is fixed, the reachable space can be roughly approximated by a half-sphere [Zha96]. However, when there is a positional or rotational constraint set for the elbow, the method discussed in [Zha96] has to be extended to deal with these cases.

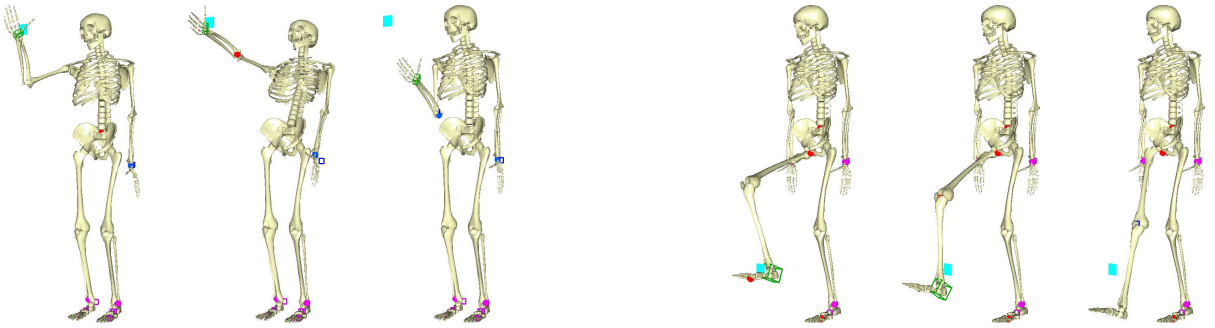


Figure 2: Different types of constraints affect the motion towards a target position. Left three images: the goal (indicated by the cyan square) is reached if no constraints are imposed on the right arm; the skeleton compensates for a rotational constraint of the right elbow by leaning backwards; the goal cannot be reached if the right elbow is pinned in space (left to right). Right three images: the right ankle reaches the goal without any constraints; a fixed rotation of the right knee prevents the ankle from fully reaching the goal; a pinned right knee makes it impossible for the ankle to get close to the goal (left to right).

If the elbow is not pinned in terms of position, the origin of the sphere is the shoulder position, the radius of the sphere is the arm length and the x-, y- and z-axis of the sphere is approximately the shoulder joint base frame. When the elbow is pinned in terms of position, the origin of the sphere is the elbow position, the radius of the sphere is the lower arm length and the x-, y- and z-axis of the sphere is approximately the elbow joint base frame.

To determine if the goal position is within the reachable space of the hand, the goal position is transformed into spherical coordinates.

Let θ define the azimuthal angle in the xz-plane and ϕ the polar angle from the y-axis. Let r be the distance from the goal position to the origin (radius). Let ℓ_L define the length of the lowerarm, ℓ_U the length of the upperarm and d the distance between the hand and the shoulder. Then,

$$\begin{aligned} r &= \sqrt{x^2+y^2+z^2}\,, \\ \theta &= \tan^{-1}\frac{x}{z}\,, \\ \phi &= \cos^{-1}\frac{y}{r} \end{aligned}$$

with

$$-\frac{\pi}{4} \leq \theta \leq \frac{3\pi}{4}\,, \tag{1}$$

$$-\frac{\pi}{2} \leq \phi \leq \frac{\pi}{2}\,, \tag{2}$$

$$0 \leq r \leq \ell_U + \ell_L\,. \tag{3}$$

If the elbow is pinned to a position in space, Equation (3) should be changed to:

$$0 \leq r \leq \ell_L\,. \tag{4}$$

If the elbow has a rotational constraint, and the arm is outstreched, Equation (3) should be changed to:

$$r = \ell_U + \ell_L\,. \tag{5}$$

If the elbow has a rotational constraint, and the arm is not outstreched, Equation (3) should be changed to:

$$r = d. \tag{6}$$

θ and ϕ should satisfy Equations (1) and (2), respectively, and r should satisfy either one of Equation (3), (4), (5) or (6), according to the constraints set for the elbow. If these equations are satisfied, the goal position is within the reachable space of the hand and the rotation of the vertebrae is fixed. Otherwise the goal position is not within the reachable space of the hand and the rotation of the vertebrae cannot be fixed.

Spine Rotation

Bending forwards or backwards Assume the wrist is pinned and the spine is rotated, bending the character forward. In Figure 3 let ℓ_L be the distance between the shoulder and wrist, $\mathbf{x}$ the distance between the current and pinned wrist position and $\mathbf{d}$ the distance between the shoulder and pinned wrist position. Angle B can then be calculated making use of the *Law of Cosines*:

$$\begin{aligned} x^2 &= \ell_L^2 + d^2 - 2\ell_L d\cos B \\ \Rightarrow \quad B &= \cos^{-1}\left(\frac{\ell_L^2 + d^2 - x^2}{2\ell_L d}\right)\,. \end{aligned} \tag{7}$$

The shoulder's rotation around the x-axis is set in such a way that it moves the wrist away from the pinned position. Now the rotation of the elbow that is necessary to move the wrist back to its pinned position, is calculated. In Figure 3 let ℓ_L be the length of the lowerarm, $\mathbf{x}$ the distance between the current and pinned wrist position and $\mathbf{d}$ the distance between the elbow and pinned wrist position. Angle B can now be calculated as discussed above.When the spine rotates, bending the character backwards, only the rotation for the shoulder should be calculated. This is illustrated in Figure 1.

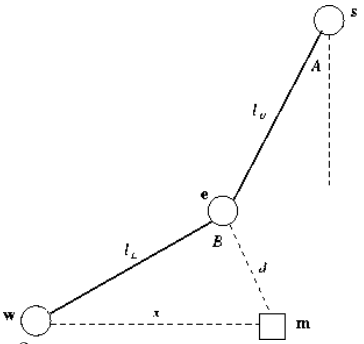


Figure 3: Triangle that is formed between the shoulder, elbow, and wrist.

Twisting Assume the wrist is pinned and the spine is rotated, resulting in a twist. Then, in Figure 3 let ℓ_L be the distance between the shoulder and wrist, $\mathbf{x}$ the distance between the current and pinned wrist position and $\mathbf{d}$ the distance between the shoulder and pinned wrist position. The rotation of the shoulder is calculated as discussed above. The rotation of the shoulder around the y-axis is set such that the wrist is moved towards its pinned position. Then the rotation of the shoulder around the x-axis is calculated in the same way, moving the wrist towards its pinned position. This is illustrated in Figure 1.

5 RESULTS

Rotating spine When the spine is rotated and the wrist is pinned, the rotation will move the wrist from its pinned position, as can be seen from the first pose from each group in Figure 1. Then by adding the constraints discussed in Section 4, the wrist is moved back to its pinned position.

Reachable space On the left of Figure 2 the task is to move the right wrist upwards towards the goal position. When there are no constraints on the arm, the goal position is within the reachable space (discussed in Section 4) of the hand and therefore the vertebrae do not rotate. When the elbow is fixed, the goal position is not within the reachable space of the hand (the distance to the goal position is not equal to the length of the arm) and therefore the rotation of the vertebrae are not fixed. To compensate for the rotational constraint of the elbow, the character leans backwards to reach the goal. When the elbow is pinned to a position in space, the goal position is further away than the length of the lowerarm and therefore not within the reachable space of the hand. In an attempt to get as close as possible to the goal position, the skeleton turns towards the goal, but the goal cannot be reached.

The full version of the paper with a detailed discussion, as well as a comparison between our method and the weighting strategy, can be found at: http://www.mpi-sb.mpg.de/resources/VirtualHumans/publ/wscg2005.pdf

6 FUTURE WORK

In future we want to apply the algorithm to motions such as walking and jumping, e.g. walking up or down stairs, where one foot needs to stay at a position while stepping downwards or upwards. Another possible application could be a ballet dancer that needs to keep the hand on the bar while doing the ballet movements and where in many movements one foot has to stay pinned at a certain position. We also want to extend the algorithm to include the possibility to pin a joint to a relative position in space, e.g. that the hands stay on the back of the dancing partner while they are dancing, i.e. moving in space.

REFERENCES

[BB98] P. Baerlocher and R Boulic. Task-priority formulations for the kinematics control of highly redundant articulated structures. In *IEEE IROS '98*, pages 323–329, 1998.

[BK03] S.R. Buss and J-S. Kim. Inverse kinematics with selectively damped least squares, July 2003.

[BMW87] N.I. Badler, K.H. Manoochehri, and G. Walter. Articulated figure positioning by multiple constraints. *IEEE Computer Graphics & Applications*, 7(6):28–38, June 1987.

[BT97] R. Boulic and D. Thalmann. Interactive identification of the center of mass reachable space for an articulated manipulator. In *Proceedings of International Conference of Advanced Robotics (ICAR)*, pages 589–594, July 1997.

[HAWG] Web3D Consortium (H-ANIM) Humanoid Animation Working Group. H-Anim: Specification for a standard VRML Humanoid, version 1.1. http://www.h-anim.org/Specifications/H-Anim1.1.

[MF98] N. Madhavapeddy and S. Ferguson. Specialised constraints for an inverse kinematics animation system applied to articulated figures. In *Proc. Eurographics UK '98)*, 1998.

[MK85] A.A. Maciejewski and C.A. Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *International Journal of Robotics Research*, 4(3):109–117, 1985.

[Wel93] C. Welman. Inverse kinematics and geometric constraints for articulated figure manipulation. Master's thesis, Simon Fraser University, 1993.

[Whi69] D.E. Whitney. Resolved motion rate control of manipulators and human protheses. *IEEE Transactions on Man-Machine Systems*, 10(2):47–53, 1969.

[Zha96] X. Zhao. Kinematic control of human postures for task simulation. Technical Report IRCS Report 96–32, University of Pennsylvania, December 1996.

Line Correspondences Between Two Images Using Local Affine Moment Invariant

C.H. Chan, Y.S. Hung and C.H. Leung
Department of Electrical and Electronic Engineering
The University of Hong Kong, Pokfulam Road, Hong Kong, P.R.China
{chchan, yshung, chleung}@eee.hku.hk

ABSTRACT

This paper proposes an algorithm for matching line segments between two images which are related by affine transformations using local affine moment invariant (AMI). Instead of using traditional methods for objection recognition in which each object is **globally** represented by a vector of affine moment invariants, here each pair of line segments extracted from each image is **locally** represented by an affine moment invariant. This algorithm is suitable for line correspondences with multi-planes and occlusion. Matches are determined through comparing invariant values and voting. Experimental results are given for both synthetic and real images. The noise model of affine moment invariant is also presented.

Keywords

Affine transformation, Line matching, Local affine moment invariant (AMI)

1. INTRODUCTION

Image matching is one of the basic problems in computer vision. It is the process of finding features such as line segments in different images that represent the same feature of the observed scene. Many papers have been published in the past on line matching under an affine transformation. Some of them make use of geometric invariants for matching. Lamdan *et.al* [Lam90a] developed a geometric hashing technique which calculates affine invariant coordinates for arbitrary point sets under various geometric transformations. The method is however very computation intensive. There are three other commonly used affine invariants: distance ratio [Hut91a], area ratio [Cha02a], and affine moment invariant (AMI) [Flu94a]. Distance ratio and area ratio are ratios of two relative affine invariants, so they may have an ordering problem. On the other hand, AMI is defined over an area and is invariant to the starting position. Besides, AMI is more robust to noise. Figure 1 shows the average error (in %) of the invariants against the noise level (in *SNR*) on 1000 arbitrary quadrangles. The upper one is area ratio, the middle one is distance ratio and the bottom one is AMI. It can be seen that AMI can tolerate a larger range of noise.

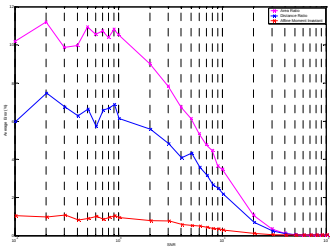


Figure 1. Robustness of different invariants to noise.

All methods which use AMIs, to the best of our knowledge, are mainly for pattern matching or object recognition. They are computed over a set of data points or a closed-boundary region (usually an object), so they may encounter the problem of occlusion. Distance ratio and area ratio have been proposed by some authors for localized line matching [Gro95a], but AMIs have not been used for the same purpose. This paper presents the key idea to match line segments between two affine transformed images using local AMI. The noise model of AMI is also presented.

2. ALGORITHM

2.1 Selection of line segments

This algorithm assumes that both images contain line segments that do not intersect except at their end points only. In the case that there are intersecting lines, they are broken up so that finally there are only

WSCG 2005 SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

two kinds of line configurations: connected lines (3 noncollinear point pairs) and disjoint lines (4 noncollinear point pairs). Since connected lines provide just enough information to determine the affine transformation, even if the point pairs are mismatched, we can still find a transformation which can exactly relate the point pairs. Hence, such configuration does not help line matching at all. On the other hand, for disjoint lines, besides 3 basic point pairs, there is one extra point pair which enforces an extra constraint for the determination of an affine transformation. There is no exact solution except when all 4 pairs are really related by an affine transformation. Since such configuration has a discriminatory power, this algorithm only considers line matching based on the affine moment invariant calculated from such configuration.

2.2 Selection of starting position

Given a pair of disjoint lines in both images, the end point correspondences are not known, so there are 4! = 24 combinations of end point correspondence. To reduce the number of combinations, the convex hull constraints are added. Hartley proved that projective transformation preserves the convex hull of a point set [Har93a]. In the four-point case, the convex hull may contain 3 or 4 points.

First consider the convex hull containing all 4 points, we can form a loop which passes through the end points one by one with no skipping as shown below.

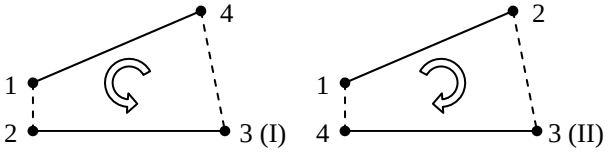


Figure 2. Convex hull containing 4 points.

If the conditions above are enforced, the number of combinations is reduced to 8 (4 clockwise and 4 anti-clockwise). If the orientation of the convex hull is chosen for all pairs of lines in both images to be in clockwise direction, the number of combinations is reduced to 4. Finally, the loop can be restricted to traverse both points of one line segment before going on to another line segment, whereby the number of combinations is reduced to 2. For example, in diagram (II) above, the only possible loops are either (1→2→3→4) or (3→4→1→2).

Now consider the convex hull containing only 3 points as shown in Figure 3. If the same rules are applied, four combinations are valid: (1→2→3→4), (2→1→3→4), (3→4→1→2) and (3→4→2→1). Note that the relative size of areas is preserved under an affine transformation, i.e., if area A is larger than area B in the first image, the affine transformed area A is still larger than the transformed area B in the transformed image. Hence, the number of loops is reduced by half if they are restricted to have a larger area. If the area enclosed by the path (1→2→3→4) or (3→4→1→2) is larger than that enclosed by the path (3→4→2→1) or (2→1→3→4), then paths (1→2→3→4) and (3→4→1→2) are selected.

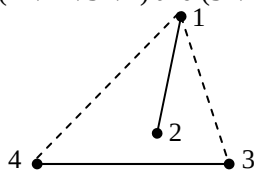


Figure 3. Convex hull containing 3 points.

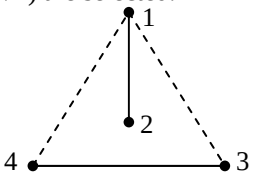


Figure 4. Symmetric line pattern.

In the case that the line pattern is symmetric as shown in Figure 4, there is an ambiguity in choosing the paths as the same area is enclosed by 4 different paths. However, the shape enclosed by the path (1→2→3→4) or (3→4→1→2) is just the reflection of the shape enclosed by the other two paths, and AMI is invariant to an affine transformation including reflection. Therefore, the AMIs for both areas are the same and can be used for matching.

2.3 Affine moment invariant

A moment invariant is a moment-based descriptor of a planar shape, and is very useful for pattern recognition. J. Flusser and T. Suk extended the idea to AMI, which is invariant under an affine transformation [Flu93a]. The moment $m(p,q)$ of order $(p + q)$ of a binary 2-D object G is defined as:

$$m(p,q) = \iint_G x^p y^q dx\, dy \quad \ldots (1)$$

while the central moment $\mu(p,q)$ of order $(p + q)$ is defined as:

$$\mu(p,q) = \iint_G (x - x_t)^p (y - y_t)^q dx\, dy \ldots (2)$$

where $x_t = m_{10}/m_{00}$ and $y_t = m_{01}/m_{00}$ are the coordinates of the center of gravity of object G. The second-order AMI is defined as [Flu94a]:

$$AMI = \frac{1}{\mu(0,0)^4}(\mu(2,0)\mu(0,2) - \mu(1,1)^2) \quad \ldots (3)$$

It is a time consuming task to compute the double integrals of equations (1) and (2). Our algorithm is based on the method described in [Sin93a]. In order to use AMI for line matching, we need to derive the variance of AMI when the line segments are perturbed by noise. Assume we are given 4 ordered points (x_i, y_i), $i = 0,\ldots,3$, which form the object G. Let $(p_1,\ldots,p_4,p_5,\ldots,p_8) = (x_0,\ldots,x_3,y_0,\ldots,y_3)$, and AMI = $f(p_1,\ldots,p_8)$, each variable p_i can be expressed as $p_i = \bar{p}_i + \delta_i$, where $\bar{p}_i$ is the true but unknown value of p_i , and δ_i is a random perturbation (noise) added to the true variable $\bar{p}_i$. Assume δ_i is independent and identically distributed with zero mean and standard deviation σ_δ. Then AMI = $f(\bar{p}_i + \delta_i; i = 1,...,8)$. Expanding AMI in Taylor series and neglect second and higher order terms:

$$AMI = f(\bar{p}_1,...,\bar{p}_8) + \sum_{i=1}^{8}\left[\delta_i \frac{\partial}{\partial \bar{p}_i} f(\bar{p}_1,...,\bar{p}_8)\right] \ldots(4)$$

Let $\bar{P} = (\bar{p}_1,...,\bar{p}_8)$, $\quad E[AMI] = f(\bar{P}) \ldots(5)$

$$\sigma_{AMI}^2 = E[(AMI - E(AMI))^2] = \sigma_\delta^2 \sum_{i=1}^{8}\left(\frac{\partial}{\partial \bar{p}_i} f(\bar{P})\right)^2 \ldots(6)$$

Let $g(\bar{P}) = \dfrac{m(2,0)m(0,2) - m(1,1)^2}{m(0,0)^4} \ldots(7)$,

From(3), $f(\bar{P}) = g(\bar{x}_i - \bar{x}_t, \bar{y}_i - \bar{y}_t; i = 0,...,3)$, where $(\bar{x}_t, \bar{y}_t)$ are the coordinates of the center of gravity of object G without noise. Let $X_i = \bar{x}_i - \bar{x}_t$; $Y_i = \bar{y}_i - \bar{y}_t$, each term in (6) can be computed as follows:

$$\frac{\partial f}{\partial \bar{x}_i} = \frac{\partial g(X_i, Y_i; i = 0,...,3)}{\partial X_i} \ldots(8); \quad \frac{\partial f}{\partial \bar{y}_i} = \frac{\partial g(X_i, Y_i; i = 0,...,3)}{\partial Y_i} \ldots(9)$$

$$\frac{\partial g}{\partial X_i} = \frac{1}{m(0,0)^4}\left[m(2,0)\frac{\partial m(0,2)}{\partial X_i} + m(0,2)\frac{\partial m(2,0)}{\partial X_i} - 2m(1,1)\frac{\partial m(1,1)}{\partial X_i}\right] - 4\frac{m(2,0)m(0,2) - m(1,1)^2}{m(0,0)^5}\frac{\partial m(0,0)}{\partial X_i}$$

$$\frac{\partial g}{\partial Y_i} = \frac{1}{m(0,0)^4}\left[m(2,0)\frac{\partial m(0,2)}{\partial Y_i} + m(0,2)\frac{\partial m(2,0)}{\partial Y_i} - 2m(1,1)\frac{\partial m(1,1)}{\partial Y_i}\right] - 4\frac{m(2,0)m(0,2) - m(1,1)^2}{m(0,0)^5}\frac{\partial m(0,0)}{\partial Y_i}$$

$$\frac{\partial m(0,2)}{\partial X_i} = \frac{1}{12}\left[Y_{i+1}\left(Y_i^2 + Y_i Y_{i+1} + Y_{i+1}^2\right) - Y_{i-1}\left(Y_{i-1}^2 + Y_{i-1}Y_i + Y_i^2\right)\right]$$

$$\frac{\partial m(2,0)}{\partial X_i} = \frac{1}{12}\left[(X_{i-1} + X_i)^2(Y_i - Y_{i-1}) + (X_i + X_{i+1})^2(Y_{i+1} - Y_i) + 2X_i^2(Y_{i+1} - Y_{i-1})\right]$$

$$\frac{\partial m(1,1)}{\partial X_i} = \frac{1}{12}\left[Y_{i+1}^2(X_i + X_{i+1}) + Y_i^2(X_{i-1} - X_{i+1}) - Y_{i-1}^2(X_{i-1} + X_i) - 2X_i Y_i(Y_{i-1} - Y_{i+1})\right]$$

$$\frac{\partial m(0,0)}{\partial X_i} = \frac{1}{2}[Y_{i+1} - Y_{i-1}]; \qquad \frac{\partial m(0,0)}{\partial Y_i} = \frac{1}{2}[X_{i-1} - X_{i+1}]$$

$$\frac{\partial m(0,2)}{\partial Y_i} = \frac{1}{12}\left[(Y_{i-1} + Y_i)^2(X_{i-1} - X_i) + (Y_i + Y_{i+1})^2(X_i - X_{i+1}) + 2Y_i^2(X_{i-1} - X_{i+1})\right]$$

$$\frac{\partial m(2,0)}{\partial Y_i} = \frac{1}{12}\left[X_{i-1}\left(X_{i-1}^2 + X_{i-1}X_i + X_i^2\right) - X_{i+1}\left(X_i^2 + X_i X_{i+1} + X_{i+1}^2\right)\right]$$

$$\frac{\partial m(1,1)}{\partial Y_i} = \frac{1}{12}\left[X_{i-1}^2(Y_{i-1} + Y_i) - X_i^2(Y_{i-1} - Y_{i+1}) - X_{i+1}^2(Y_i + Y_{i+1}) + 2X_i Y_i(X_{i-1} - X_{i+1})\right]$$

2.4 Procedure

This subsection describes the whole procedure of line matching in details. Given *M* and *N* line segments extracted from two images which are approximately related by unknown affine transformations, the matching procedure consists of the following steps:

2.4.1 Step 1

Construct connectivity tables of size *M*×*M* and *N*×*N* for the first and second images respectively. These tables show if segments are connected to the starting or end points of other line segments in the image.

2.4.2 Step 2

Construct affine moment invariant tables of size *M*×*M* and *N*×*N* for the first and second images respectively. In the calculation of the affine moment invariant between two segments, a path through the line segments is chosen such that the line configuration fulfills the requirement described in section 2.2. Besides, given the standard deviation of noise σ_δ, the range for each affine moment invariant (AMI $\pm$ σ_{AMI}) is computed. In the case that two line segments are found to be connected from the previous step, they will not be considered in the following steps.

2.4.3 Step 3

Each affine moment invariant in the moment invariant table for the first image is compared with all invariants of the second image. If the affine moment invariant between two line segments in the second image is within the range of the invariant (defined by the standard deviation of noise σ_δ) between two line segments in the first image, the pairs of segments are regarded as a putative match which is then verified as follows. Since they are arranged in the configuration described in section 2.2, each pair of line segments would only have 2 possible paths. Therefore, by calculating the two affine transformations and projecting the two line segments in the first image into the second image, the average projection error for each transformation is found. If the smaller of the two errors is less than a predefined threshold, then the pairs of line segments are regarded as matched with known segment correspondences.

Next, form a voting table of size *M*×*N*. A vote is added to each of the (*i*,*k*) and (*j*,*l*) cells of the vote table if the segment pair (*i*,*j*) in the first image is matched to (*k*,*l*) in the second image. This table accumulates the number of votes for line correspondences between the two images. The higher the number of votes, the higher the possibility that the corresponding line segments are matched.

Besides, form an ordering table of size *M*×*M* and *N*×*N* for the first and second images. As described above, each line segment may change its vertex order such that it fulfills the requirement described in section 2.2. The vertex ordering information is stored in the ordering tables. The (*i*,*j*) entry states whether the line segment *i* needs to change its vertex order when pairing up with the line segment *j*.

The ordering information is entered into the ordering tables only when the pair of line segments has found a match during comparison, i.e. the projection error is less than the predefined threshold.

2.4.4 Step 4

Given the vote table formed from Step 3, the final matches between the two images are determined. A winner-takes-all approach is adopted.

2.4.5 Step 5

After a set of line matches are found from Step 4, the end point correspondences are performed. For each matched line segment, by examining the ordering information of all other matched line segments which have been paired with it, we can deduce the correct sense of direction of the line segment in a

correspondence.

2.5 Improving line matching

There are other finer data available to improve the validity of computed matches and possibly to find new ones. One such information is the connectivity. If two line segments are connected in the first image, the corresponding line segments in the second image should also be connected. Moreover, the sense of direction of the segments relative to the connected vertex should be consistent.

3. EXPERIMENTAL RESULTS

In this section, the algorithm is tested using 2 sets of images. The first set is a pair of images of a planar object as shown in Figure 5. Both images contain 68 line segments with noise.

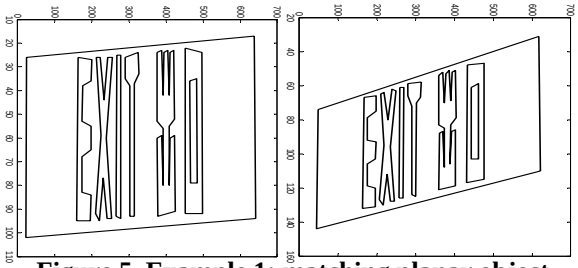


Figure 5. Example 1: matching planar object.

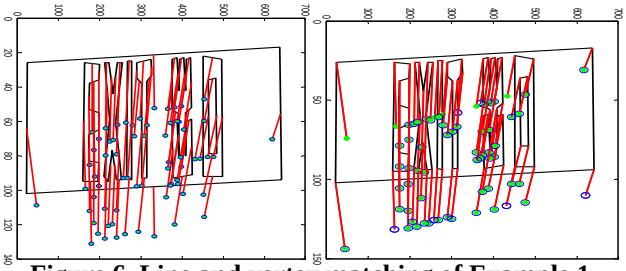


Figure 6. Line and vertex matching of Example 1.

Figure 6 shows the matching results. The left figure shows the vector flows between the mid-points of matched line segments superimposed on the first image. Each flow is a match and the dot represents the mid-point of the corresponding line segment in the second image. In this example, all line matches are correct. The right figure shows the vector flows of vertex matching superimposed on the first image. As each line segment has two vertices, the first vertex is represented by a solid dot while the second vertex is represented by a circle. In this example, all vertices are correctly matched.

Figure 7 shows a pair of images with multi-planes. The edges are approximated by noisy line segments in the two images that may not correspond. The first image contains 122 line segments and the second has 93. Figure 8 shows the matching results. 91 line segments (182 vertices) are matched, with 13 (resp. 26 vertices) mismatches. The main reason for the mismatches is that the series of windows at the bottom of the images in Figure 7 are very similar and errors often occur in such situations.

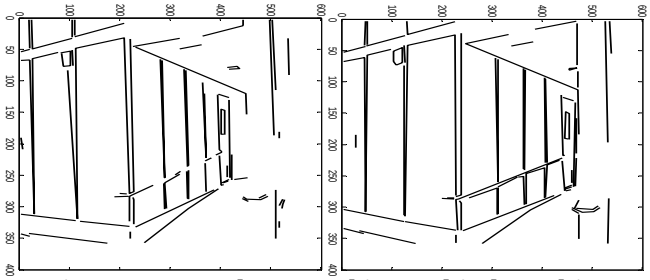


Figure 7. Example 2: matching multi-plane objects.

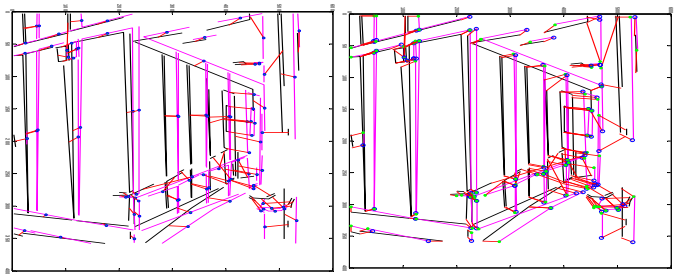


Figure 8. Line and vertex matching of Example 2.

4. CONCLUSION

The algorithm for line matching between two images related by affine transformations is developed, which is based on the matching of local second-order affine moment invariants. Experimental results are presented which show the applicability to multi-planes line matching with noise.

5. ACKNOWLEDGMENTS

The work described in this paper was supported by a grant from the Research Grants Council of Hong Kong SAR, China (Project No.HKU7058/02E) and partially by CRCG of the University of Hong Kong.

6. REFERENCES

[Cha02a] H. B. Chan and Y. S. Hung, “Matching patterns of line segments by eigenvector decomposition”, SSIAI’02, pp. 286-289, 2002.

[Flu93a] J. Flusser and T. Suk, “Pattern recognition by affine moment invariants”, Pattern Recognition, vol. 26, pp. 167-174, 1993.

[Flu94a] J. Flusser and T. Suk, “A moment-based approach to registration of images with affine geometric distortion”, IEEE Transactions on Geoscience and Remote Sensing, vol. 32, no. 2, March, 1994.

[Gro95a] P. Gros, O. Bournez, and E. Boyer, “Using geometric quasi-invariants to match and model images of line segments”, research report INRIA, No2608, July 1995.

[Har93a] R. Hartley, “Cheirality invariants”, In Proc. of DARPA Image Understanding Workshop, Washington, D. C., pp. 745-753, 1993.

[Hut91a] D. P. Huttenlocher, “Fast affine point matching: An output-sensitive method”, CVPR91, pp. 263-268.

[Lam90a] Y. Lamdan, J. Schwartz, and H. Wolfson, “Affine invariant model-based object recognition”, IEEE Trans. Robotics and Automation, vol. 6, No. 5, pp. 578-589, 1990.

[Sin93a] Mark H. Singer, “A general approach to moment calculation for polygons and line segments”, Pattern Recognition, vol. 26, No. 7, pp. 1019-1028, 1993.

Acceleration of Genetic Algorithm with Parallel Processing with Application in Medical Image Registration

B. Laksanapanai* W. Withayachumnankul * C. Pintavirooj * P.Tosranon**
*Department of Electronics, Faculty of Engineering
King Mongkut's Institute of Technology Ladkrabang, Thailand.
**Department of Physical apply and Medical instrumentation
King Mongkut's Institute of Technology North Bangkok, Thailand.

kpchucha@kmitl.ac.th

ABSTRACT

Generally, image registration using genetic algorithm is a time-consuming process since the algorithm needs to evaluate the objective function several hundred times depending on the vastness of search space. The situation appears worse if the registration is intensity-based due to the interpolation loops prior to each objective function. However, with the availability of parallel processing method, one can accelerate the application of genetic algorithm for iterative-based image registration process of up 80 % for multi-modality alignment

Keywords

image registration, genetic algorithm, parallel processing

1. INTRODUCTION

Image registration is essential in many medical tasks. It provides useful information for diagnosis, surgical planning, event tracking, radiotherapy, and so on. The key of image registration is to find the proper transformation of one image to another image so that each point of one image is spatially aligned with its corresponding point of the other. The intrinsic registration is more preferable since it needs no extra marker adhered to the patient while he/she is exposed by the imaging equipment. The intrinsic registration methods are divided into 3 types including landmark-based, segmentation-based, and intensity-based methods. Standing apart from other methods, intensity-based methods use full content available from the images since they deal directly with grey-level information but not with extracted or intrinsic feature. These methods, however, suffers from long computational time of full-plane grayscale transformation leading to limitation of usage. Several intensity-based methods are available including, for example, the maximizing mutual information [Col90a], correlation coefficients [Jun90a], or minimization of squared intensity differences. For more details about these methods and medical image registration, the reader should consult [Ant98a].

To find the optimum of transformation, the genetic algorithm (GA) [Gol89a] is chosen because of its strong immunity to local minima, flexibility to multidimensional function, and simplistic implementation procedure. Several image registration techniques use GA as a parameter-search-for procedure, but with intensity-based registration the GA is rarely found because the repetitive call to the objective function together with the computational cost of the transformation makes the time of convergence crucial. [Raj99a] are examples from the minority of literatures that use GA searching for parameters from distance function of grayscale images.

Conference proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

The time consuming and the complexity of the image registration process are the critical problems. In addition, the registration process requires very high performance of the computer. Therefore, in this paper, concepts of parallel programming method is employed to speed up a medical image registration process. In the parallel processing, the appropriate amount of work is distributed to each computer (node) in the clustering system. The processing time is then diminished as a function of number of dedicated computer.

The paper is organized as follows: - The second topic presents the basic idea of genetic algorithm. The third topic, objective function, describes the parameters used to transform the image and the correlation coefficient used to measured the similarity between registered images. The forth topic proposed the concept of parallel programming. The demonstrations are done with unregistered images.The paper is finalized with discussion and conclusion.

2. BASIC IDEA OF GENETIC ALGORITHM

GA mimics all the processes based on the concept of natural evolution to find the optimized solution to the given problem residing in the search space. The GA pool contains a number of individuals called chromosomes. Each chromosome encoded from the parameters holds the potential solution. According to the evolutionary theories, the chromosomes which only have a good fitness are likely to survive and to generate the offsprings and pass its strength to them by the genetic operator. The fitness of chromosome is the way that is linked to the predefined problem or objective function. Figure 1. shows the possible stages of evolution.

GA cycle can be decomposed into five steps described as follows:-

1.) Randomly initialize the population in the pool. With more population, the coverage in search space is good but traded off by the calculation time in each generation.

In the simplest way, the real-value parameter is binary-coded to give a bit string. The bit strings for several parameters are concatenated to form a single string or chromosome. In accord with the biology, each bit corresponds to gene.

2.) Evaluate the chromosomes by objective function. After the evaluation, all the chromosomes are ranked for the fitness values in the descending or ascending order depending on the purpose of objective function.

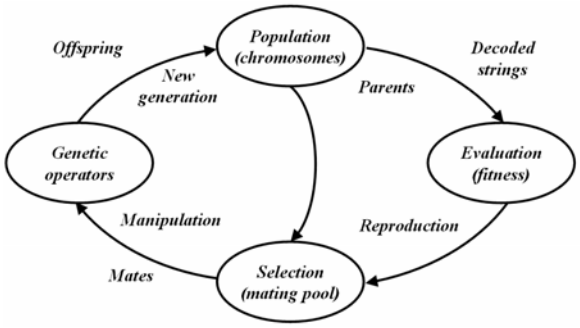


Figure 1. GA cycle.

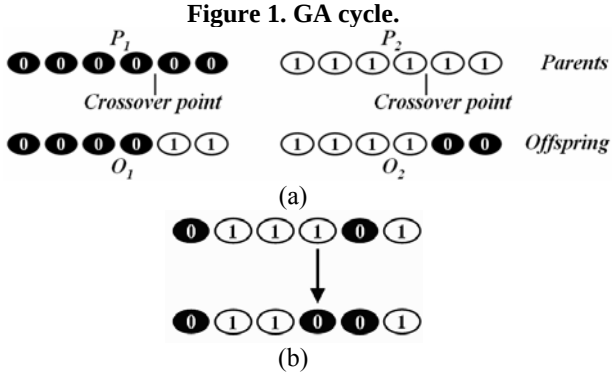


Figure 2. Genetic operators, a) crossover and b) mutation.

3.) Select the parents from the chromosomes with the biased chances. The higher-fitness chromosome is prone to survive.

4.) Generate the offspring using genetic operators consisting of crossover and mutation. Crossover is a recombination operator that swaps the parts of two parents as shown in figure 2a. Two random decisions are made prior to this operation, whether to do it or not and where the crossover point is. Mutation gives a good chance to explore the uncovered search space. It mutates, or complements some genes in the chromosome of the offspring, so that the new parameter value takes place.

5.) Entirely replace the elder generation in the pool with the newer one and return to step 2. In some case, the few best elders may be kept away from replacement. This is known as elitist strategy.

The criteria for stopping the revaluation loops are met when a) the loop number is over some predefined point or b) the steady state lasts for predetermined times.

3. OBJECTIVE FUNCTION

The image transformation is defined by the following equation:

$$X_i = s_1(M(\vec{t}_i)) \quad (1)$$

$$Y_i = s_2(\vec{t}_i)\,,\ i=1,...,N \quad (2)$$

where $\vec{t}_i$ denotes the spatial location, $s_1(t)$ and $s_2(t)$ denote two original images, $X_i=(X_1,...,X_N)$ and $Y_i = (Y_1,...,Y_N)$ denote the transformed and normal image, $s_1(M(\cdot))$ denotes a spatial transformation and interpolation of $s_1(\cdot)$ Generally, a similarity model is sufficient to regulate the unregistered images especially the tomographic-scanned images because they have no perspective distortion. Therefore, the transformation model M can be formed by multiplication of scaling S, rotation R, and translation T matrices in the order that there is no non-orthogonal scaling.

$$M = S \cdot R \cdot T = \begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ dx & dy & 0 & 1 \end{bmatrix} \quad (3)$$

Noted that z-concerned parameters are discarded for non-perspective 2D registration, and the matrices are valid for row vector.

The validation of registration is measured by the correlation coefficient between two aligned images. The correlation coefficient method is most likely able to measure similarity of multimodal images with the simplicity. Given vectorized image X and Y, the correlation coefficient ρ is defined as follows

$$\hat{\rho}_s(X,Y) = \frac{\hat{C}_s(X,Y)}{\sqrt{\hat{\sigma}_x^2 \hat{\sigma}_y^2}}, \; -1 \le \hat{\rho}_s(X,Y) \le 1 \quad (4)$$

where covariance $\hat{C}_s(X,Y)$, variances $\hat{\sigma}_x^2$, $\hat{\sigma}_y^2$, and means $\overline{X}$, $\overline{Y}$ are defined by

$$\hat{C}_s(X,Y) \equiv \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \overline{X})(Y_i - \overline{Y}) \quad (5)$$

$$\hat{\sigma}_x^2 \equiv \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \overline{X})^2 \quad \hat{\sigma}_y^2 \equiv \frac{1}{N-1}\sum_{i=1}^{N}(Y_i - \overline{Y})^2 \quad (6)$$

$$\overline{X} \equiv \frac{1}{N}\sum_{i=1}^{N} X_i \qquad \overline{Y} \equiv \frac{1}{N}\sum_{i=1}^{N} Y_i \quad (7)$$

Hence, the correlation coefficient method can be used as an objective function which has to be optimized to maximum value.

4. PARALLEL PROCESSING

The computational expense of genetic algorithm and the vast memory requirement of intensity-based image registration have motivated the development of parallel implementation on multi-computers. In general, parallel implementations can be grouped into 3 categories: 1.) Hardware architectures designed specially for parallel processing 2.) Software implementations on machines with hardware support for parallel processing [Kau88a] and 3.) Parallel processing algorithms implemented entirely in software on general-purpose hardware[Pet99a, Pot89a]. This research falls into the third category. It includes the clustering architecture [Buy99a] (shown in figure 1) consisting of a homogeneous collection of general purpose computer systems connected via networks, also termed a clustered computing environment, provides very powerful and cost effective image registation. The parallel-implementation platform uses a public domain software Massage Passing Interface (MPI), which is easy-to-use and freely available.

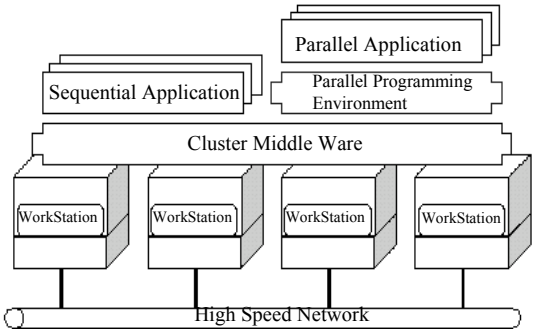


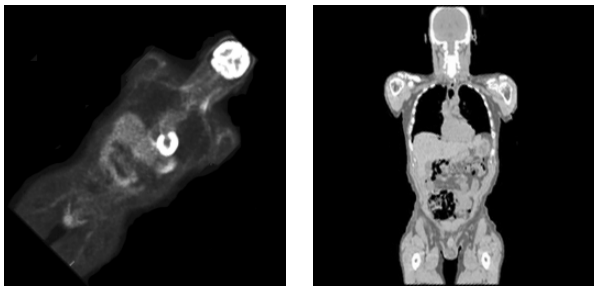
Figure 3. Clustering system architecture developed by using parallel programming environment such as MPICH.

5. EXPERIMENTAL RESULTS

We test the purposed system for multimodality image. The searched-for parameters consist of: $-0.8 \le sx, sy \le 1.31$, $-\pi \le \theta \le \pi$, and $-127 \le dx,dy \le 128$. String length for each parameter is equal to 8, or the step of quantization is equal to 256. The population size in each generation is restricted to 150. With the crossover probability of 0.6 and mutation probability of 0.06, GA cycle always meets the criterion of 50-time steady-state within 300 generations.

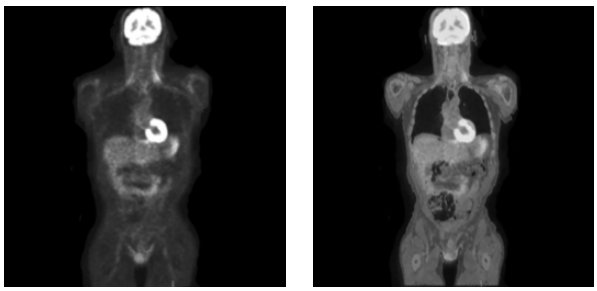
All the experiments in this paper are tested on homogeneous system consists of 5 machines of Intel Xeon 2.4GMHz Dual CPU, ECCRAM 1 Gbytes connected via 1 Gbps LAN running Linux operating system. The software is written on C++ using parallel programming environment such as MPICH.

The PET and CT image to be aligned are shown in figure 4. With the size of 256 × 256 pixels × 8 bits,



(a) **(b)**

Figure 4. Unregistered (a) PET and (b) CT images



(a) **(b)**

Figure 5.(a) Transformed PET image,(b) PET-CT fusion

process	Multimodal	
	PET-CT 256 x 256	
	linear	nearest
1	0.59844	0.422585
2	0.309701	0.217344
3	0.217391	0.147382
4	0.165579	0.115491
5	0.13389	0.092646

Table 1. Averaged time per Genetic Cycle

the PET image is rotated by 45 degrees to see if there are some transformation irregularities during GA cycle. After certain point, the PET image is transformed to the correct position resulting in growth of coefficient from 0.300463 to 0.753600. The parameters obtained from GA are $\theta = 46.9$, $sx = 1.056$, $sy = 0.97$, $dx = -3$,$dy = -3$. Figure 5 shows the aligned position of PET and simple PET-CT fusion image that gives both anatomical and functional details. The times per generation are recorded in Table 1.

Table 1 shows the average time per genetic cycle for multimodality image registration. In multimodality registration, the result of nearest and linear interpolations are compared. One can be inferred that system speed-up factor increases as the number of node increases. Specifically, four nodes can accelerate the registration task more than one node does about 75%.

6. DISCUSSION AND CONCLUSIONS

This paper proposes the new method for intensity-based image registration process on clustering system that faster than compute on single machine and single memory. There are two contributions of this paper. The first contribution is the application of genetic algorithm for intensity-based image registration. The second contribution is the application of parallel programming method to distribute works to be processed concurrently on each computer in the clustering system. The result for multi-modality image registration is very promising.

7. REFERENCES

[Ant98a] Antoine Maintz, J.B. and Viergever, M.A., "A Survey of Medical Image Registration," Medical Image Analysis, vol. 2, pp. 1-37, 1998.

[Buy99a] Buyya R. "High Performance Cluster Computing: Architectures and Systems", Volume 1. Prentice Hall PTR. 1999.

[Col90a] Collignon, A., Maes, F., Delaere, D., Vandermeulen, D., Suetens, P.,and Marchal, G., "Automated multimodality imageregistration using information theory," Bizais, Y., and Barillot, C., (eds), Information Processing in Medical Imaging, Kluwer Academic Publishers, Dordrecht, pp. 263–274,1995.

[Gol89a] Goldberg, D.E., Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, 1989.

[Jun90a] Junck, L. , Moen, J.G., Hutchins, G.D. , Brown, M.B. and Kuhl, D.E. , "Correlation methods for the centering, rotation, and alignment of functional brain images," Journal of Nuclear Medicine, 31, 1220–1276., 1990.

[Kau88a] Kaufman, A. and Bakalash, R., " Memory and Processing Architectures for 3D Voxel-Based Imagery", IEEE Computer Graphics and Applications., vol. 8, no. 11, pp. 10-23, 1988.

[Pet99a] Petersen, J., " Introduction to Programming on Distributed Memory Multiprocessor", Computer Physics Communications, vol. 73, no. 1-3, pp.72-92, 1999.

[Pot89a] Potmesil, M. and Hoffert, E. M., " The Pixel Machine: A parallel Image Computer", Comput. Graphics, vol. 23, no. 3, pp. 69-78, 1989.

[Raj99a] Rajapakse, J.C., and Guojun, B., "Functional MR Image Registration Using a Genetic Algorithm," Proceeding of Int. Conf. on Neural Information Processing, pp. 922-927, 1999.

3D Modeling from Multiple Projections: Parallel-Beam to Helical Cone-Beam Trajectory

W.Narkbuakaew C.Pintavirooj W.Withayachumnankul*
M.Sangworasil** S.Taertulakarn***
Department of Electronics
* Department of Information Technology
Faculty of Engineering and Research Center for Communications and Information Technology
** Faculty of Engineering and Computer Research and Service Center
King Mongkut's Institute of Technology Ladkrabang, Thailand
kpchucha@kmitl.ac.th
*** Faculty of Allied Health Sciences, Thammasat University, Pathumtani, 12121.
somchatt@yahoo.com

ABSTRACT

Tomographic imaging is a technique for exploration of a cross-section of an inspected object without destruction. Normally, the input data, known as the projections, are gathered by repeatedly radiating coherent waveform through the object in a number of viewpoints, and receiving by an array of corresponding detector in the opposite position. In this research, as a replacement of radiographs, the series of photographs taken around the opaque object under the ambient light is completely served as the projections. The purposed technique can be adopted with various beam geometry including parallel-beam, cone-beam and spiral cone-beam geometry. From the process of tomography, the outcome is the stack of pseudo cross-sectional image. Not the internal of cross section is authentic, but the edge or contour is valid.

Keywords: Image Reconstruction, 3D Rendering, Photographic Tomography, Helical Cone-Beam Tomography

1. INTRODUCTION

Shape extraction is the first step of many 3D applications, including a 3D modelling, object recognition, robot navigation, machine inspection, geometry measurement, and so on. To satisfy these applications, it requires an appropriate shape extraction method. Nowadays, several shape extraction systems are proposed, and each one is suitable in the limited range of applications. Some of those, including stereo disparity [Bar82a-Dho89a], laser range finder [Bos98a-Oka98a], structured light [Sat94a-Boy14a], shape from shading [Hor89a-Pen93a], optical flow [Hor80-Bro97], etc.

In this research, we investigated the process for various beam geometry. In the simple case, the object is placed a long distance. In this case, each scan line on the images taken from different projection is served as the projection data for parallel-beam image reconstruction algorithm.

The parallel-beam geometry is not applicable due to the long focal-object distance (FOD). Cone-beam geometry can be used in the case of small FOD. However, when the long object is applied, in order to cover the whole part of object, the wide-angle lens must be used. This results in distortion of the resulting 3D model. To avoid the problem, the spiral cone-beam tomography is purposed. With the small pitch distance and small field of view on the camera, the trajectory of light from the object to the camera resembles a parallel ray and hence lessen the distortion problem.

This work is organized as the following: - Section 2 discusses theory involved in tomography in various beam geometry. Shape extraction process is provided in section 3. Section 4 explains the acquisition system. The results are provided in section 5. Discussions and conclusions are given in section 6.

Conference proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. MODIFIED HELICAL CONE-BEAM RECONSTRUCTION ALGORITHM

The cone-beam process [Fel84a] can be modified to cope with long object. By changing path of camera to helix and using helical cone-beam reconstruction algorithm [Wan93a], the speed of acquisition and the resolution of results for long object are improved. Since we can simultaneously change the azimuthal angle and the z-level of camera while taking the projections, the time used to complete all projections is reduced.

And since we can place the camera close to the object, the details in the projections are enlarged. Mathematical model of the modified helical cone-beam reconstruction algorithm is given by [Wan93a]

$$f(x,y,z)=\frac{1}{2}\int_0^{2\pi}\frac{D_{SO}^{\ 2}}{(D_{SO}-s)^2}R_\beta\left(\frac{D_{SO}t}{D_{SO}-s},\frac{D_{SO}\tilde{z}}{D_{SO}-s}\right)d\beta \qquad (2.1)$$

where $f(x, y, z)$ is the volumetric data, $R_\beta()$ is the two-dimensional silhouetted projection, β is the rotational angle, D_{SO} is the cone distance, (t, s) are the coordinates which are rotated from (x, y) by β, and $\tilde{z}$ is given by

$$\tilde{z}=z-\frac{h_p\beta}{2\pi} \qquad (2.2)$$

where h_p is the pitch of the helix turn.

3. SHAPE EXTRACTION PROCESS

The entire process for the shape extraction from a series of photographs is shown as a diagram in Figure 1. This process starts with capturing a number of images around the inspected object, which will be elaborated in section of imaging unit. As stated before, the method concerns only the shape or outline of the object; therefore, the segmentation procedure, the thresholding for high-contrasting image or the blue-screen technique for the others, is brought up to turn the captured images into binary format where

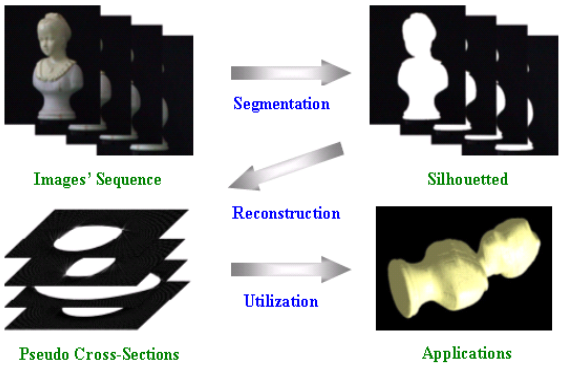


Figure 1. The process diagram of the photographic tomography

the background is equal to zero and the foreground is equal to one. As a result, this kind of shape extraction algorithm can extract the shape of any object regardless of the type of surface either the Lambertian or non-Lambertian surface. When the binary images or the silhouetted projections are passed into the modified reconstruction algorithm described previously, the stack of enhanced pseudo cross-sections whose shape is comparable to its original is attainable. This stack of cross sections can be utilized in such applications as a 3D shape modeling, geometric measurement, and so on.

4. ACQUISITIONING SYSTEM

In order to collect a series of photographs from the inspected object, it is suggested to rotate the object about the origin i.e. its axis and take the photograph using a fixed camera rather than to move the camera around the vertex path. This idea is substantial for the object which is motionless and small to medium-sized, the common characters of most of the objects. By adhering to this idea, the imaging unit can be assembled simply from a few components, and less in complexity.

The imaging unit shown in Figure 2 is mainly composed of a digital camera, a controllable rotating platform and the translatable platform. The object is placed on the rotating platform and the camera is placed translatable platform which is on the dual rails. Controlled by a computer, the rotating platform is capable of rotating precisely to any specified angle, while the digital camera is capable of moving to the specified z coordinate. For cone-beam tomography, the camera is stationary while the object is rotated. In the case of spiral cone-beam tomography, the camera is translated when the object is rotated.

For simply extracting the silhouette from the photograph, the object is placed in front of the background contrasting from the color of object, and illuminated by an ambient light to avoid specula artifacts.

5. RESULTS

When only the surface of the object is needed regardless of an internal structure, it is advised to use the surface rendering techniques, the most prominent technique of which is the marching cube [Lor87a]. This technique uses the divide-and-conquer strategy to complete the surface; it breaks up the volumetric data into a number of small cubes and matches each cube with the pre-calculated surface pattern. Hence, it delivers the continuous surface made up of an enormous number of linked triangles. However, the marching cube only illustrates the first-order interpolated frontier of the object. The rendering technique together with the Gourad [Gou71a] and

Phong [Pho75a] shading and illumination can closely resemble the real surface properties which response to the specula and ambient lights.

The surface of the object is created by segmenting the reconstructed data, calculating normal vectors for every polygon from gray levels of the same information, and then rendering using technique described above. Regarding to the conditions verified in the preceding chapter, the volumetric data of shown in Figure. 3 to Figure. 5 along with their solid models rendered in the distinctive viewpoints (cube size of the marching cube equals to one). It is noted assorted objects, whose surfaces are either Lambertian or non-Lambertian type, are reconstructed from their corresponding sequences of photographs. All of the investigated objects are that some objects use the number of projections twice as many as the specific value to make the gradient smoother.

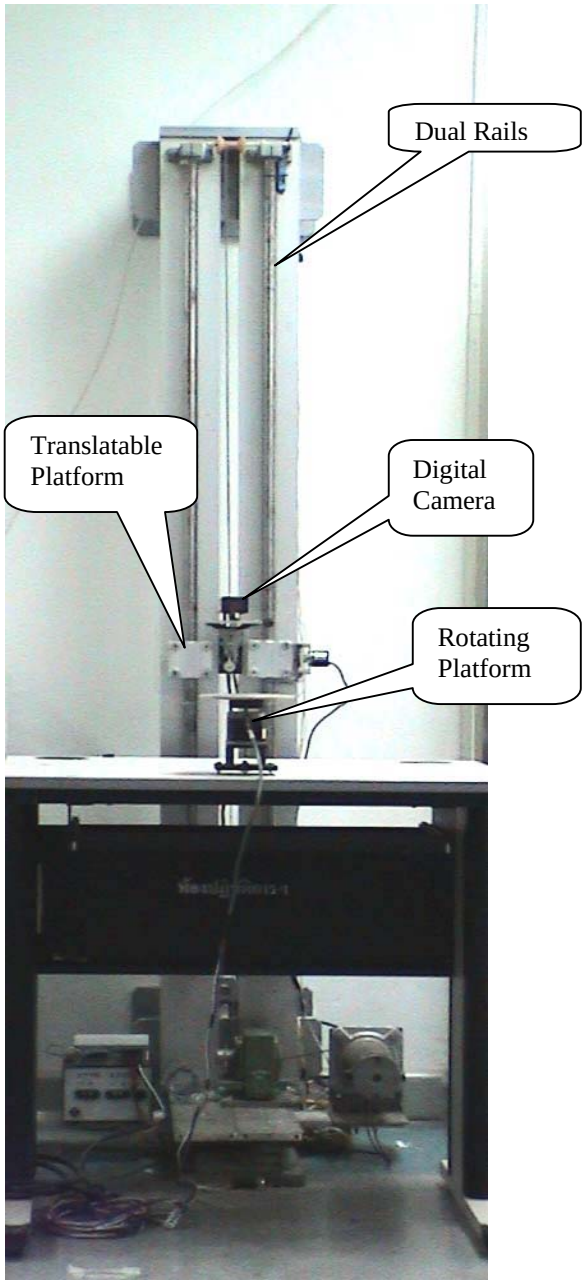


Figure 2. The prototype of an imaging unit for acquiring the sequence of photographs consisting of digital camera, rotating platform and translatable platform placed on dual rails.

Form the results, the process achieves successfully on the simple-shaped objects such as the woodcarving. Nevertheless, some information is missing in the high-detailed object caused by the totally concave problem. The ceramic cup is the best examples of this problem;

6. CONCLUSION AND DISCUSSION

The 3D shape extraction using photographic tomography is studied extensively. The main idea of the method is that the tomographic imaging is used to reconstruct the stack of pseudo cross-sectional images from a series of photographs taken around the object. The shape of the stack can closely resemble the shape of original object. Compared with other 3D shape extraction techniques, the equipment used to collect the data for this technique is much lesser in complexity. It mainly consists of the rotating platform and the digital camera which is locally available and affordable. Various beam geometry is also investigated in this paper. Parallel-beam geometry can be used when the focal-to-object distance is large. Conversely, cone-beam geometry can be used when the focal-to-object distance is small. Spiral cone beam is considered when the long object is modelled or when the perspective distortion is meaningful. Despite some drawbacks, the proposed technique is tested successfully to generate the shape of a variety of objects.

(a) **(b)**

Figure 3. Ceramic cup, (a) actual model and (b) rendered model (Az.70,El.15,Tw.10) (Cone-Beam: D_{SO} = 50 cm)

(a) **(b)**

Figure 4. Vase, (a) actual model and (b) rendered model (Helical Cone-Beam: 40 Prj/360°, number of turn = 3 D_{SO} = 47 cm and Pitch size = 256 pixel)

(a) **(b)**

Figure 5. Spray bottle, (a) actual model and (b) rendered model (Helical Cone-Beam: 40 Prj/360°, number of turn = 3, D_{SO} = 47 cm and Pitch size = 256 pixel)

7. REFERENCE

[Ali81a] Alitschuler, M. D., " The Numerical Stereo Camera", *Proc. Soc. Photo-Opt. Eng. Conf. 2-D Math. Perception*, SPIE, WA, vol. 283. 1981.

[Bar82a]Barnard, S. T. and Fischler, M. A., " Computational Stereo", *ACM Computing Surveys,* vol. 14, no. 4, pp. 553-572, 1982.

[Bic92a] Bichsel, M. and Pentland, A. P. "A Simple Algorithm for Shape from Shading", *CVPR Conference on Computer Vision and Pattern Recognition*, Champaign, Illinois, June 15-18, pp. 459-463, 1992.

[Beh86a] Beheim, G. and Frisch, K., " Range Finding using Frequency-modulated Laser Diode," *Appl. Opt.,* vol. 25, pp. 1439-1442, 1986.

[Boy87a] Boyer, K. L. and Kak, A. C., " Color-coded Structured Light gor Rapid active ranging, *IEEE. Trans. Pattern Anal. Machine Intell.,* vol. 9, no.1, pp. 14-28, 1987.

[Bro97a] Brooks, M., Chojnacki, W., van den Hengel, A., and Baumela, L.," 3D Reconstruction from Optical Flow Generated by an Uncalibrated Camera Undergoing Unknown Motion," *Proc. International Workshop on Image Analysis and Information Fusion*, Adelaide, November 1997, pp. 35-42.

[Dho89a] Dhond, U. R. and Aggarwal, J., " Structure from Stereo- A Review," *IEEE Trans. On Syst. Man. and Cybern*., vol. 19, no. 6, pp. 1489-1510, 1989.

[Dho89b] Dhond U. R., Aggarwal J. "Structure from Stereo - A Review." *IEEE Trans. on Syst. Man. and Cybern*. vol. 19, pp. 1489-1510, 1989.

[Fel84a] Feldkamp, L. A., Davis L., Kress K., "Practical Cone-Beam Algorithm," *Jounal of the Optical Society of America,*vol.1,Jun.1984.pp 612-619

[Ger99a] Gering, D.T., Wells, W.M., III. "Object modeling using tomography and photography". *Multi-View Modeling and Analysis of Visual Scenes*, (MVIEW '99) Proceedings. IEEE Workshop on. pp. 11-18, 1999.

[Gor71a] Gouraud H. "Continuous Shading of the Curved Surface," *IEEE Trans. Computer,* vol.20, no.6, 1971.pp.623-629

[Hor80a] Horn, B. K. P., and Schunck, B. G., *Determining optical flow*. AI Memo 572. Massachusetts Institute of Technology, 1980.

[Hor89a] Horn, B. and Brooks, M. (eds.). *Shape from Shading*. MIT Press, 1989.

[Hor93a] B.K.P. Horn et. al. "Impossible Shaded Images." *IEEE Trans. Pattern Analysis and Machine Intelligence*. vol. 15, no. pp. 166-170, 1993.

[Hu89a] Hu. G. and Stockman, G., " 3-D Surface Solution using Structured Light and Constraint Propagation", *IEEE. Trans. Pattern Anal. Machine Intell.,* vol. 11, no.4, pp. 390-402, 1989.

[Joh99a] Johnson A.J. et. al. "Three-Dimensional Surface Reconstruction of Optical Lambertian Objects Using Cone-Beam Tomography." *Proc. Int. Conf. on Image Processing*. vol. 2, pp. 663 -667, 1999.

[Kak88a] Kak A.C., Slaney M. *Princicles of Computerized Tomographic Imaging*. New York : IEEE Press, 1988.

[Kra93a] Krattenthaker, W., Mayer, K. J. and Duwe, H. P., " 3D-Surface Measurement with Coded Light Approach", *Proc. Öesterr. Arbeitsgem. MusterKennung*, vol. 12, pp. 103-1114, 1993.

[Lic98a] Lichtenbelt B.,Crane R. and Naqvi S. "Introduction to Volume Rendering (Hewlett-Packard Professional Books)." Practice Hall PTR., 1998

[Lor87a] Lorensen W.E. and Cline H.E. "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *Computer Graphics (Proc.SIGGRAPH),*vol.21, no.3, Jul.1987.pp 163-169

[Mea70a] Meadows D. M. et. al. "Generation of Surface Contours by Moiré Pattern." *Appl. Opt*. vol. 9, pp. 942-947, 1970.

SHAPE MATCHING USING SET OF CURVE GEOMETRIC INVARIANT POINT

C. Pintavirooj*, P. Nantivatana***, P. Putjarupong*,
W. Withayachumnankul**, M. Sangworasil*
*Department of Electronics
**Department of Information Engineering,
Faculty of Engineering and Research Center for Communication and Information Technology (ReCCIT)
King Mongkut's Institute of Technology Ladkrabang, Thailand.
***Department of Electrical Engineering, Faculty of Engineering, Sripatum University, Thailand.
kpchucha@kmitl.ac.th

ABSTRACT

We introduce a non-iterative geometric-based method for shape matching using a novel set of geometric landmarks residing on a 2D contours. These landmarks are intrinsic and are computed from the differential geometry of the curve. We exploit the invariant properties of geometric landmarks that are local and preserved under the affine and some perspective transformation. Geometric invariant exploits coplanar five-point invariant and ration of area constructed from a sequence of consecutive landmarks. These invariants are preserved not only in affine map but weak perspective map as well. To reduce the sensitivity of the landmarks to noise, we use a B-Spline surface representation that smoothes out the curve prior to the computation of the landmarks. The matching is achieved by establishing correspondences between the landmarks after a conformal sorting based on derived absolute invariant and registering the contours. The experiments have shown that the purposed methods are robust and promising even in the presence of noise.

Keywords Geometric Invariant, Affine Invariant, B-Spline

1. INTRODUCTION

Shape matching is a central problem in visual information system, computer vision, pattern recognition image registration, and robotics. Application of shape matching includes image retrieval, industrial inspection, stereo vision, and fingerprint matching. The term shape is referred to the invariant geometrical properties of the relative distance among a set of static spatial features of an object. These static spatial features are known as shape features of the object. After extracting the shape features for a model and a scene, a similarity may be used to compare the shape features. The similarity measure is referred to as a shape measure.

Conference proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

The shape measure should be invariant under certain class of geometric transformation of the object. In the simple scenario, shape measures are invariant to translation rotation and scale. In this case, the shape measures are invariant under similarity transformation. When included the invariance of shape measures to shear effect, the shape measures are said to be invariant under affine transformation. Finally in the complicated case, shape measures are invariant under perspective transformation when included the effect caused by perspective projection.

There are many techniques available to shape matching which can be classified mainly into two main categories; a global method and a local method. The global method works on an object as a whole; while the local method on a partially visible object or occlusion. The well-known global method includes Wavelet transform [Job95a, Wan97a], Moment-based approach [Hu62a], Fourier descriptors [Bri68a, Zah72a, Ott92a] and Median Axis Transformation [Blu67a, Blu78b, Pel81a]; while the well-known local method exploits intrinsic properties of the shape which includes geometric invariant [Bes88b, Gov99a, Mun92a], curvature extrema point [Bol82a, Mil89a, Chi89a, Rao94a], zero-curvature points

[Ali98a], line intersections [Sto82a], and [Kan81a], centroid of closed-boundary region [Gos86a], knot points [Coh98a], etc. In this paper study local geometric intrinsic features from which we derive a set of fiducial points which are related to curve derivative. Our shape matching is based on image registration. We use the set of fiducial points as a landmark. We then establish the correspondence between two sets of fiducially points by exploiting a local geometric projective (affine and weak perspective) invariants which are extracted from the five-point coplanar [Mun92a] spanned by the landmark points. Once the corresponding landmark is found, linear transformation parameter is estimated and the shape are aligned.

This paper is organized as follows. Section 2 introduces the local geometric curve features used as landmarks on the curve. The registration process is thoroughly discussed in section 3. Experimental results are shown in section 4. Discussion and conclusion are presented in section 5.

2. INTRINSIC GEOMETRIC FEATURE OF CURVE

Let r(t) =[x(t), y(t)], where t is a parameter, represent a shape (or curve) C in Cartesian coordinate system. The Inflection points are the point at which their curvatures are zero, i.e. The points at which

$$k(t) = \frac{r^{(1)}(t) \times r^{(2)}(t)}{| r^{(1)}(t) |^3} = \frac{\dot{x}(t)\ddot{y}(t) - \ddot{x}(t)\dot{y}(t)}{(\dot{x}(t) + \dot{y}(t)^2)^{3/2}} = 0$$

or the points at which

$$r^{(1)}(t) \times r^{(2)}(t) = \dot{x}(t)\ddot{y}(t) - \ddot{x}(t)\dot{y}(t) = 0 \qquad (2.1)$$

As a results, $r^{(1)}(t)$ and $r^{(2)}(t)$ are parallel at the inflection point.

Inflection points on the affine transformed curve are the points at which

$$k_a(t) = \frac{r_a^{(1)}(t) \times r_a^{(2)}(t)}{| r_a^{(1)}(t) |^3} = \frac{\dot{u}(t)\ddot{v}(t) - \ddot{u}(t)\dot{v}(t)}{(\dot{u}(t) + \dot{v}(t)^2)^{3/2}} = 0$$

or the points at which

$$\dot{u}(t)\ddot{v}(t) - \ddot{u}(t)\dot{v}(t) = \det[T]\left(\dot{x}(t)\ddot{y}(t) - \ddot{x}(t)\dot{y}(t)\right) = 0$$
$$\dot{x}(t)\ddot{y}(t) - \ddot{x}(t)\dot{y}(t) = 0 \qquad (2.2)$$

where [T] is the transformation matrix. Equation (2.2) is the same as (2.1). As a consequence, at the inflection points, the curvature or affine curvature is zero and $r^{(1)}(t)$ and $r^{(2)}(t)$ are parallel. Since the affine map preserve parallelism, we have shown that $r_a^{(1)}(t)$ and $r_a^{(2)}(t)$ are also parallel. Therefore the inflection points of the affine transformed curve are the transformed inflection points of the original curve and hence are affine invariant. Inflections have been widely used as the candidates for curve matching [Wal98a].

If we take derivative of (2.1), we have

$$\dot{x}(t)\dddot{y}(t) + \ddot{x}(t)\ddot{y}(t) - \ddot{x}(t)\ddot{y}(t) - \dddot{x}(t)\dot{y}(t) =$$
$$\dot{x}(t)\dddot{y}(t) - \dddot{x}(t)\dot{y}(t) = r^{(1)}(t) \times r^{(3)}(t) = 0 \qquad (2.3)$$

which is the point at which $r^{(1)}(t)$ and $r^{(3)}(t)$ are parallel and hence affine invariant. This point is the point at which the affine curvature is maximum. We call this point the maximum affine curvature point. Compared with zero affine curvature points, the maximum affine curvature points are more robust to noise. Moreover, threshold of affine curvature can be set such that only the maximum affine curvature point of which its affine curvature exceeding the threshold is selected. As a result, maximum affine curvature points caused by local disturbance are excluded.

The same concept can be applied to derive a set of derivative of curve. Table 1 shows a set of geometric feature. The maximum order of derivative of curve used in the computation of relative affine invariant is limited at two. In such case, we have 6 features; three of which associate with points of minimum relative affine invariant and the other three features associated with points of maximum relative affine.

Relative affine Invariant	Feature related to the zero relative affine Invariant	Feature related to the maximum of Relative affine Invariant
$k_1(t) =$ $r(t) \times r^{(1)}(t)$	$f_1(t) \equiv$ $k_1(t) = 0$ $r(t) \times r^{(1)}(t) = 0$	$f_2(t) \equiv$ $k_1^{(1)}(t) = 0$ $r(t) \times r^{(2)}(t) = 0$
$k_2(t) =$ $r(t) \times r^{(2)}(t)$	$f_3(t) \equiv$ $k_2(t) = 0$ $r(t) \times r^{(2)}(t) = 0$ same as $f_2(t)$	$f_4(t) \equiv$ $k_2^{(1)}(t) = 0$ $[r(t) \times r^{(3)}(t)] +$ $[r^{(1)}(t) \times r^{(2}(t)] = 0$
$k_3(t) =$ $r^{(1)}(t) \times r^{(2)}(t)$	$f_5(t) \equiv$ $k_3(t) = 0$ $r^{(1)}(t) \times r^{(2)}(t) = 0$	$f_6(t) \equiv$ $k_3^{(1)}(t) = 0$ $r^{(1)}(t) \times r^{(3)}(t) = 0$

Table 1. Set of Geometric Feature of Curve

3. CONSTRUCT ABSOLUTE INVARIANT

Geometric invariants are shape descriptors that remain unchanged under geometric transformations such as perspective and affine transformation that will be used mainly as object recognition and/or

image registration. A new approach of obtaining the invariants has been developed and used in the curve matching and recognition. A feature point explained in section 3 which is the local and invariant intrinsic property of the curve can be uniquely identified both before and after transformation. The identified feature points then are used to compute the geometric invariants which serve as the curve signature for the curve matching and recognition. The well-known perspective invariant is five-point coplanar.

Any five nonlinear points in the plane, namely $P_1,...,P_5$ can also form perspective invariant-the so-called five-point coplanar [Eam91a] with their image, $P_1',...P_5'$,

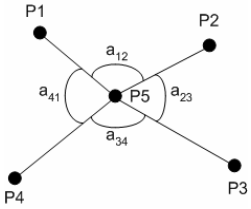


Figure 1. Five-point Coplanar Invaraint

$$\frac{|m'_{431}||m'_{521}|}{|m'_{421}||m'_{531}|} = \frac{|m_{431}||m_{521}|}{|m_{421}||m_{531}|} \qquad (3.1)$$

where $m_{ijk} = (P_i\ P_j\ P_k)$ with $P_i = (x_i\ y_i\ 1)^t$, $m'_{ijk} = (P_i'\ P_j'\ P_k')$ with $P_i' = (x_i'\ y_i'\ 1)^t$ and $|m|$ is the determinant of m.

Since the invariant relationship in equation (3.1) holds under a perspective transformation, a set of perspective invariant can be constructed by considering 4 consecutive maximum-curvature points and a centroid which is also preserved under an affine and perspective transformation. For a curve with n maximum-curvature points, there are n set of geometric invariants denoted as $I(k)$ for $k = 1,2,...,n$

In the absence of noise and/or occlusion, each of $I_a(j)$ in the affine-transformed curve (3.1) will have a counter part *I(i)* on the original curve with $I_a(j) = I(j)$, with that counterpart easily determined through a circular shift involving n comparison where n is the number of invariant. In the presence of noise, occlusion and non-linear transformation, we allow smaller error percentage between counterpart invariant. We adopted a run length method to decide on the correspondence between the two ordered set of maximum curvature point. For every starting point on the transformed, this run length method computes a sequence of consecutive invariant that satisfies $|I(i) - I_a(j)| < 0.05|I(i)|$ and declares a match based on the longest string. Once this correspondence is found, these matched landmarks are used to estimate the transformation matrix

4. EXPERIMENT

To demonstrate the capabilities of the shape matching exploiting a set of geometric invariance curve, we applied the algorithm for shape classification thirteen contours of fish which have the various patterns. Each contour has approximately 400 points of coordinates. The third order B-Spline with control points of 30 is employed to fit the contours of fish. To find the fish which matches with the 0^{th} fish, a feature point f1 is determined. Figure 2 shows the results of shape alignment

Figure 2. The alignment of contour of fish o^{th} (dark contour) against thirteen contours of fish (light contour) using feature f1 and five-point coplanar invariance

5. DISCUSSION AND CONCLUSION

This paper we explore the geometric invariant points. The fact that the geometric invariant points are local and preserved under an affine transformation makes them feasible to use for matching and classification of curve. By exploiting the absolute affine and perspective invariant using the feature points, the correspondence between the feature points on the original and the transformed curve is established, the transformation parameters are computed and the transformed curve is aligned against an original.

6. REFERENCES

[Ali98a] Ali, W.S.I. and Cohen, F.S., "Registering coronal histological 2-D sections of a rat brain with coronal sections of a 3-D brain atlas using geometric curve invariants and B-spline

representation," *IEEE Trans. on Medical Imaging*,vol. 17, no. 6, pp. 957-966, 1998

[Ans89a] Ansari, N. and Delp, E.J., "Partial shape recognition: a landmark-based approach," *IEEE International Conference on Systems, Man and Cybernetics*, pp. 831-836, 1989

[Bar91a] E.B. Barrett and P. Payton, "General Methods for Determining Projective Invariants in Imagery", *CVGIP: Image Understanding*, Vol. 53, No. 1, pp. 46-65, Jan. 1991.

[Bes89a] Besl, P. J. and Jain, R. C., "3D Object Reject Recognition," *Comput. Serveys*, vol. 17, no. 1, Mar. 1985.

[Bes88b] Besl, P. J., "Geometric Modeling and Computer Vision," Proc. IEEE, vol. 76, pp. 936-958, Aug. 1988.

[Blu67a] Blum, H. "A Transformation for Extracting New Descriptors of Shape," In Whaten-Dunn, editor, Models for the Perception of Speech and Visual Forms, pp. 362-380, MIT Press, 1967.

[Blu78b] Blum, H. and Nagel, R., "Shape Descriptor using Weighted Symmetric Axis Features," Pattern Recognition, vol. 10, pp. 167-180, 1978.

[Bol82a] Bolles, R. and Cain, R., "Recognizing and Locating Partially Visible Objects, The Local-Feature Focus Method," Int'l J. Robotics. Res., Vol. 1, No. 3, pp. 57-82, 1982.

[Bri68a] Brill, E. L., "Character Recognition via Fourier Descriptors." WESCON Convention Record, Paper 25/3 Los Angeles, 1968.

[Car76a] Do Carmo, M. P. 1976. Differential geometry of curves and surfaces, Prentice hall, Englewood Cliffs, NJ.

[Chi89a] Chien, C.-H.; Aggarwal, J.K, "Model construction and shape recognition from occluding contours," *IEEE Trans. Patt. Anal. Machine Intell*., vol. 11, no. 4, pp. 372-389, 1989.

[Coh98a] Cohen, F. S., Yang, Z., Huang, Z. and Nissanov, J., "Computer Matching of Histological Rat Brain Sections," *IEEE Trans. on Biomedical Engineering* 45, 5, pp. 642-649, 1998.

[De72a] De Boor, C., "On Calculation with B-splines," *J. Approx. Theory*, vol. 6, pp. 50-62, 1972.

[De78b] De Boor, C., *A Practical Guide to Splines*, New York, Springer-Verlag, 1978.

[Do76a] Do Carmo, M. P. 1976. *Differential geometry of curves and surfaces*, Prentice hall, Englewood Cliffs, NJ.

[Gos86a] Goshtasby, A., "Piecewise Linear Mapping Functions for Image Registration," *Patt. recog*., 19, 6, pp. 459-466,1986

[Gov99a] Govindu, V. and Shekhar, C., " Alignment using Distributions of Local Geometric Properties," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 21, no. 3, pp.1031-1043, 1999.

[Job95a] Jacobs, C., Finkelstein, A. and Salesin, D., "Fast Multiresolution Image Query," *Computer Graphic Proceeding, SIGGARPH*, pp. 277-286, 1995.

[Kan81a] Kanal, L. N., Lambird, B. A., Lavine, D. and Stockman, G. C., " Digital Registration of Images from Similar and Dissimilar Sensors," In *Proceedings of the International Conference on Cyberbetics and Society*, pp. 347-351,1981.

[Kim89a] Kimia, B., " Conservation Laws and Theory of Shape," Ph. D. Thesis, McGill Univ., 1989.

[Mil89a] Milios, E. E., "Shape Matching using Curvature Processes," Comput. Vision, Graphics Image Process., vol. 47, pp. 203-226, 1989.

[Mun92a]J. Mundy and A. Zisserman. Appendix --- projective geometry for machine vision. In J. Mundy and A. Zisserman, editors, *Geometric invariances in computer vision*. MIT Press, Cambridge, 1992.

[Ott92a] Otterloo, P. J., " A contour-Oriented Approach to Shape Analysis," Helmel Hamstead, Prentice Hall, 1992.

[Pel81a] Peleg, S. and Resenfeld, A. "A min-max Medial Axis Transformation," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 3, pp. 208-210, 1981.

[She99a] Shen, D. Wong, W. and Horace, H. S., " Affine-Invaraint Image Retrieval by Correspondence Matching of Shapes," Image and Vision Computing, vol. 17, pp. 489-499, 1999.

[Sto82a] Stockman, G.C., Kopstein, S., and Benett, S., "Matching Images to Models for Registration and Object Detection via Clustering," *IEEE Trans. Patt. Anal. Machine Intell.,* 4, pp.229-241, 1982

[Wan97a] Wang, J. Z., Wiederhold, G., Firschein, O. and Wei, S. X., "Wavelet-based Image Indexing Techniques with Partial Sketch Retrieval Capability," *Proceeding of the Fourth Forum on Research and Technology Advances in Digital Libraries*, IEEE, 1997.

[Wal98a] Walid S. Ibrahim Ali, Fernand S. Cohen "Registering Coronal Histological 2-D Sections of a Rat Brain with Coronal Sections of 3-D Brain Atlas Using Geometric Curve Invariants and B-Spline Representation" IEEE Transactions on Medical Imaging, Vol. 17, No. 6, December 1998. pp. 957-966.

Skeleton-based temporal segmentation of human activities from video sequences

Sébastien Quirion Alexandra Branzan-Albu Robert Bergevin
Computer Vision and Systems Laboratory
Dept of Electrical and Computer Engineering
Laval Univ., Ste-Foy, Qc, Canada, G1K 7P4
{squirion,branzan,bergevin}@gel.ulaval.ca

ABSTRACT

This paper presents a new multi-step, skeleton-based approach for the temporal segmentation of human activities from video sequences. Several signals are first extracted from a skeleton sequence. These signals are then segmented individually to localize their cyclic segments. Finally, all individual segmentations are merged with respect to the global set of signals. Our approach requires no prior knowledge on human activities and can use any generic stick-model. Two different techniques for signal segmentation and for the fusion of the individual segmentations are proposed and tested on a database of fifteen video sequences of variable level of complexity.

Keywords

Video Understanding, Motion Analysis, Periodicity Analysis, Signal Segmentation, Data Fusion.

1. INTRODUCTION

Human activity description and recognition is currently an active research area in computer vision. As in [Pol94a], cyclic activities are defined as regularly repeating sequences of motion events. Among others, [Pol94a] and [Bob01a] have proposed new methods for the description and recognition of such activities. These methods are based on the assumption that each video sequence contains exactly one activity. However, activity recognition systems should be able to handle sequences containing several cyclic and non-cyclic activities.

Our research aims at extending the applicability of such algorithms by extracting activities from video sequences through temporal segmentation. The proposed approach uses skeleton sequences to represent the human motion and divides the segmentation task into three sequential steps. Our algorithms are based on periodicity analysis.

Conference proceedings ISBN 80-903100-7-9
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

The use of a skeleton[1] model enables us to accurately follow the periodic motion of a human subject despite the periodic motion present in the background. Another advantage of this high-level motion description is the ability to detect an activity by using anatomical information.

Related Work

Few relevant papers have been published on the temporal segmentation of human activities. Among the few, [Yaz04a] describes a temporal segmentation method of symmetric activities based on a 2D inter-frame similarity plot and requires no prior knowledge on human activities.

2. PROPOSED APPROACH

The proposed approach is divided in three steps. In the first step, signals are *extracted* from a skeleton sequence in order to obtain simple but significant data to work with. In the second step, these signals are *segmented individually* in order to localize their cyclic segments. In the final step, all individual segmentations are coherently *merged* with respect to the entire set of signals. This approach is highly modular and allows for different implementations depending on the priorities of the task at hand.

[1] A skeleton is a generic graph where each node has spatial coordinates. These coordinates are changing over time, thus forming a skeleton sequence.

Signal Extraction

Using signals to represent significant attributes of an activity (e.g. movement of a skeleton node, variation of a joint angle, etc.) is an interesting choice as one can expect such attributes to exhibit cyclic behavior during activities. As a general rule, every node or edge of the stick-model subject to independent and significant motion (e.g. hands, feet, etc.) is to be represented by at least one signal. These signals are needed for the analysis of relative cyclic movements (e.g. waving an arm). Also, at least one global attribute, like the instantaneous speed of the centroid of all nodes, is to belong to the signal-set. This global feature is needed for the analysis of global cyclic movements (e.g. jumping up and down). The actual set of signals depends on the stick-model used.

Individual Signal Segmentation

A straightforward approach to implementing a signal segmentation algorithm is based on periodicity analysis. This approach computes a periodicity score to rate different possible segments in the signal and uses these ratings to select relevant segments. Other approaches are also possible. For instance, an effective way to reduce the complexity of the problem is to first form a rough segmentation by removing the 'silences' (i.e. portions of very low amplitude) in the signal. A more informed approach based on periodicity analysis can then be applied to the resulting segments to refine this rough segmentation.

2.1.1 Periodicity Score

The periodicity score rates the periodicity of a signal S in the interval [0,1]. The periodicity scoring algorithm used in this paper is a 1-D adaptation of the lattice matching technique presented in [Cut00a].

Considering A_S, the normalized mean-removed autocorrelation of the signal S, M_S, the ordered set of maxima found in A_S, and c_S, the estimated cycle-length (i.e. the mean length between consecutive autocorrelation maxima), the expression of the score is:

$$\Psi_S = \frac{1}{|M_S|-1}\sum_{i=2}^{|M_S|}\left(1-\frac{|c_S\cdot(i-1)-M_S(i)|}{c_S}\right)\cdot A_S(M_S(i)) \qquad (1)$$

This score is a measure of proximity in lag and value between actual maxima of A_S and expected maxima for a perfectly periodic signal of period c_S. The score of a periodic signal is equal to one and decreases as the signal becomes less and less cyclic. The score may be negative for degenerate cases.

For increased robustness, an adjusted score Ψ_S' can be computed in the same fashion using only the first 90% of the mean-filtered values of A_S. Moreover, as long cyclic segments are preferred over smaller ones, the length of the segment (i,j) can be normalized by the length l_S of the whole given signal S. To favor length only in cyclic segments, a threshold η is needed to define what is considered cyclic and what is not. The length-normalized score is computed as follows:

$$\Upsilon_{S(i:j)} = \eta^{1-\left(\frac{(j-i+1)}{l_S}\right)}\cdot\Psi_{S(i:j)}{}'^{\frac{(j-i+1)}{l_S}} \qquad (2)$$

The length-normalized score properties can be summarized in a few key statements:

- As the length of the segment (i,j) approaches l_S, $\Upsilon_{S(i:j)}$ approaches $\Psi_{S(i:j)}'$;
- As the length of the segment (i,j) approaches 0, $\Upsilon_{S(i:j)}$ approaches η ;
- $(\Psi_{S(i:j)}' < \eta) \Leftrightarrow (\Upsilon_{S(i:j)} < \eta)$;
- $(\Psi_{S(i:j)}' > \eta) \Leftrightarrow (\Upsilon_{S(i:j)} > \eta)$.

These statements imply that length improves the score of a cyclic segment (i.e. a segment (i,j) with $(\Psi_{S(i:j)}' > \eta)$) but decreases the score of a non-cyclic segment.

2.1.2 Segmentation

The proposed algorithm first detects the 'most cyclic' segment in a given signal using a ***bestSegment*** algorithm. It then validates if the segment is cyclic enough. If its periodicity score is above a threshold τ_L, it is included in the segmentation set. The algorithm then proceeds in the same fashion on the remaining portions of the signal until the analysis of the entire signal is completed. With the exception of ***bestSegment***, the remainder of the implementation is rather simple.

A first implementation of the ***bestSegment*** algorithm simply computes the length-normalized score for every possible segment (i,j) and returns the segment with the maximum score. The segmentation algorithm is called ***MaxS*** when it uses this implementation of ***bestSegment***.

A second implementation seeks the segment with the best length-normalized score through numerical optimization[2]. This approach is based on the fact that the length-normalized score increases as a segment grows within a cyclic portion of a signal, and it decreases as this segment grows out of the cyclic portion. The segmentation algorithm is called ***OptS*** when it uses this implementation of ***bestSegment***.

[2] Our implementation uses the DHC algorithm [Yur94a].

Segmentations Fusion

In this last step, the idea is to use the segments detected on each individual signal as candidates for the global segmentation. The score threshold τ_L for the signal segmentation should be high in order to minimize the number of false detections. The set of candidates therefore contains temporal segments during which there is good reason to believe an activity occurs. To create a robust segmentation, the general idea is to compute a global score for each candidate and keep the highest scoring non-overlapping subset of candidates.

In the presented implementations, the global score of a segment is the sum of the scores greater than a threshold τ_G obtained on each signal in the signal-set.

SimF is a straightforward fusion algorithm. It iteratively removes the highest scoring candidate from the initial set, adds it to the final segmentation and removes from the set of candidates any remaining candidate that overlaps the chosen candidate.

A second fusion algorithm, ***GenF***, is an alternative of the one presented above. Instead of discarding overlapping candidates entirely, ***GenF*** discards the overlapping portion of the candidates. The remainders are being considered as candidates if their periodicity score is greater than τ_L on at least one signal.

3. EXPERIMENTAL RESULTS

Fifteen test sequences were captured in front of a static background using a monocular camera at 30fps. A skeleton was then adjusted at each frame using the single-frame extraction algorithm presented in [Vig03a]. The resulting unfiltered skeleton sequences are typically noisy and represent input data for the approach. The duration of these test sequences ranges from 300 to 1200 frames. The activities in the sequences are cyclic, articulated motions such as arm waving, side-stepping, etc. Nine signals were extracted from each skeleton sequence to form the corresponding signal-set: four angle signals (shoulders and hips), four vertical position signals (hands and feet) and the instantaneous speed of the centroid of all nodes.

The test sequences are partitioned according to their complexity. The sequences of type A are sequences where all the activities are temporally bounded by pauses. The sequences of type B are sequences where at least one activity is temporally adjacent to another activity or to non-cyclic movements. Finally, the sequences of type C are sequences where at least one activity fuses with another, like waving an arm immediately followed by waving the two arms.

The two fusers (***SimF*** and ***GenF***) were first confronted using ***MaxS*** in both cases. The most promising fuser was then used to compare the two segmentation algorithms.

In all performed tests, $\tau_L = 0.8$ was used as the lower limit on the periodicity score for signal segmentation, $\tau_G = 0.5$ as the lower limit on the periodicity score for global score computing and $\eta = 0.5$ as the lower limit on the periodicity score for length-normalized score computation.

The result tables provide two different validation measures. The first measure indicates the number of activities in the reference segmentation that were correctly detected[3] by the program and the number of false detections made (i.e. the total number of activities detected minus the number of correct detections). These numbers can be compared to the number of activities present in the reference segmentation, noted on the right of the sequence identifier. The second measure is a similarity score between the reference segmentation (**R**) and the segmentation obtained by the program (**F**). In short, it represents how well the proposed approach detects all the activities in their *totality*. This score is in the interval [0,1], 1 being a perfect match, and is computed in the following fashion:

1. Considering segments as sets of frames, a matching score for each pair (**r**,**f**) in **R**×**F** is computed as the cardinal of their intersection divided by the cardinal of their union;
2. For each **r**, the (**r**,**f**) pair with the maximum matching score is noted. The value of segment **f** is set to the score of (**r**,**f**) divided by |**R**|;
3. The value of all other segments in **F** is set to a penalty value of $-1/(2\cdot|\mathbf{R}|)$ for false detection;
4. The values computed in step 2 and 3 are summed. A negative score is set to 0.

The reference segmentation in these two measures is the mean of manual segmentations performed by ten volunteers. Considering the small standard deviation in these different segmentations, which was always smaller than 6 frames, the mean of segmentations represents a robust estimated ground truth.

Table 1 presents the results of ***MaxS-SimF*** and ***MaxS-GenF*** on all of the test sequences. It can be observed that both combinations make very few false detections. Also, in both cases, in sequences of type A and B, activities are almost always detected in their totality as their high scores show. The

[3] If over 90% of the length of a segment overlaps a reference segment, the activity is correctly detected.

advantage of using ***GenF*** rather than ***SimF*** is observed with the scores on sequences of type C.

Sequence (number of activities)	***MaxS - SimF***		***MaxS - GenF***	
	Detections (good:bad)	Score	Detections (good:bad)	Score
A_1 (2)	(2 : 0)	0.98	(2 : 0)	0.98
A_2 (4)	(4 : 0)	0.92	(4 : 0)	0.92
A_3 (3)	(3 : 0)	0.92	(3 : 0)	0.92
A_4 (3)	(3 : 0)	0.92	(3 : 0)	0.92
A_5 (2)	(1 : 1)	0.73	(1 : 1)	0.73
B_1 (5)	(5 : 0)	0.74	(5 : 0)	0.74
B_2 (2)	(2 : 0)	0.93	(2 : 0)	0.93
B_3 (4)	(3 : 1)	0.81	(3 : 1)	0.81
B_4 (3)	(3 : 0)	0.86	(3 : 0)	0.86
B_5 (4)	(3 : 0)	0.69	(4 : 0)	0.87
C_1 (2)	(1 : 0)	0.41	(1 : 1)	0.85
C_2 (5)	(2 : 0)	0.32	(2 : 0)	0.32
C_3 (5)	(1 : 1)	0.36	(2 : 2)	0.70
C_4 (3)	(3 : 0)	0.86	(3 : 0)	0.92
C_5 (5)	(1 : 1)	0.30	(1 : 1)	0.30

Table 1. *SimF / GenF* comparison on all test sequences

Table 2 presents the results of ***OptS-GenF*** on all of the test sequences. Since ***OptS*** is non-deterministic, these results are presented as means and standard deviations based on 500 runs of the program on each sequence. It can be observed that ***OptS*** generally makes little false detection but generally makes fewer good detections than ***MaxS***. Nonetheless, in the most simple cases, ***OptS*** performs well and represents an interesting alternative to ***MaxS*** in terms of efficiency: while the running time of ***MaxS*** on the tests sequences ranges from 4 to 240 seconds, the running time of ***OptS*** ranges from 0.4 to 2 seconds[4].

Sequence (number of activities)	***OptS – GenF***			
	Detections (good:bad)		Score	
	mean	st.dev.	mean	st.dev.
A_1 (2)	(1.53 : 0.00)	±(0.71 : 0.00)	0.63	±0.30
A_2 (4)	(2.61 : 0.00)	±(0.67 : 0.00)	0.47	±0.14
A_3 (3)	(1.23 : 0.03)	±(0.66 : 0.16)	0.31	±0.19
A_4 (3)	(1.71 : 0.02)	±(0.78 : 0.13)	0.45	±0.22
A_5 (2)	(1.76 : 0.24)	±(0.43 : 0.44)	0.68	±0.16
B_1 (5)	(3.57 : 0.04)	±(0.61 : 0.21)	0.51	±0.09
B_2 (2)	(1.00 : 0.78)	±(0.00 : 0.51)	0.16	±0.17
B_3 (4)	(3.01 : 0.76)	±(0.78 : 0.70)	0.80	±0.11
B_4 (3)	(2.10 : 0.13)	±(0.39 : 0.34)	0.61	±0.14
B_5 (4)	(2.86 : 0.05)	±(0.72 : 0.22)	0.62	±0.15
C_1 (2)	(1.80 : 0.20)	±(0.40 : 0.40)	0.79	±0.07
C_2 (5)	(1.05 : 0.08)	±(0.22 : 0.22)	0.14	±0.06
C_3 (5)	(1.07 : 0.97)	±(0.78 : 0.78)	0.36	±0.13
C_4 (3)	(1.73 : 0.34)	±(0.49 : 0.47)	0.54	±0.11
C_5 (5)	(0.84 : 0.21)	±(0.37 : 0.43)	0.17	±0.03

Table 2. *OptS - GenF* results on all test sequences (500 runs)

[4] Running times are based on a C++ implementation running on a P4 at 3GHz.

4. CONCLUSIONS & FUTURE WORK

This paper presents a new approach for the temporal segmentation of human activities in video sequences. This approach requires no prior knowledge on the activities to be detected and does not impose severe constraints on the type of the activity. We have proposed two different versions of the individual signal segmentation techniques and of the segmentation fusion techniques respectively.

MaxS-GenF proved to be effective on simple and moderately complex sequences. Moreover, all presented implementations have a low rate of false detections, making their detections reliable. It is also worth mentioning the noise robustness of the algorithms, which faced noisy skeletons in all test sequences. It was also demonstrated that in simple cases ***OptS*** can be effective and efficient. Finally, while a few thresholds were involved in the presented algorithms, experimental results showed that reliable results were obtained on all test sequences using fixed thresholds.

5. ACKNOWLEDGMENTS

We would like to thank NATEQ for the financial support of this research through a postgraduate scholarship. We would also like to thank Annette Schwerdtfeger and the ten volunteers for their help.

6. REFERENCES

[Bob01a] Bobick, A.F. and Davis, J.W., The Recognition of Human Movement Using Temporal Templates, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.23, no.3, 2001, pp. 257-267

[Cut00a] Cutler, R. and Davis, L.S., Robust Real-Time Periodic Motion Detection, Analysis, and Applications, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.22, no.8, 2000, pp. 781-796

[Pol94a] Polana, R. and Nelson, R., Low Level Recognition of Human Motion, *IEEE Computer Science Workshop on Motion of Non-Rigid and Articulated Objects*, Austin, TX, 1994, pp.77-82

[Vig03a] Vignola, J., Lalonde, J.F. and Bergevin, R., Progressive Human Skeleton Fitting, *Proceedings of the 16th Conference on Vision Interface*, 2003, pp. 35-42

[Yaz04a] Yazdi, M., Branzan-Albu, A. and Bergevin, R., Morphological Analysis of Spatio-Temporal Patterns for the Segmentation of Cyclic Human Activities, *17th International Conference on Pattern Recognition*, 2004, pp. 240-243

[Yur94a] Yuret, D., *From Genetic Algorithms to efficient optimization,* Technical Report 1569, MIT AI Laboratory, 1994

Appearance Based Recognition of Complex Objects by Genetic Prototype-Learning

Martin Stommel
Realtime Learning Systems
University of Siegen
Hoelderlinstrasse 3
57068 Siegen, Germany
stommel@fb12.uni-siegen.de

Klaus-Dieter Kuhnert
Realtime Learning Systems
University of Siegen
Hoelderlinstrasse 3
57068 Siegen, Germany
kuhnert@fb12.uni-siegen.de

ABSTRACT

This paper describes a method to recognize and classify complex objects in digital images. To this end, a uniform representation of prototypes is introduced. The notion of a prototype describes a set of local features which allow to recognize objects by their appearance. During a training step a genetic algorithm is applied to the prototypes to optimize them with regard to the classification task. After training the prototypes are compactly stored in a decision tree which allows a fast detection of matches between prototypes and images. The proposed method is tested with natural images of highway scenes, which were divided into 15 classes (including one class for rejection). The learning process is documented and the results show a classification rate of up to 93 percent for the training and test samples.

Keywords

Pattern Recognition, Decision Trees, Genetic Algorithms

1. INTRODUCTION

The recognition of complex objects in digital images is one of the major tasks in computer vision with many applications in automatisation. Traditional graph based or grammar based approaches model complex objects as sets of homogeneous image regions [BB97][KC03]. The challenge in this approach is to achieve a robust segmentation under varying lighting conditions. Recent appearance based approaches avoid the segmentation step. Instead, pose, lighting conditions and the object composition are part of the model. Murase and Nayar [MN95] showed experimentally that it is possible to estimate the pose and illumination projecting an unknown image into a low dimensional subspace constructed from the sample images. Ettelt [Ett02] applied a decision tree to accomplish a fast object detection with an intensity based model. Low-pass filters were used to reduce the search space. Olson and Huttenlocher [OH97] used an edge-pixel based object model for object recognition in a military application. Nelson and Selinger [NS00] researched unsupervised learning in the context of appearance based object recognition.

The method presented in this paper uses a decision tree for the representation of a prototype based object model. This model combines edge-based and intensity based features in a uniform way. A genetic feature selection algorithm is used to obtain prototypes which generalize over one class but separate between different classes. The algorithm is tested with car images from highway videos using the background of the images as a natural rejection class.

VRBF RGNQS o`odqr oqnbddc⁄mfr+ ⁄HRAM 7/,8/20//,8,4
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. OBJECT RECOGNITION

Structure of Prototypes

The use of prototypes is motivated by a probabilistic understanding of the object recognition process. From this point of view, an image of width *w* and height *h* is perceived as the joint ensemble $C_{0,0}C_{1,0}...C_{w,0}...C_{w,h}$ where $C_{x,y}$ is a triple (c_{xy}, A_C, P_{Cxy}) with c_{xy} denoting a random value (the colour at coordinate x,y) which takes on values from the alphabet A_C (a discrete colour space) with the probabilities P_{Cxy}. Assuming we have a certain number of sample images which represent objects of different classes a simple nearest neighbour classification algorithm would execute a pixelwise comparison between a new image and the sample images

and then return the most similar one as the result. Following the probabilistic approach this can be regarded as the maximisation of the conditional probability

$$P(k \mid c^u_{0,0} = c^k_{0,0},...,c^u_{w,h} = c^k_{w,h})$$

that the unknown image corresponds to a certain sample image given the result of the pixelwise comparison. Here c^k_{xy} denote the pixel value for the coordinate *x,y* of sample image *k* and c^u denote the pixel values of the unknown image. From the observation that in natural images the colour is nearly constant over small distances follows that some pixels have a negligible influence on *P(k)*. To achieve a viable computation time we exclude these pixels from the recognition process and benefit more from the information contained in the geometric relationship between the remaining pixels. In this paper the notion of a prototype is used to refer to such a set of pixels. More abstractly, a prototype is regarded as a list of local features $F=\{f_1, f_2,...,f_n\}$. Each feature $f = \{x,y,T,v\}$ describes an object with regard to the the position (x,y) of the feature measured in pixels, the type *T* of the feature and the discrete number *v* giving the value of the feature up to a sufficient accuracy. In this paper the red, green and blue component of a pixel, as well as the gradient orientation serve as features.

The comparison of a prototype with an unknown image is executed using special thresholds t_R, t_G, t_B, t_ϕ, according to the type of the feature. We obtain the box classifier

$$box(x,y) = \begin{cases} 1, if \left| c_{xy} - v \right| < t_{R,G,B} \\ 0, otherwise \end{cases}$$

for RGB-features and an analogous classifier for the gradient direction which takes care of the fact that the domain of *v* is a modulo ring here. With the number

$$M_{feat} = \sum_{i=1}^{n} box(x_i + x_{Offs}, y_i + y_{Offs})$$

of positive classifications for the features $\{f_1, f_2,...,f_n\}$ shifted by the offset (x_{Offs},y_{Offs}) relatively to the image we obtain a matching function

$$match(\{f_1,..., f_n\}, x_{Offs}, y_{Offs}) = \begin{cases} 1, if\, n = M_{feat} \\ 0, otherwise \end{cases}$$

which returns one if a prototype matches to the relative image position (x_{Offs},y_{Offs}) with regard to a number of fixed thresholds.

Genetic Feature Selection

After creating an initial set of prototypes a number of training cycles are executed which consist of a generation and a mutation operation. After each operation a measure of fitness is computed for every new prototype. The best prototypes are kept for further processing, while the worst prototypes are discarded. They will be replaced by new ones during the next generation or mutation step.

2.1.1 Prototype Generation

New prototypes are initialised with a random number of features at random positions. If the gradient at a feature position is above a certain threshold, the direction of the gradient is used as feature. Otherwise the red, green and blue component of the pixel serve as features. A [-0.5 -1 0 +1 +0.5] filter matrix is used to compute the horizontal and vertical gradient. The orientation is computed via the arc tangent. Since the box classificator is sensitive to impulsive noise, the sample images are first filtered with the known olympic filter.

2.1.2 Selection

The fitness of prototypes is measured in terms of selectivity and generalisability. A high selectivity means that a prototype matches only the desired class and produces few false positive results. A high generalisability on the other hand means that a prototype matches many sample images from one class. To determine the generalisability the number of matching relative positions

$$M_{prot} = \sum_{x,y \in I} match(F,x,y)$$

between an image *I* and the prototype *F* from class *k* is determined. An image is covered by *F* if there is at least one match. Denoting the number of images in class *k* covered by *F* with M_{cov}, the generalisability *g* is computed according to the equation

$$g = \frac{M_{\mathrm{cov}}}{class\ size(k)}. \qquad (1)$$

The division by the class size is used to normalize the result to the interval [0 1]. To determine the selectivity the total number of matches of *F* to images from other classes than *k* is computed according to the equation

$$H = \sum_{I \notin k} \sum_{x,y \in I} match(F,x,y).$$

To speed up the computation a upper limit H_{Stop} of total matches is introduced, so the matching procedure can be stopped early for non-selective prototypes. The selectivity *s* is then computed as

$$s = 1 - \frac{\log 10(1+H)}{\log 10(H_{Stop})}. \qquad (2)$$

The logarithm expresses that the selectivity is more interesting for lower numbers of matches. Like the generalisability the selectivity is normalized to the interval [0 1]. The fitness is then computed as the product of the selectivity and the generalisability.

2.1.3 Mutation

Mutation is carried out here by adding zero-mean gaussian noise of a certain standard deviation to the feature coordinates. If an edge feature moves into a homogenous region or vice versa, the type of the feature is adapted. Features are discarded if their coordinates leave the image borders. The prototypes are randomly chosen for mutation. The probability of a prototype to be chosen depends linearly on its position in the list of prototypes sorted by fitness.

Classification via Decision Trees

After training the prototypes are converted into a more compact decision tree representation which allows a fast image recognition. The tree structure is vaguely related to Quinlan's C4.5-Algorithm [Qui93]. The decision tree is used to classify all pixel positions of an image, which corresponds to adding an offset to the position of every feature *{x,y,T,v}* of a prototype. Since single non-selective prototypes have a strong influence on the search result, a further threshold is introduced to exclude bad prototypes from the tree generation.

3. EXPERIMENTS

Data Base

The experiments were conducted on a data base of 1900 car images and 120 images of highway scenes without cars. The car images stem from different highway videos and were cut out by hand and classified by a human. To make the object recognition process independent of the context, the background of the car images was removed manually. Figure 3.1 shows four cars from the data base in different positions, sizes and lighting conditions.

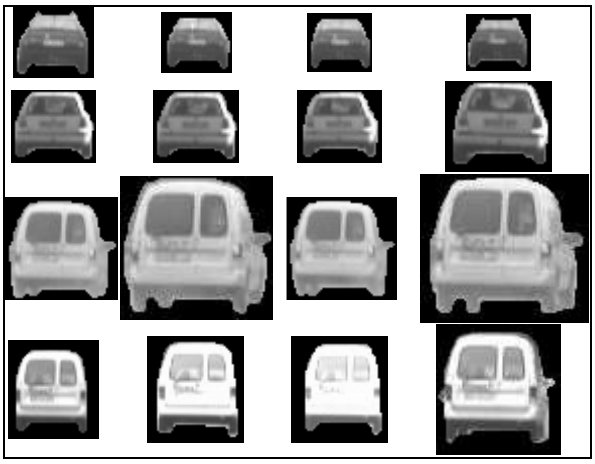


Figure 3.1. Examples from the car data base

The Learning Process

For the experiments the number of prototypes was set to 35 per class. Initial experiments on a two class problem showed that the number of prototypes is uncritical for values between 20 and 200. The learning process consisted of 899 training cycles. The number of prototypes obtained by a generation and mutation operation in every cycle was set to five.

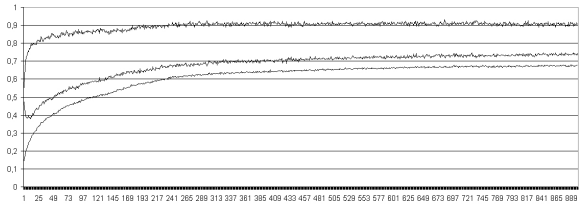


Figure 3.2. Learning rate during training: mean selectivity (upper curve), mean generalisability (mid), mean fitness (lower curve)

Figure 3.2 shows the learning curves for the genetic feature selection algorithm. The learning rate is measured in terms of mean selectivity and mean generalisability over all prototypes. It can be seen that the curves steadily increase during the first 300 training cycles and later converge to fixed values. The theoretical maximum of 1.0 resulting from the equations 1 and 2 is not reached in practice. From figure 3.3 it can be seen that for some classes no optimal set of prototypes can be determined.

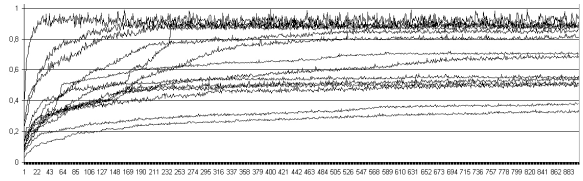


Figure 3.3. Mean fitness separated by classes

The reproduction of prototypes by mutation reduces the variety of sample images whose features are represented by the prototypes. Figure 3.4. illustrates this effect showing the number of different sample images used to form the prototypes of one class averaged over all classes. This observation has lead to the fixed assignment between classes and a set of prototypes.

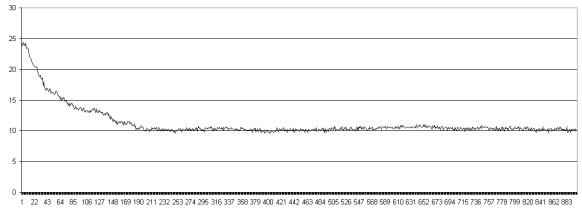


Figure 3.4. Variety among prototypes

Figure 3.5 shows the coverage of the training samples by the prototypes. The rate of recognized training samples stays between 92 and 100 percent during the training process. The curve does not approach 100 percent because the set of prototypes is not trained for a complete coverage of the sample images as a whole. Instead the algorithm searches for single prototypes which represent good features to recognize different classes of objects.

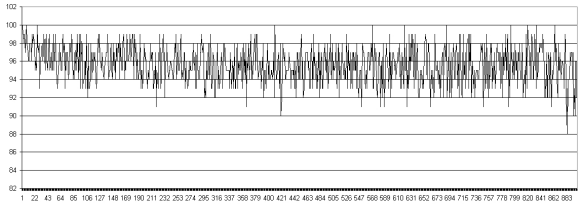


Figure 3.5. Coverage of the training samples

Classification Results

The classification results for the trained prototypes are shown as confusion matrices. To facilitate the computation of confusion matrices objects and background were treated separately. All samples images were classified giving one vote to every image, i.e. the distribution of positive results was normalised to a sum of 1.0 for every image. The background images were used to compute the classification rate for the rejection class. Here both positive and negative results are counted. Table 3.1 shows the proportion of true positives and the total number of predictions (from another training process with identical parameters). A classification rate of 93.1 percent is reached after 25 training cycles, which is an improvement of 6.8 percent over the untrained prototype set. For the following training cycles the results are worse, which seems to be a result of the decreasing variety among the prototypes. For later training cycles the accuracy increases again as a result of the growing generalisability. The similarity for the training and test samples indicates that the results are representative.

Training cycles	Training samples	Test samples
0	0.883	0.863
25	0.926	0.931
50	0.848	0.859
100	0.870	0.874
200	0.850	0.847
400	0.890	0.894
800	0.927	0.908

Table 3.1. Accuracy of the object recognition

Figure 3.6 shows the confusion matrix after 800 training cycles. Row 1 represents the rejection class. As apparent from the pictures the false matches are not evenly distributed over the confusion matrices but can be divided into objects which are not recognized at all and classes which are often misclassified as one single other class. The first group results from the training of single prototypes rather than sets of prototypes. The second kind of error is caused by single ambiguous prototypes.

4. CONCLUSION, FUTURE WORKS

The object recognition method presented in this paper employs trained prototypes to find and classify objects by their appearance. The training is conducted with objects from real world images using their natural background as rejection class.

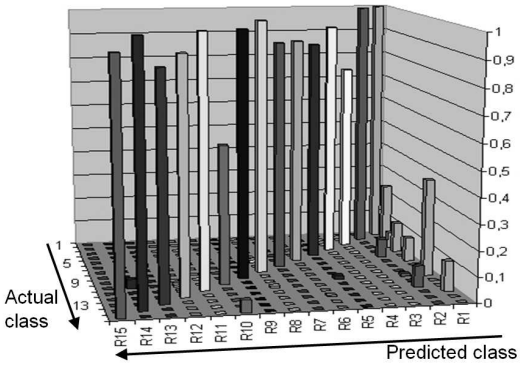


Figure 3.6. Confusion matrix for the test samples

As our experiments show, the structure of prototypes is capable of recognizing complex objects without a previous segmentation step and an explicit higher-level model. The problem of model construction and comparison to noisy segmentation results is avoided. The analysis of the learning process shows that single prototypes can be trained successfully to generalize over one class and at the same time to distinguish between objects of different classes. The experiments show also a certain potential for improvements concerning the coverage of the samples by the prototypes. Thus, in our future work we will investigate how whole sets of prototypes can be trained instead of single prototypes.

5. REFERENCES

[BB97] Burge, M., Burger, W., Learning Visual Ideals. Proc. of the 9th ICIAP, Florence, Italy, Sept. 1997.

[Ett02] Ettelt, E. Appearence Based Object Recognition by Use of Optimized Template Trees. Doctoral Thesis, TU Muenchen, Germany, 2002.

[KC03] Kerr, J., Compton, P., Towards Generic Model-based Object Recognition by Knowledge Acquisition and Machine Learning. Mixed Initiative Intelligent Systems, Eds. Gheorghe Tecuci, Morgan Kaufmann, Acapulco, Mexico, 2003, pp. 107-113.

[MN95] Murase, H., and Nayar, S.K. Visual learning and recognition of 3d objects from appearance. Int. Journal of Computer Vision, 14(1):5-24, Kluwer, January 1995.

[NS00] Nelson, R.C., Selinger, A., Learning 3D Recognition Models for General Objects from Unlabeled Imagery: An Experiment in Intelligent Brute Force. 15th Int. Conf. on Pattern Recognition (ICPR 2000), Barcelona, Spain, Sept. 3-8, 2000, Vol.1, pp.1-8.

[OH97] Olson, C.F., Huttenlocher, D.P., Automatic Targed Recognition by Matching Oriented Edge Pixels. IEEE Transactions On Image Processing, Vol.6, No.1, January 1997.

[Qui93] Quinlan, J.R. C4.5: Programms for Machine Learning. Morgan Kaufmann, San Mateo, California, 1993.

Physically-based Simulation of Hair Strips in Real-Time

Hasan Doğu Taşkıran and **Uğur Güdükbay**
Department of Computer Engineering
Bilkent University
Bilkent 06800 Ankara, Turkey
Phone: {(+90) 533 513 1426, (+90) 312 290 1386}
E-mail: {taskiran, gudukbay}@cs.bilkent.edu.tr

ABSTRACT

In this paper, we present our implementation of physically-based simulation of hair strips. We used a mass-spring model followed by a hybrid approach where particle systems and the method of clustering of hair strands are employed. All the forces related to springs are implemented: gravity, repulsions from collisions (head and ground), absorption (ground only), frictions (ground and air), internal spring frictions. Real-time performance is achieved for physically-based simulation of hair strips and promising results in terms of the realistic hair behavior and hair rendering are obtained.

Keywords: hair simulation, physically-based, real-time, strip-based

1 INTRODUCTION

Hair is very important to create natural looking virtual characters. However, producing realistic looking hair is one of the most challenging problems in computer graphics. There are approximately 100,000 to 150,000 individual hair strands to simulate on the scalp of a human head. Besides, different hairs have different colors and varying degrees of waviness and thickness. These complexities and the number of hair strands on the head create difficulties for generating a formulation for modeling and simulating hair [HT00, LK01]. There are three main aspects of simulating hair: *hair modeling*, *hair animation (dynamics)*, and *hair rendering*. There are various approaches to hair modeling, dynamics and rendering such as the explicit models, particle systems [BCN03], volumetric textures, cluster models and hair as a fluid etc. A detailed survey and discussion on the state of the art in these aspects can be found in Hadap et al. [HT00].

Each individual hair strand interacts with its environment. From a dynamics point of view, most important interactions occur with the head, the body itself, and the other strands. Strands are also responsive to the external forces such as the movement of the head, collisions with obstacles, the wind and the forces created by the static electricity, etc. All these interactions have to be simulated for each strand to achieve realistic hair behavior and accurate and smooth animation. Since the number of strands is very large we need somehow to simplify physical formulas for the motion of the hair and fast and high-quality collision detection mechanisms to be able to do the simulation in real-time [KH00, KH01, CJY02].

WSCG SHORT Papers Proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2 OUR WORK

In this paper we are mainly concerned with the dynamics and rendering aspects of human hair simulation. We used a mass-spring model for implementing hair dynamics and followed a hybrid approach where particle systems and the cluster model are used.

Hair Modeling and Rendering

Human hair may be modeled in strips (clusters) but this somehow results in less realistic images. However, increasing the number of the strips may solve this problem. The more strips you create, the better you approximate the hair. Of course, the better you approximate the hair, the more calculations are needed for the physically-based simulation of the hair [WLL+03, YXWY00, KH01].

The rendering of hair involves dealing with hair color and texture, shadows, specular highlights, varying degrees of transparency, and anti-aliasing. Mainly, we only used strips to visualize the hair realistically in our implementation. An alpha blended texture, which is used to make the strips that has randomized tips, is mapped onto the patches that were created on the fly from top to bottom of each hair strip. The texture and the method we used is the same as in [KH01]. Since we mainly concentrated on the dynamics and rendering of the hair, no hair style was applied. However, some curliness was given to the strands by procedural meth-

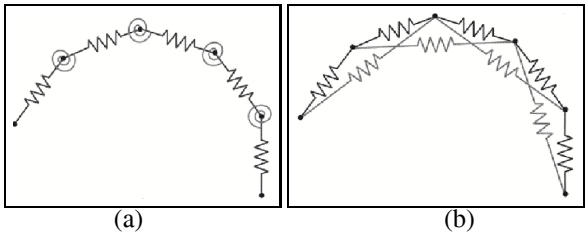


Figure 1: The angular spring model for hair strands (a) and the linear spring representation of the corresponding model (b).

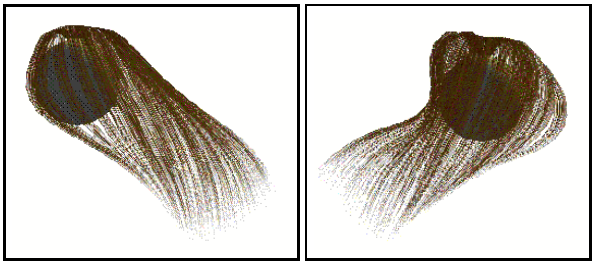


Figure 2: Collisions with the head and the bending stiffness of the strips.

ods. The connection points of hairs are determined by using the tools 3ds max 4.2 and Milkshape3D.

Hair Dynamics

A simple mass-spring system for an individual hair's dynamics includes a damping factor that is used to calculate the bending stiffness. To be able to simulate the bending stiffness and elasticity of each hair strand, we may see the hair as an oriented particle system as shown in Figure 1 where each spring constant affects both bending and stretching of the strand [iAUK92, RCT91, HT01]. The effects of the bending stiffness and collision with the head is shown in Figure 2.

The spirals shown at each particle position represents the angular springs. This is used to obtain the bending stiffness needed for each hair strand. However, linear springs are used to represent those angular springs for uniformity and performance. Linear representations of angular springs supply bending and torsion effects. We used springs with relatively lower spring constants and lower internal friction constants in their implementation. Thus, some other springs are connected with particles such that each particle also has a connection with the particle next to its neighbor.

The implemented forces, which have an impact on the motion of the hair strands, are spring forces, gravity, repulsions from collisions (head and ground), absorption (ground only), friction (ground and air), and internal spring frictions.

Although the cantilever dynamics and collisions of each hair strand with the body are within the reach of current computing power, modeling the complex hair-to-hair interactions is still a challenge [iAUK92, RCT91, KH01, KH00]. Therefore, hair-to-hair interactions are not considered in this implementation to achieve real-time performance.

3 IMPLEMENTATION

The hair simulation algorithm is given in pseudo-code in Figure 3 and the algorithm for solving the equations of motion of the hair particles is given in Figure 4. At each frame update, we iterate the springs used in the simulation a number of times according to the seconds-per-frame rate and a predefined maximum seconds that may pass between two frames of the animation. This is needed in order to prevent a pass over of the non-precise time calculation.

A simulation is first initialized, solved and then updated. In our implementation, each hair strand is a separate simulation of particles and springs that hold those particles together. Particles are initialized to their stable states at each frame, i.e. the forces applied on the masses are set to zero. Each new force adds a new constraint to the movement of the mass. A spring is just an additional force on the two masses that this spring is virtually connecting together.

At each frame of the simulation, each hair strip, which is also a simulation itself, is simulated independently. All the external forces and the forces that stem from the springs are added together, which give the hair its final movement direction. When the end of the frame update comes, the changes in the velocities of the particles are calculated, which then affects the change in the position of the mass. Forces are applied according to the time between two frames and the motions of the virtual particles are simulated afterwards.

Masses have properties such as their position, and the force that is applied on this mass at an instance. Springs have properties such as their spring constants, inner friction constants and their stable lengths. There are other constants such as the air friction constant, ground friction constant, ground repulsion constant, ground absorption constant and most importantly the gravitational acceleration constant. All these constants are given when initializing the simulation. A simple sphere is used for the head and a simple plane is used for the ground. No collisions applied with the body of the model.

After the forces on each particle are calculated, these forces are applied on each particle. Basic physics formulas apply at this point and the velocity of the particle and its new position are calculated. We used Euler Method for solving the equations of motion. This method is not always accurate, but it is simple to implement.

4 RESULTS

We implemented the described methods using the C++ language and OpenGL[1]. Performance results of our im-

[1]Animations generated with our implementation can be seen at `http://www.cs.bilkent.edu.tr/~gudukbay/hairmodeling.html`.

```
procedure SolveHairSimulation {

  for i from 0 to numHairStrands do {
    // Initialize the masses for this frame
    for j from 0 to hair(i)->numParticles do
        hair(i)->particle(j)->force := 0;

    // Solve spring forces and apply forces
    // on the masses that this spring connects
    for j from 0 to hair(i)->numSprings do {
      hair(i)->spring(j)->force := 0;
      // Solve spring equation by taking the
      // spring's internal friction into account
      solve(hair(i)->spring(j));

      // Apply the spring force to the masses
      // in opposite directions
      applyForce(hair(i)->spring(j)->particle1,
                 hair(i)->spring(j)->force);
      applyForce(hair(i)->spring(j)->particle2,
                 -1 * hair(i)->spring(j)->force);
    }

    // Apply other forces
    for j from 0 to hair(i)->numParticles do {
      // Apply gravitation
      applyForce(hair(i)->particle(j),
                 gravitation_force);
      // Apply air friction
      applyForce(hair(i)->particle(j),
                 air_friction_force);
      // Collide with the head
      if hair(i)->particle(j) is colliding
         with the head then do
         applyForce(hair(i)->particle(j),
                    head_repulsion_force);

      // Collide with the ground
      if hair(i)->particle(j) is colliding
         with the ground then do
      {
        // Absorb some energy from the hair
        // by applying an absorption force
        applyForce(hair(i)->particle(j),
                   ground_absorption_force);
        // Apply the ground friction
        applyForce(hair(i)->particle(j),
                   ground_friction_force);
        // Apply the ground repulsion
        applyForce(hair(i)->particle(j),
                   ground_repulsion_force);
      }
    }
  }
}
```

Figure 3: The algorithm for solving the hair simulation

```
procedure SimulateHair(dt) {

  // Solve the simulation
  SolveHairSimulation();

  for i from 0 to numHairStrands do {
    for j from 0 to hair(i)->numParticles do {
      // Calculate new velocity of each particle
      hair(i)->particle(j)->velocity +=
        dt * (hair(i)->particle(j)->force /
              hair(i)->particle(j)->mass);
      // Calculate new position of each particle
      hair(i)->particle(j)->position
        += dt * hair(i)->particle(j)->velocity;
    }
  }
}
```

Figure 4: The algorithm for solving the equations of motion for hair particles

NHS	*PES*	*NSP*	*SES*	*NSS*	*FPS*
100	10	1000	17	1700	62.50
100	15	1500	27	2700	62.50
100	20	2000	37	3700	62.50
100	25	2500	47	4700	31.50
200	5	1000	7	1400	62.50
200	10	2000	17	3400	62.50
200	12	2400	21	4200	32.50
200	14	2800	25	5000	10.60
300	6	1800	9	2700	62.50
300	7	2100	11	3300	45.20
300	8	2400	13	3900	32.00
300	10	3000	17	5100	6.41
400	3	1200	3	1200	62.50
400	5	2000	7	2800	45.20
400	6	2400	9	3600	21.25
400	7	2800	11	4400	8.00

Table 1: Performance results of the simulation. **NHS**: Number of Hair Strips, **PES**: Particles at each Strip, **NSP**: Number of Simulated Particles, **SES**: Springs at each strip, **NSS**: Number of Simulated Springs, **FPS**: Frames-per-Second.

plementation, running on a computer that has a Pentium IV (HT) 2.80GHz CPU, NVIDIA GeForce FX 5600 chipset VGA Adapter and 512 MB RAM, is given in Table 1.

All the other parameters that affect the frame rate were held fixed as the number of particles are increased in the simulation. As the number of particles changes, the number of springs connecting the particles together changes too. Each additional particle adds two more springs to the system. Total number of springs are the number of hair strips multiplied by the number of springs for each strip.

We may derive from the table that after some point, an increase in the number of particles - and so the springs - creates an exponential degradation in performance. Of course there are other factors affecting these results such as the usage of alpha blending, increase in the number of patches as the number of particles increase, the width of the patches used, etc. However, we could still obtain the realtime performance with 300 hair stripes each having 8 particles, or 400 hair stripes each having 5 particles. Most of the force calculation steps are dedicated to the springs. The spring force calculation includes a square-root calculation, 3 multiplications and a division operation. It is possible to optimize the simulation for speed. We could create a look-up table for the square-root values to eliminate the costly square-root function calls.

5 CONCLUSION AND FUTURE WORK

We presented our implementation of physically-based simulation of hair strips. It is still very far to imagine hair blowing with full rendering and collisions in real-time, however we obtained promising results. We will work on other applications of our implementation.

As a future work, levels-of-detail for hair may be added as described in [KH01] and [WLL+03, WL03]. This can be done in two ways. Either we may decrease the number of hair strands, or we may decrease the

number of particles used to represent each hair strand while the viewer is far from the hair. Both of these methods reduce the calculations needed in terms of dynamics and graphics.

Besides, other collisions, such as the ones with the body and obstacles, etc., and other external forces, such as the wind, static electricity, etc., could be handled. Wet hair simulation could be done by increasing the weight of each particle in the hair. Another improvement could be adding hair shape modeling into our framework. A comb, which is used to brush the hair would not be just a fine addition, but also it would be a good practice of dynamics.

6 ACKNOWLEDGEMENTS

This work is partially supported by Turkish State Planning Organization (DPT) under Grant No. 2004K120720. 3ds max is a registered trademark of Autodesk; Milkshape3D is a registered trademark of ChumbalumSoft; Pentium is a registered trademark of Intel; GeForce is a registered trademark of NVIDIA; OpenGL is a registered trademark of Silicon Graphics.

References

[BCN03] Yosuke Bando, Bing-Yu Chen, and Tomoyuki Nishita. Animating hair with loosely connected particles. *Computer Graphics Forum*, 22(3):411–418, 2003.

[CJY02] Johnny T. Chang, Jingyi Jin, and Yizhou Yu. A practical model for hair mutual interactions. In *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 73–80, 2002.

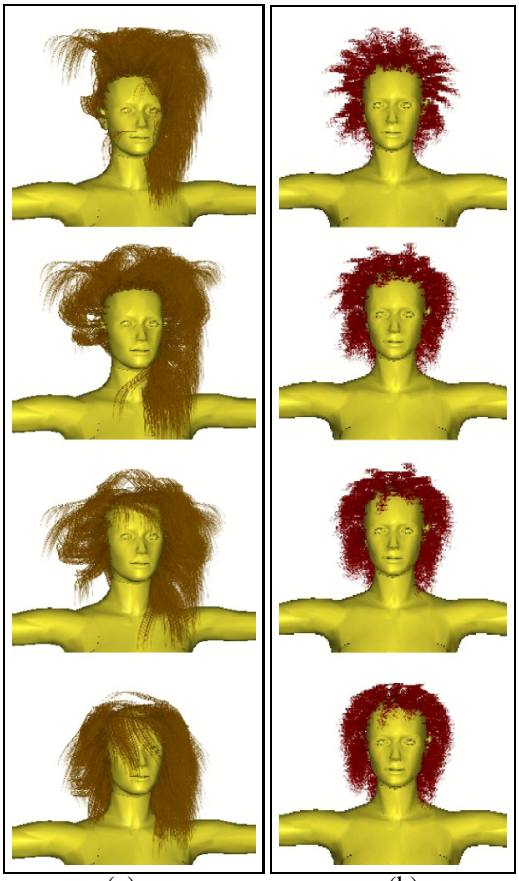
[HT00] Sunil Hadap and Nadia-Magnenat Thalmann. State of the art in hair simulation. In *Proc. of International Workshop on Human Modeling and Animation*, pages 3–9, June 2000.

[HT01] Sunil Hadap and Nadia-Magnenat Thalmann. Modeling dynamic hair as a continuum. *Computer Graphics Forum (Proc. of Eurographics 2001)*, pages 329–338, 2001.

[iAUK92] Ken ichi Anjyo, Yoshiaki Usami, and Tsuneya Kurihara. A simple method for extracting the natural beauty of hair. In *Proc. of SIGGRAPH'92*, pages 111–120, 1992.

[KH00] Chuan Koon Koh and Zhiyong Huang. Real-time animation of human hair modeled in strips. In *Proc. of the Eurographic Workshop on Computer Animation and Simulation*, pages 101–110, 2000.

[KH01] Chuan Koon Koh and Zhiyong Huang. A simple physics model to animate human hair modeled in 2d strips in real time. In *Proc. of the Eurographic Workshop on Computer Animation and Simulation*, pages 127–138, 2001.



(a) (b)

Figure 5: Still frames generated with our implementation, showing long straight hair (a) and curly hair (b) while falling down after jumping.

[LK01] Doo-Won Lee and Hyeong-Seok Ko. Natural hairstyle modeling and animation. *Graphical Models*, 63(2):67–85, 2001.

[RCT91] R. E. Rosenblum, W E. Carlson, and I. E. Tripp. Simulating the structure and dynamics of human hair: modeling, rendering and animation. *The Journal of Visualization and Computer Animation*, pages 141–148, 1991.

[WL03] Kelly Ward and Ming C. Lin. Adaptive grouping and subdivision for simulating hair dynamics. In *Proc. of the 11th Pacific Conference on Computer Graphics and Applications*, pages 5–9, 2003.

[WLL+03] Kelly Ward, Ming C. Lin, Joohi Lee, Susan Fisher, and Dean Macri. Modeling hair using level-of-detail representations. In *Proc. of the 16th International Conference on Computer Animation and Social Agents (CASA 2003)*, page 41, 2003.

[YXWY00] Xue Dong Yang, Zhang Xu, Tao Wang, and Jun Yang. The cluster hair model. *Graphical Models*, 62(2):85–103, 2000.

An Efficient One-Dimensional Fractal Analysis for Iris Recognition

Chuan Chin Teo
Faculty of Information Technology
Multimedia University
Jalan Multimedia, 63100,
Cyberjaya, Selangor, Malaysia
ccteo@mmu.edu.my

Hong Tat Ewe
Faculty of Information Technology
Multimedia University
Jalan Multimedia, 63100,
Cyberjaya, Selangor, Malaysia
htewe@mmu.edu.my

ABSTRACT

An iris recognition system is proposed in this paper. This system implements (a) the combination of proposed black hole search method and integro-differential operators for iris localization, (b) one-dimensional fractal analysis for feature extraction and (c) dissimilarity operator for matching. The black hole search method achieves 100% accuracy of pupil detection for both CASIA and MMU Iris databases. Experiment results show that the proposed system has an encouraging result with low EER to 4.63% in CASIA iris database and 4.17% in MMU iris database. It provides an alternative solution for iris recognition and acts as a spring board for further investigation.

Keywords

Black hole Search, 1-D Fractal Dimension, Dissimilarity Operator.

1. INTRODUCTION

The latest threats of security have led to increased awareness of biometric technologies. Biometric, in general terms, is defined as automated personal identification or authentication based on a physiological and behavioural characteristics [Tis02a]. Fingerprint-based authentication is the most widely accepted and mature biometric, dominating the majority of commercial, civilian, and law enforcement markets. However, it requires physical contact with scanner, and dirt on the finger can affect the recognition.

Recently, iris-based authentication presents promising results for biometric technologies. The iris, as showed in Figure 1, has rich and unique patterns of striations, pits, coronas, freckles, fibers, filaments, furrows, rifts, rings, and vasculature. As reported in Daugman [Dau95a] the uniqueness of every iris parallels the uniqueness of every fingerprint, but owns further practical advantages over fingerprint and other biometrics. This indicates that no two irises are indistinguishable and that the chance of finding two people with identical iris pattern is almost zero [Tis02a], [Bo98a]. Furthermore, the iris pattern is stable throughout life and can be noninvasive authenticated due to its externally visible behaviour [Ma03a]. These advantages make it a promising solution to security and commercial applications.

In this paper, we propose a novel analysis for an iris recognition system based on one-dimensional fractal dimension. The rest of this paper is structured as follows. Section 2 contains a complete account of the overall process of proposed system. Experimental results are displayed in section 3.

WSCG SHORT papers, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

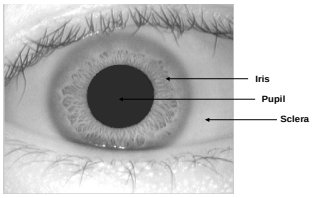


Figure 1. Sample of iris image

2. Iris-based Authentication System

This section describes an overview of all fundamental processes in our iris recognition system. These processes are generally referred to as image acquisition, image preprocessing, feature extraction and matching, respectively.

2.1 Image Acquisition

We are using the iris image database from CASIA Iris Image Database [CAS03a] and MMU Iris Database [MMU04a]. CASIA Iris Image Database contributes a total number of 756 iris images which were taken in two different time frames. Each of the iris images is 8-bit gray scale with a resolution of 320x280. Details of iris image acquisition technique can be referred from [Ma03a]. Nevertheless, MMU Iris Database contributes a total number of 450 iris images which were captured by LG IrisAccess®2200.

2.2 Iris Region Localization

Daugman's integro-differential operators [Dau95a] have been proven effective in iris localization. Our studies serve the dual purposes of gaining a broad understanding on integro-differential operators, and combining a new mathematical solution with it to improve the computational speed and reliability during localization process. For this reason, we introduce a combination of black hole search method and integro-differential operators for iris localization. Firstly, a black hole search method is proposed to detect the center of a pupil. For simple objects like circle and square, the center of mass is at the center of the object [Gre94a]. The center of mass refers to the balance point (x, y) of the object where there is equal mass in all directions [Gre94a]. Both the inner and outer iris boundaries can be taken as circles. Therefore, we can find the center of pupil by calculating its center of mass. The steps of black hole search method are as follows:

1. Find the darkest point of image (referred as black hole) in the global image analysis.
2. Determine a range of darkness (based on 1) designated as the threshold value (t) for identification of black holes.
3. Determine the number of black holes and their coordinates according to the predefined threshold. Calculate the center of mass of these black holes.
4. Construct a square region centered at the estimated center.
5. Repeat step 3 to improve the estimation of actual center of pupil.

The black holes represent the pixels exist within the interior of the pupil. Similarly, the total number of black holes represents the area of the pupil. In Equation 1, Ex denotes the estimated center coordinate-x and Ey denotes the estimated center coordinate-y from the global image analysis, where W and H represent the sum of detected coordinate-x and coordinate-y which satisfy $I(x, y) < t$ respectively.

$$Ex = \frac{\sum_{x=0}^{W-1}\sum_{y=0}^{H-1} x}{WH} \quad \text{if } I(x, y) < t;$$

$$Ey = \frac{\sum_{x=0}^{W-1}\sum_{y=0}^{H-1} y}{WH} \quad \text{if } I(x, y) < t \qquad (1)$$

In order to acquire more accurate result, we repeat the steps and calculate the actual center of mass (Cx, Cy) from the local region analysis where a square window with side 60 centered at Ex, Ey is chosen for the analysis. Equation 2 defines the actual center of the pupil, and w and h represent the sum of detected coordinate-x and coordinate-y which satisfy $I(x, y) < t$ in predefined region respectively.

$$Cx = \frac{\sum_{x=Ex-L}^{Ex+L}\sum_{y=Ey-L}^{Ey+L} x}{wh} \quad \text{if } I(x, y) < t;$$

$$Cy = \frac{\sum_{x=Ex-L}^{Ex+L}\sum_{y=Ey-L}^{Ey+L} y}{wh} \quad \text{if } I(x, y) < t \qquad (2)$$

In Equations 1 and 2, t is the threshold value and $I(x, y)$ is the image intensity. CASIA iris images are captured by near-infrared illumination sensor. Therefore, they have no reflected spot on the pupil. In contrast, MMU iris images are captured by non-intrusive illumination and have less reflected spot on the pupil. As a result, the proposed algorithm is very effective for both CASIA and MMU iris database. Furthermore, the radius can be calculated from the given area (total number of black holes in the pupil), where

$$Radius = \sqrt{\frac{Area}{\pi}} \qquad (3)$$

In order to complete the iris localization process, a Gaussian smoothing is used to filter out the high frequency noise of the image. Next, the outer boundary of an iris can be detected by using integro-differential operators [Dau95a]. This method is found to be efficient and effective in CASIA and MMU iris images as well. Figure 2 shows the detected inner and outer boundary of an iris. At the same time, the localized iris region is positioned into rectangular iris template.

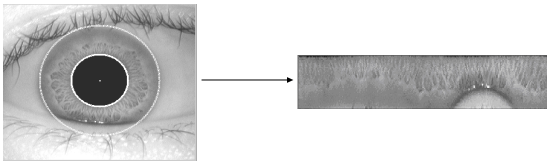


Figure 2. Sample iris after localization

2.3 Image Normalization

An iris has the function of controlling the light entering into the eye by adjusting the size of pupil. Consequently, it causes the elastic deformation in iris image. On the other hand, the size of iris image can be inconsistent due to different distance between the iris and camera.

Thus, it is important to normalize the iris image into a standard size (87x360) using interpolation technique. Figure 3 shows the normalized template of an iris.

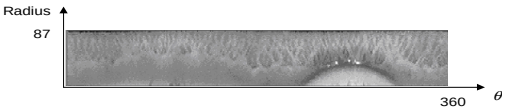


Figure 3. Sample iris after normalization

2.4 Feature Extraction

Fractal geometry can intuitively be thought as irregular geometric representation in human's iris. Thus, the fractal information can be used to illustrate an iris. In fact, the fractal information from a multi-dimensional image can be calculated by constructing it into three-dimensional representation [Low99a], where the additional h-axis is represented by its gray levels. On the other hand, the unique details of the iris generally spread along the angular direction which corresponds to the variation of pattern in vertical direction [Ma04a]. Owing to the two fundamental properties above, we use one dimensional fractal analysis to generate a structured feature vector for an iris.

The earliest step in fractal analysis endeavor is to normalize all the pixels from the preprocessed iris. The gray level value of *I(x, y, h)* for all pixels in the iris template is normalized according to Equation 4.

$$I(x,y,h) = I(x,y,h) \times \frac{L}{H} \qquad (4)$$

where the L is the appropriate window size and H is the maximum value of gray level.

In advance, the pixels within each row along the angular direction are positioned into an appropriate square with L x L window size. This pixel allocation process is illustrated in Figure 4.

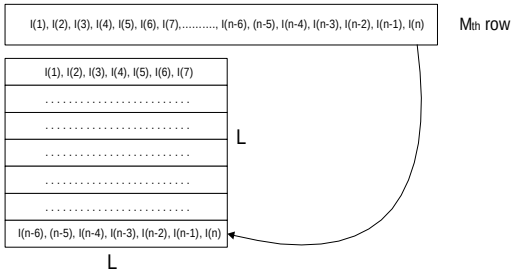


Figure 4. Pixel allocation with L x L window size

Next, a surface coverage method [Ewe93a] is performed to calculate the fractal dimension.

The fractal dimension [Sch91a] is defined as:

$$D = \frac{\log(N)}{\log(1/r)} \qquad (5)$$

where *N* is the number of scaled down copies (difference of dimension between neighboring pixels within a box) of the original object which is needed to fill up the original object and r = 1/L. Consequently, each row of the iris template will produce the fractal information and form a structured feature vector (f) for particular iris.

$$f = \{D_1, D_2, D_3 \ldots, D_{m-3}, D_{m-2}, D_{m-1}, D_m\} \qquad (6)$$

where D_m is the fractal information of m_{th} row in a iris template. In feature extraction stage, we generate 87 features to illustrate an iris code. This record is saved in a database for further comparison.

The approach mentioned above has fast computational speed and less memory usage due to the compression of a list of circular data into a single fractal feature. Moreover, it achieves rotational invariant as the 1-D feature represents the whole circular data.

2.5 Matching

In the matching phase, a simple dissimilarity operator is proposed to compare a pair of iris code. The steps of formulating the dissimilarity operator are described as follows:

1. Initialize a tolerance threshold. In our case, we use *Tt* (= 0.0025).
2. Compute the absolute differences between default feature vector (D1) and current feature vector (D2).
3. Increase the number of occurrence for every absolute feature-by-feature differences that exceeds tolerance threshold through accumulator
4. Calculate the average dissimilarity between two feature vectors.

The equation of our dissimilarity operator can be expressed as:

$$\text{Average Dissimilarity} = \frac{1}{87}\sum_{i=1}^{87} |\, D1i - D2i \,|$$

$$\text{if } |\, D1_i - D2_i \,| > Tt \qquad (7)$$

where D1 is the default feature vector, D2 is current feature vector and *Tt* is the tolerance threshold. In real application, the average dissimilarity equal to zero seldom happens. Therefore, the algorithm to select a good match is based on average dissimilarity

which is nearest to zero and falls below a predefined threshold (*t*).

3. Experiment Results

The proposed black holes search method proves its ability to detect inner boundary of the iris. It exhibits 100% success of pupil detection for both CASIA and MMU Iris databases. Figure 7 shows the actual results of black hole search method.

On the other hand, integro-differential operators work properly on both CASIA and MMU iris database for outer iris detection. However, the failure may occurs due to low contrast, eyelids and eyelashes occlusion. The probability of failure on outer iris detection is around 2% in CASIA and MMU iris database.

The experiment was continued to validate the performance of proposed algorithm. In verification procedure, the False Reject Rate (FRR), False Accept Rate (FAR) and Equal Error Rate (EER) tests were computed in both databases, where Figure 5 and Figure 6 shows the results of CASIA and MMU iris datasets respectively. Accordingly, EER is calculated from the ROC curve. The EER is the point on the ROC curve where the FAR is equal to the FRR. In our experiment on CASIA iris database, the EER is approximately 4.63 %. Nevertheless, MMU iris database has lower EER which is 4.17%.

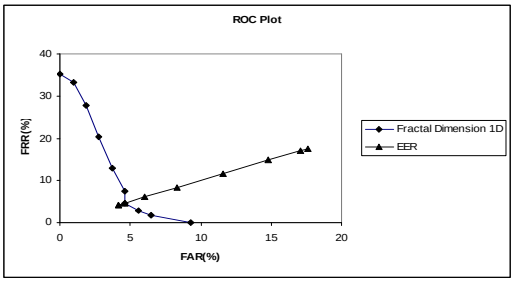


Figure 5. ROC plot for CASIA iris database

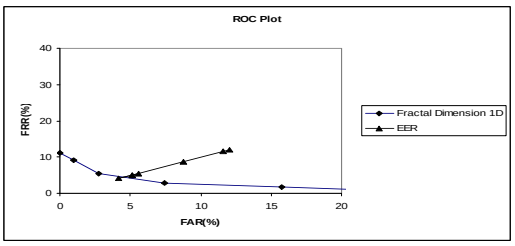


Figure 6. ROC plot for MMU iris database

4. Acknowledgement

This work was supported by Malaysia IRPA (Intensified Research in Priority Areas) funding. The authors wish to thank Institute of Automation, Chinese Academy of Sciences for providing CASIA iris database.

5. Reference

[Tis02a] Tisse, C., Martin, L., Torres, L. and Robert, M. (2002): Person Identification Technique using Human Iris Recognition. In Proc, Vision Interface, pp. 294-299.

[Dau95a] Daugman, J. (1995): High Confident Personal Identification by Rapid Video Analysis of Iris Texture. In Proc, European Convention on Security and Detection, pp. 244-251.

[Bo98a] Boles, W. and Boashash, B. (1998): A Human Identification Technique Using Images of the Iris and Wavelet Transform, IEEE Trans. Signal Processing, vol 46, no.4, pp. 1185-1188.

[Ma03a] Ma L., Tan T., Wang Y. and Zhang, D. (2003): Personal Identification Based on Iris Texture Analysis. IEEE Trans. PAMI, vol 25, no.12, pp. 1519-1533.

[Ma04a] Ma L., Tan T., Wang Y. and Zhang, D. (2004): Efficient Iris Recognition by characterizing Key Local Variations, IEEE Trans. Image Processing, vol 13, no.6, pp. 739-750.

[Low99a] Low, H.K., Chuah, H.T and Ewe, H.T. (1999): A Neural Network Landuse Classifier for SAR Images using Texture and Fractal Information, in Proc. Geocarto International, vol. 14, no.1, pp. 67-74.

[Gre94a] Gregory A., B. (1994): Digital Image Processing: Principles and Applications, Wiley.

[CAS03a] CASIA Iris Image Database (version 1.0): Institute of Automation (IA), Chinese Academy of Sciences (CAS), http://www.sinobiometrics.com

[MMU04a] MMU Iris Image Database: Multimedia University, http://pesona.mmu.edu.my/~ccteo/

[Ewe93a] Ewe, H.T., Au, W.C., Shin, R.T. and Kong, J.A. (1993): Classification of SAR Images Using a Fractal Approach, in Proc. Electromagnetic Research Symposium, pp. 493.

[Sch91a] Schroeder, M.R. (1991): Fractal, Chaos, Power Laws: Minutes from an Infinite Paradise, Freeman.

Multi-agent Animation Techniques for Traffic Simulation in Urban Environments

Wen Tang
School of Computing,
University of Teesside,
United Kingdom
TS 1 3BA, Middlesbrough, UK
w.tang@tees.ac.uk

Tao Ruan Wan
School of Informatics
University of Bradford,
United Kingdom
BD7 1DP, Bradford, UK
t.wan@bradford.ac.uk

ABSTRACT

This paper presents a traffic simulation system based on multi-agent animation method. The behaviour model of individual virtual agent defines the driving characteristics of the vehicle agent and cell-based configurations allow real-time dynamic path planning and efficient traffic flow simulations. By incorporating the advantages of discrete cellular automation algorithms and the continuous model of fluid dynamics, our 3D virtual reality traffic simulation system achieves realistic and accurate simulations in virtual environments.

Keywords

Multi-agent behaviour modelling, Computer Animation, Traffic simulation in virtual environments.

1. INTRODUCTION

Traffic simulation for urban environments has an important role in meeting future urban development and planning. In the context of traffic simulation, numerical results can provide useful and detailed quantitative descriptions of a traffic network, while 3D virtual environments of the simulation demonstrates the overall performance of specific traffic flow situations visually, from which a trained operator can quickly access the dynamic changes within a road network and select plans to enable the network to adapt the situations. For example the operator can specify traffic light signal timing to ease congestions at a particular location point of the network during rush hours or when an incident occurs [van Katwijk and Koningsbruggen 2002]. As the simulation system deals with large numbers of social agents (cars, pedestrians, motorcycles, and bicycles) and each of them has its own behaviour characteristics, the challenge is to model the massively distributed parallel behaviours [Reeves 1983, Metoyer and Hodgins 2000]. However, individual characteristics of traffic elements including human factors, vehicle specifications and dynamics, could be very difficult to model using traditional traffic flow theories or macroscopic simulation approaches []. The multi-agent traffic simulation system for urban environments described in this paper is capable of modelling large volumes of traffic in response to current network loading at the time of interest. The responsive simulation system calculates traffic control management and operations according to different traffic flow situations. The system models the motion behaviours of each individual vehicle agent using a set of object-orientated simulation components, which acts upon the vehicle agents to manage the behaviours of the agents during the simulation. Based on an origin-destination matrix, the system is capable of providing an accurate description of the expected traffic flows using a real-time path-planning algorithm, which estimates route choices and the effect of unexpected congestions.

We present a multi-agent 3D simulation system, in

Conference proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

which a roadway network is represented as strings of cells and vehicles in the roadway-network are modelled as a continuous flow. The discrete space for the roadway network provides a framework for high-speed system update and traffic management registration, while a continuous space and time for modelling vehicle dynamic movements produces realistic traffic flow. Therefore the approach takes the advantage of discrete models at the same time maintains the features of fluid dynamic models. The model is capable of taking into account the human factors for more accurate simulation results. Compared with existing systems, which are mainly two-dimensional simple graphical representations [Nagel and Schreckenberg 1992], a three-dimensional visualisation system is developed based on the approach. At the current development stage, our system is able to simulate small and medium scale traffic flows of urban environments.

2. CONFIGURATION OF TRAFFIC NETWORK

A traffic network in the simulation system is modelled as a connection of desecrated cells. Each road configuration maintains a table of these cells. As shown in Figure 1, a vehicle agent is an individual element, which travels from one location to another via a route with dynamic velocity V through a list of cells along the roadway network to destination. Figure 2 shows a snapshot of the roadway network in the simulation system for a housing estate.

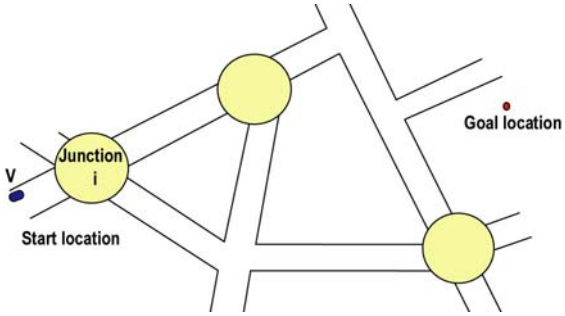


Figure 1. Description of a roadway network system

Figure 2. A screen capture of the roadway network in the simulation system

Traffic congestion and uncertainty will occur when traffic demand is higher than usual supply, which brings inefficiency to all road users. During the simulation, the state of each cell is updated according to the traffic flow. A cell is registered as occupied when there are vehicle in the cell. Otherwise it is registered as an empty cell. There will be no difference for the vehicle's exact position to the road situation as long as the vehicle is in the area defined by the cell configuration at the period of interest. The network configuration for a particular urban environment is loaded from the configuration map of the network at the initialisation stage.

3. PATH PLANNING

Based on an origin-destination matrix for the period of interest, the system is capable of providing an accurate description of the expected traffic flow using real-time A* path-planning algorithm, which estimates route choices of the vehicle agents.

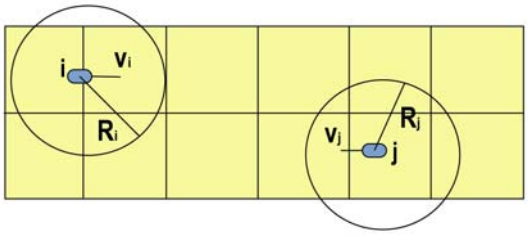


Figure 3 Description of the perception of virtual vehicle agents: Vi – velocity of vehicle i, Ri-perception radius of i, Vj- velocity of vehicle j, Rj-perception radius of j.

The average travel time for a vehicle traversing a specific route can be calculated. Upon arrival of an individual vehicle, the traffic stream is split to follow routes according to the cost value of the chosen route. The vehicle will choose the least cost route to travel. Each vehicle in the system has been assigned a circular vision zone representing the views of

mirrors and of the front screen.

At time *t*, the position of a vehicle moving along a roadway is designated $x_0(t)$, then its velocity is $dx_0(t)/dt$, and its acceleration $d^2x_0(t)/dt^2$. With N vehicles in the system, there are N different velocities, each depending on time, and the velocity of each vehicle is designated $v_i(t)$, i=1,…,N. At time *t* and position *x*, the velocity of the vehicle i can be expressed as a point in space of a velocity field:

$$v_i(x_i(t),t) = v_i(t) \qquad (1)$$

In order to measure the traffic, the traffic flow *q*, which is the average number of vehicles passing per lane per unit time, say per hour, is defined. The number of vehicles in a given length of roadway, which might be converted into the number of vehicles per unit length (per lane), is called the density of vehicles *p*. The general relation of the three variables is:

$$q = p \cdot v \qquad (2)$$

The dynamics of an individual moving vehicle must follow the basic laws of motion, which may be represented as a set of general ordinary deferential equations with delay. The behaviour of each vehicle is modelled in relation to the vehicle ahead. This is refereed as car-following models []. This model captures individual driver's reaction to the road conditions ahead. Although drivers will decelerate and accelerate quite differently, the driving behaviour also depends on the distance to the preceding vehicle. The closer the driver is, the more likely the driver is to respond strongly to an observed relative velocity. Therefore, the sensitivity is inversely proportional to the distance ahead.

$$\frac{d^2x_n(t+T)}{dt^2} = c \cdot \left(\frac{dx_{n+1}(t)}{dt}\right)^m \cdot \frac{\left(\frac{dx_n(t)}{dt} - \frac{dx_{n-1}(t)}{dt}\right)}{[x_n(t) - x_{n-1}(t)]^l} \qquad (3)$$

where *T* is the reaction time between stimulus and response, which summarizes all delay effects such as human reaction time or time the vehicle mechanics need to react to input, *c* is a chosen constant, which can be interpreted as the velocity corresponding to the maximum density p_{max}. The constant *m* describes the perception-reaction time and various responses to the movement of other vehicles including individual driving attitudes, while constant *l* expresses the road conditions and other related factors. Important human performance characteristics can therefore be added into the formulation although no clear solution has yet been found. We consider two general cases. First, Equation 3 applies for high densities of a roadway. Second, Equation 4 is for small densities. Because for small densities of a roadway, the speed changes of a vehicle are more likely influenced by the speed limit, say 30 miles per hour for residential areas. Therefore, $v = v_{max}$, and the velocity at the maximum flow is:

$$v\left(\frac{p_{\max}}{e}\right) = c \qquad (4)$$

The system updates the internal state of each agent of the network consistently. Agents are maintained by Finite State Machines (FSMs). At each simulation frame, the state of an agent has a specific set of outputs.

4. SIMULATION RESULTS

The traffic management module of the system carefully controls the traffic flows for vehicles both entering and leaving a roadway of the network. If the flow exceed the designed capacities of that roadway, congestions and delays are introduced. Figure 4 shows the virtual vehicles agents arriving at a roundabout.

Figure 4. Vehicles approaching a roundabout

The design of the network must take into account the congestion and queuing at junctions. The system uses the average number of arrivals per unit time for each junction. There is a maxim queue length beyond which the change of traffic signal is enforced. It is desirable for junctions of the network to be in equilibrium so that the entering and leaving of the traffic flow of a roadway are the same. This process can be summarised by Figure 5 for a traffic light controlled junction.

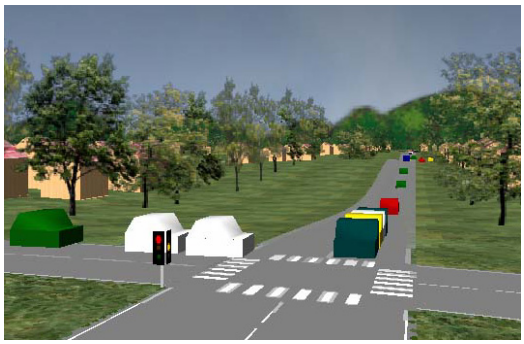


Figure 5. A queuing system of a traffic light controlled roadway.

Traffic jam occurs when the flow of input is greater the throughput.

Figure 6. Traffic jam at roundabout during morning rush hours of a residential housing estate

It is very common in reality that there are vehicles waiting at traffic lights and roundabouts. A long-lived queue is considered as congestion, if the delay time of a vehicle at a junction is greater than expected Figure 6 demonstrates a traffic jam at a roundabout of a residential area in the morning rush hours.

Cells of the roadways are used as sensors for detecting congestion in different regions of the network. Information will be posted to inform the following vehicles on the same roadway of the congestion. These vehicles would change routes by re-planning a path from their current position to destinations.

As the behaviour modelling in our simulation is an individual agent based approach, it enables good capture of stochastic characteristics of traffic in reality. For a period of interest, the system subdivides the time period into short measuring intervals.

In ambient traffic situations, a roundabout is assumed more efficient than a traffic light controlled junction if the design of other sections of the roadway remains the same. However, in an ill-designed roadway network, a roundabout can cause traffic congestion very quickly in heavy traffic. The simulation for the large residential area as shown in above figures runs at 13 to 15 frames per second for 500 moving vehicles, which is near real-time visualisation.

5. CONCLUSION AND FUTURE WORK

Using a computer animation approach, we have simulated a large complex roadway network with different designs and configurations in a 3D visualisation of a housing estate. Users of the system can populate the network during the simulation to test the roadway traffic flow. Users can set the origin -destination matrix of the vehicle agents. Timing of the traffic signals can also be altered in the event of a traffic jam. New network design and alterations of junctions including the type and control management of the junctions can be loaded from a design map before a simulation. We are able to simulate a large number of vehicles in the environment effectively with different traffic situations and conditions, and the results agree well with the reality of similar situations.

6. REFERENCES

[van Katwijk and van Koningsbruggen 2002] R. T. van Katwijk and P. van Koningsbruggen. Coordination of traffic management instruments using agent technology. Transportation Research Part C: Emerging Technologies, 10(5-6): 455-471, 2002

[Metoyer and Hodgins, 2000] R. A. Metoyer and J. K. Hodgins. Reactive Pedestrian Path Following From Examples. In Proceedings of 16th International Conference on Computer Animation and Social Agents 2000. pages 149-156. May 8 – 9, New Brunswick, New Jersey.

[Nagel and Schreckenberg 1992] K. Nagel and M. Schreckenberg. A cellular automaton model for freeway traffic. Journale de Physique, 1(2):2221–2229, December 1992

[Reeves 1983] W. T. Reeves. Particle systems – a technique for modelling a class of fuzzy objects. Computer Graphics, 17 (3): 359-376, 1983

Towards reverse design of freeform shapes

J.S.M. Vergeest,
R.Langerak, Y.Song,
C. Wang
Faculty of Industrial Design Engineering
Delft University of Technology
The Netherlands, Delft
j.s.m.vergeest@io.tudelft.nl

W.F. Bronsvoort
P.J. Nyirenda
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
The Netherlands, Delft

ABSTRACT

The need for more intuitive, faster and more effective tools for freeform product design is still an outstanding research issue in shape modeling. We propose a new methodology in which the designer can define optimal shape modification tools for the situation at hand. The key to this method is a dialogue between the designer and the computer, in which the details of the requested shape modifier are settled. The proposed tool, called user-defined modifier (UDM), is based on recent techniques from freeform shape recognition and parameterized, template-controlled shape modification. The dialogue between user and the system is described, and the basic techniques for the UDM tool are presented as well.

Keywords
Freeform shape design, freeform features, parameterization, reverse design

1. INTRODUCTION

In the past decade effective freeform modeling tools have emerged. The *creation* of freeform product shapes is relatively well supported by current CAID (Computer-Aided Industrial Design) systems, commonly based on a workflow in which surfaces are defined from planar or 3D curves, and surfaces are subsequently synthesized into a shape model. However, any deviation from this *forward design* paradigm is much less supported, or even unsupported at all, forcing the designer to redo a significant amount of work. A general, effective method for freeform modification is still not available.

The main assumption behind the requirements for such method is the ability of the designer to define a shape modification in terms of displacements relative to the current shape. Recent studies have revealed that for the exploration of freeform shape concepts, designers are able to express intended shape modifications in a natural way by putting displacement vectors in the effect region [Coo02a].

The main research challenge of the proposed tool is the trade-off between user requirements and technical requirements. A sophisticated solution to meet the user requirements may imply a high complexity of the model and even a degradation of the model into low-level geometry.

WSCG SHORT papers proceedings, ISBN 80-903100-9-5
January 31-February 4, 2005
Plzen, Czech Republic.

Reverse design versus forward design

Forward design is a common workflow of design, where a model is built up from low-level elements to features and parts, which are assembled into a product model. If a strict work plan of the design process exists, this workflow will be efficient. However, the workflow is not supportive to freeform shape modeling decisions in any order other than the sequential one prescribed by the workflow. It does support pure synthesizing of shape by combining elements and features, where constraints within and among the features may be defined during the building process. Also

refinements and features can be included in later stages.

Reverse design of freeform shapes, on the other hand, supports referring back to previous designs. Existing shapes or features can be extracted and be inserted into a model, or otherwise reused for the creation of a new design. Here it is essential to note that the existing features might *not be designed as such*, but are perceived as an entity by the designer. In addition, the designer might expect that the feature possesses parameters that he/she can control, whereas such parameters were never defined. An important aspect of reverse design is therefore the interactive assignment of complex controls to shapes or features. These controls are needed by the designer to achieve shape modifications, which could be very situation dependent. The interactive assignment can be dependent on, or expressed in terms of, for example, characteristic points or curves in the shape under construction or in any other existing shape. This referring to features which were not designed as such is one form of reverse design.

This paper presents a new approach towards a tool, called user-defined modifier (UDM), that meets the 6 requirements. A formal problem statement will be provided in Section 3. Whereas the problem formulation applies to classical contexts, a target application is specified in Section 4. In Section 5 the dialogue and workflow concerning the reverse design of styling lines will be presented. In the Conclusions in Section 6 it will be pointed out how recent techniques can form the basis for the implementation of a UDM.

2. RELATED WORK

Forward design

In the prismatic/spherical/cylindrical shape domain, the requirement of high-level parameterization of geometry was early recognized and led to the introduction of features [Sha95a]. Most commercial solid modelers have adopted the concept of feature, and typically a user follows a workflow in which parts are created in a sequence of operations, such as extrusions, intrusions, Boolean operations and parameter setting.

A big contrast between regular-shaped features and freeform features is that the latter can hardly be predefined generically, but should evolve in a specific design context and be customized consequently. This seems to require a different design workflow than the one known for mechanical feature-based design. Yet it has been shown in [Mit00a] that an object-specific feature anatomy enables accurate performance predictions of the product design. In Mitchell's method, a tennis racket design is generically defined as a system of features, with predefined types of relationships. In general, however, the principle of aggregation of feature instances as to form the entire design model seems too restrictive. This may explain the considerable attention that has been paid to the development of detail feature modeling techniques, as for example by [Cav95a], [Els98a] and [Per02a].

Reverse design

In reverse design, an existing portion of a shape model is regarded as a starting point for shape modification. The user may, however, expect shape handles to be available which were never defined for the selected portion.

In current techniques for reverse engineering of shapes, freeform shapes can be extracted from geometrical data, where the shapes are represented as surface descriptions, typically B-splines. A B-spline surface can be regarded as a geometric object with parameters, where the parameters are the control points of the B-spline surface. It is clear, however, that "recognition" of a shape portion as a B-spline surface is not reverse design; the designer will normally expect higher level shape handles than the control points are.

3. PROBLEM STATEMENT

We formally state the problem of supporting reverse design in the freeform domain as follows.

Given: A shape S in 3D space. We assume that S is all or part of the boundary of a compact subset of 3-space and can be considered as a 2-manifold. There are a number of implicit control elements (points, curves or surfaces) e_i, i=1, ..., n associated to S. The e_i are not explicitly represented and therefore called implicit. The e_i have significance for the user as a reference object to control a shape modification. We assume that there exists a derivation procedure D that derives the elements e_i from the given shape S, so that $\{e_1, ..., e_n\} = D(S)$. D specifies the association between the shape and the implicit control elements. For each e_i there is a target element t_i. Each t_i specifies an intention of the designer concerning the shape modification.

Wanted: A shape S' such that $D(S') = \{t_1, ..., t_n\}$. Furthermore we need for i=1, ..., n a control function f_i of a real variable a_i such that $f_i(0) = e_i$, $f_i(1) = t_i$, where f_i produces a smooth transition from e_i to t_i when a_i increases from 0 to 1. Instead of [0,1], any other parameter interval could be chosen.

Although the formal problem statement is dedicated to reverse design in the freeform domain, the basic principle can be clarified in the prismatic domain, where the e_i and t_i happen to be explicitly defined. The problem statement is even relevant for

forward design. Below we will, for didactic reasons, provide an example situation to which the problem statement applies.

4. TARGET APPLICATIONS

A more complicated example can be given in case of car body development. A styling line is a contiguous set of points of high curvature in the car body's surface. Although the styling line may never have been explicitly constructed as a curve, the styling line is generally perceived as a curve. Let us denote this curve as s. Let us assume that the designer wants to achieve a shape modification local to s. He/she selects a point p_j on s. If we determine the intersection curve e_i of the car body surface with a plane containing p_j, perpendicular to the direction of s at p_j , then the curvature of e_i will have a maximum in p_j, due to the definition of the styling line. Suppose that the designer intends to reduce the sharpness of the entire styling line s. He/she can specify this intent by modifying the profile of the intersection curve, which is now a controlling element, e_i to a curve t_i which is less sharp near p_i. Formally, the shape of the car body should now modify in such a way that the new intersection curve at p_j becomes consistent with t_i. Moreover, the surface should not be modified at point p_j only, but everywhere along the styling line s. The modification specified at point p_j should be applied to all other points of s, because that is typically what the designer expects. The modification specified at point p_j is thus a kind of exemplar of the modification of the surface shape. Obviously, the exemplar does *not* specify a unique modification of the surface. However, some "most likely" modifications can be implemented and be offered to the designer as options. It should be noted that the amount of curvature reduction can be represented by a continuous parameter a_i.

5. TYPICAL WORKFLOW

The whole process is illustrated with Figures 1 to 4. The surface of the computer mouse in figure 1 appears to have a rounded edge (indicated by arrows) which could be called a styling line. It should be emphasized that "line" is a terminology only; it is neither known nor assumed that the design of the shape of the mouse contains or is based on a line or curve.

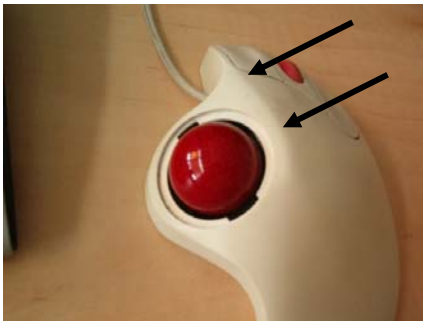


Figure 1 Computer mouse and perceived styling line indicated.

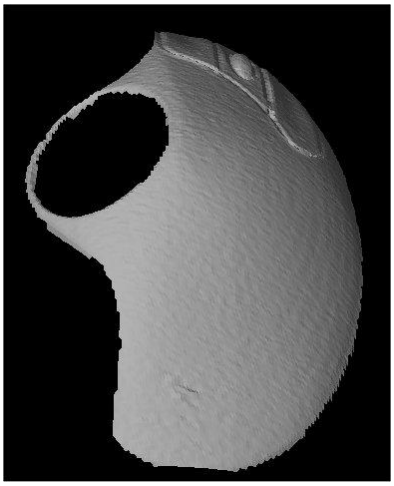


Figure 2. Surface mesh *S* obtained after 3D scanning of the mouse.

The visibility of the styling line is obviously depending on the positions of the light sources, the viewing direction etc. Suppose that for aesthetic, ergonomic or other reasons, the shape should be redesigned in such a way that the styling line would be "more rounded" (or oppositely "sharper"). The intent is then to leave the shape and the location of the styling line unchanged and to make the transition between the surface parts on both sides of the styling line smoother (or sharper). If the shape were designed with CAD, using a curve forming the connection between the two surface parts, then the redesign could have taken place by editing the geometric elements of the model. However, if the model would have been created differently, or if no CAD model is available at all, then the "standard" reverse engineering method could be applied to obtain a solid or surface model from a 3D digitization from the physical shape.

In our proposed methodology, the user designates the styling line on the surface mesh (Figure 2) obtained from 3D digitization. The curve following the locus of points at maximal curvatures is shown in Figure 3. A number of planes

perpendicular to the curve are generated to determine the intersection profiles associated with the styling line (Figure 4). These profiles constitute the elements e_i which the user can manipulate.

6. CONCLUSIONS AND ONGOING RESEARCH

Locally a styling line can be characterized by the intersection curve of a plane with shape S. Along the styling line this intersection profile may vary, but in general the variation will be moderate, otherwise the styling line would not have been perceived as such. Therefore, the various profiles will have something in common, which helps to recognize them and the similarity also simplifies the user interaction with the UDM system: a modification made to one profile should be applied to all profiles.

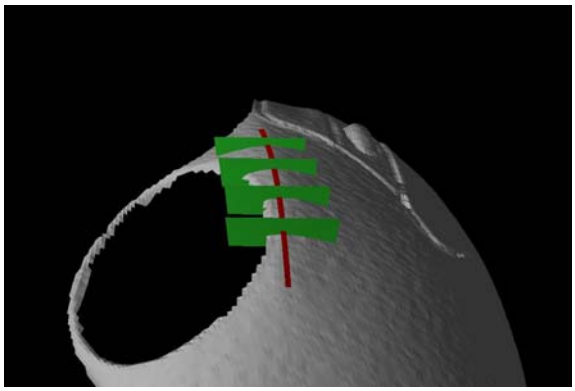


Figure 3. Intersection profiles can be generated using planes perpendicular to the curve approximating the styling line.

However, much of the reverse design tool UDM still requires theoretical investigation and algorithm development.

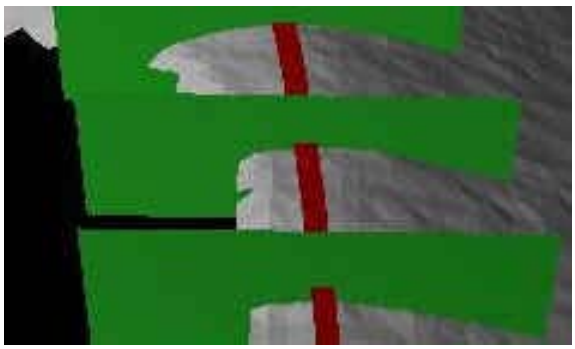


Figure 4. A closer look at the intersection between plane and surface mesh.

The styling line tracking have been developed based on the marching template method. Tthe adaptation of the shape S to the modified profiles t_i can be based on template-based deformation [Son04a).

REFERENCES

[Cav95a] J.C. Cavendish (1995), "Integrating feature based surface design with free form deformation". Computer-Aided Design, Vol. 27, Nr. 9, pp 703-711.

[Coo02a] T. Coomans and M. Roukema (2002), "Shape alterations for freeform models". Technical Report, Delft University of Technology, December 2002, **http://www.tudelft.dynash.nl**.

[Els98a] P.A. van Elsas and J.S.M. Vergeest (1998), "Displacement feature modelling for conceptual design". Computer-Aided Design, Vol. 30, Nr. 1, pp 19-27.

[Mit00a] Mitchell, S.R., Jones, R. and Catchpole, G.(2000), "Modelling a Thin Section Sculptured Product Using Extended from Feature Methods". Journal of Engineering Design , 11(4), pp 331-346.

[Sha95a] J.J. Shah and M. Mäntillä, "Parametric and feature-based CAD/CAM". New York, John Wiley, 1995.

[Per02a] J.P. Pernot , S. Guillet, J.C. Léon, C.E. Catalano, F. Giannini, B. Falcidieno , Ť A shape deformation tool to model character lines in the early design phases ť, Shape Modeling Int. Conf., Banff, Canada, pp. 165-173, 2002.

[Son04a] Song Y. Vergeest J.S.M. and Spanjaard, S (2004)., Fitting and Manipulating Freeform Shapes by Extendable Freeform Templates, Proceedings of the International Conference on Shape Modeling and Applications, Edited by Franca Giannini and Alexander Pasko, IEEE, pp 43-52

Minimization of the mapping error using coordinate descent

Gintautas Dzemyda	Jolita Bernataviciene	Olga Kurasova	Virginijus Marcinkevicius
Institute of Mathematics and Informatics	Institute of Mathematics and Informatics	Institute of Mathematics and Informatics	Institute of Mathematics and Informatics
Akademijos St. 4	Akademijos St. 4	Akademijos St. 4	Akademijos St. 4
Vilnius 08663, Lithuania	Vilnius 08663, Lithuania	Vilnius 08663, Lithuania	Vilnius 08663, Lithuania
Dzemyda@ktl.mii.lt	JolitaB@ktl.mii.lt	Kurasova@ktl.mii.lt	VirgisM@ktl.mii.lt

ABSTRACT

Visualization harnesses the perceptual capabilities of humans to provide the visual insight into data. Structure preserving projection methods can be used for multidimensional data visualization. The goal of this paper is to suggest and examine the projection error minimization strategies that would allow getting a better and less distorted projection. The classic algorithm for Sammon's projection and two new its modifications are examined. All the algorithms are oriented to minimize the projection error because even a slight reduction in the projection error changes the distribution of points on a plane essentially. The conclusions are made on the results of experiments on artificial and real data sets.

Keywords

Multidimensional data, visualization, Sammon's mapping, mapping error, coordinate descent

1. INTRODUCTION

Objects from the real world are frequently described by an array of parameters (multidimensional data). Data understanding is a difficult task, especially when it refers to a complex phenomenon that is described by many parameters. One of the ways in analyzing data is visualization. It involves the constructing of a graphical interface that enables to understand complex data. Visualization is also used to display the properties of data that have a complex relation – possibly patterns not obtainable by the current computation methods.

In this paper, we discuss visualization of multidimensional data by using structure preserving projection methods. These methods are based on the idea that the multidimensional data points can be projected on a lower dimensional space so that the structural properties of the data are preserved as faithfully as possible. Examples of such techniques are principal component analysis [Tay03a], multidimensional scaling [Kas97a], [Bor97a], Sammon's mapping [Sam69a], and others.

This paper deals with Sammon's mapping. Sammon's mapping comes from the area of multidimensional scaling. The only difference between both methods is that the errors in distance preservation are normalized with the distance in the original space. Because of the normalization, the preservation of small distances will be emphasized [Kas97a]. The analysis of relative performance of the different algorithms in reducing the dimensionality of multidimensional vectors, starting from the paper by Biswas [Bis81a], indicates Sammon's projection to be still one of the best methods of this class (see also [Fle97a]) and finds new applications (see, e.g. [Dze01a]). When visualizing the multidimensional data using the nonlinear projection, the projection errors are inevitable. The goal of this paper is to suggest and examine the projection error minimization strategies that would allow getting a better and less distorted projection.

Conference proceedings ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. STRATEGIES FOR PROJECTION ERROR MINIMIZATION

Let us have s objects (points, vectors) $X_i = (x_1,...,x_n)$, $i = 1,...,s$ from an n-dimensional

space R^n. The aim of this method is to find *s* points in an *m*-dimensional space ($m < n$, usually *m*=2) $Y_i = (y_1,...,y_m) \in R^m$, $i = 1,...,s$ so that the corresponding distances of *m*-dimensional points approximate the original ones as well as possible. Sammon's mapping [Sam69a] is one of such methods. It tries to minimize the projection error:

$$E_s = \frac{1}{\sum_{\substack{i,j=1 \\ i<j}}^{s} d_{ij}^*} \sum_{\substack{i,j=1 \\ i<j}}^{s} \frac{(d_{ij}^* - d_{ij})^2}{d_{ij}^*} \quad (1)$$

Here d_{ij}^* is the distance between two *n*-dimensional points; d_{ij} is the distance between two points in the two-dimensional space. Even a slight reduction in E_s changes the distribution of points on a plane essentially. This proves the necessity to make every effort for minimizing the distortion of projection E_s. In this paper, three strategies for projection error minimization are investigated: (1) classical Sammon's algorithm [Sam69a] (**S1**); (2) applying the Seidel coordinate descent for Sammon's method (**S2**); (3) applying the noise for **S2** method (**S3**).

Classical Sammon's algorithm (S1). In this method, the coordinates y_{ik}, $i = 1,...,s$, $k = 1, 2$ of two-dimensional vectors $Y_i = (y_{i1}, y_{i2})$ are computed by the iteration formula:

$$y_{ik}(m'+1) = y_{ik}(m') - a \frac{\partial E_s(m')}{\partial y_{ik}(m')} \Big/ \frac{\partial^2 E_s(m')}{\partial y_{ik}^2(m')}$$

Here m' denotes the iteration order number; a is a step length, also called a „magic factor", because the obtained projection error depends on it.

One iteration of the algorithm contains calculations, where both components of all the points Y_i, $i = 1,...,s$ are recalculated. These components are recalculated taking into account the coordinates of Y_i, $i = 1,...,s$, obtained in the previous iteration.

Seidel-type coordinate descent for Sammon's mapping (S2). Seidel-type coordinate descent method is used for solving linear equation systems and in optimization [Kar03a]. We suggest applying coordinate descent for Sammon's mapping. The coordinates of two-dimensional vectors Y_i are recalculated, taking in to consideration not only the coordinates, obtained in the previous iteration, as in classical Sammon's algorithm, but also the new coordinates, obtained in the current iteration: the coordinates $y_{jk}(m'+1)$, if $j = 1,...,i-1$, and $y_{jk}(m')$, if $j = i+1,...,s$.

The coordinate descent method with noise (S3). It has been noted that the 1st derivatives of E_s are smooth enough in algorithm **S1** (Fig. 1), but the 2nd derivatives are alternating (Fig. 2). Iris data set is analyzed. The 1st and 2nd derivatives of E_s were measured in each iteration, when new coordinates of the 4th data point were determined. Numerous experiments allowed us to see that the disperse of two-dimensional points from the line (initialization rule of two dimensional points) increases with an increase in fluctuation of the 2nd derivatives. We decided to apply artificial fluctuation to the 2nd derivatives (to apply noise) in algorithm **S2.** In such a way a new algorithm **S3** has been created.

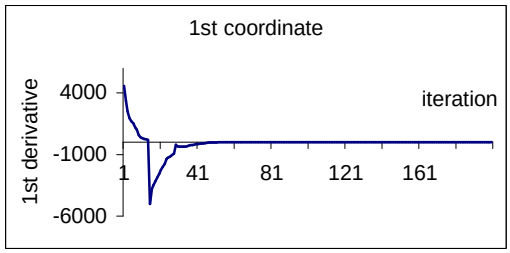


Figure 1. Fluctuation of the 1st derivative in S1.

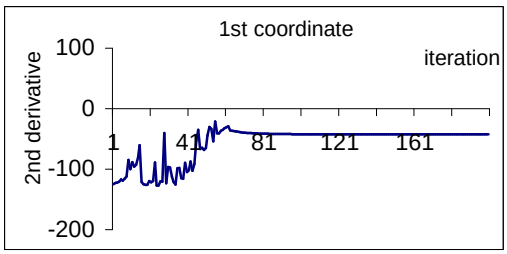


Figure 2. Fluctuation of the 2nd derivative in S1.

The first idea was to add random noise to the second derivative as follows:

$$\frac{\partial^2 E_s}{\partial y_{ik}^2} \Leftarrow \frac{\partial^2 E_s}{\partial y_{ik}^2} + \xi$$

Here x is a random number. However, the problems arose to schedule the level of noise depending on the value of the second derivative and the order number of the current iteration. Therefore, a more effective way has been found to define noise by some heuristic rule:

$$\frac{\partial^2 E_s}{\partial y_{ik}^2} \Leftarrow \frac{\partial^2 E_s}{\partial y_{ik}^2}(1 - e^{-\lambda m'})\left|\sin(\beta m')\right|, \text{ for } m' < \frac{t}{n}.$$

Here l, b are some constants, selected experimentally, t is the total number of iterations, m' is the order number of the current iteration.

3. RESULTS OF THE ANALYSIS

Experiments were carried out with real and artificial data. The dependence of E_s on different factors is

investigated on: the computing time, the number of iterations, and the value of "magic factor" a .

Data for analysis. Artificial data sets:

1. Uniformly distributed data: 100 10-dimensional points generated at random in the interval [-1; 1];
2. Uniformly distributed data: 500 points generated at random, as No1.
3. Clustered data: ten 10-dimensional points are generated at random; in the area of each point, nine 10-dimensional points are generated by normal distribution.

The experiments have been repeated for 100 times with different sets of 10-dimensional vectors generated as given above (data sets No1-No3). The average results have been calculated.

Real data:

- The classical Fisher iris data set [Fis36a].
- The Wood data set [Dra66a].
- The HBK data set [Haw84a].

Dependence of the projection error on computing time

The advantages of algorithm **S3** in comparison with algorithm **S1**, **S2** have been shown by analyzing data No2 ($\alpha = 0.25$). A lower projection error and its faster convergence to optimal value have been obtained using algorithm **S3**. In order to get a lower value of E_s , it suffices to perform less iterations, i.e. the computing time is saved (Fig. 3).

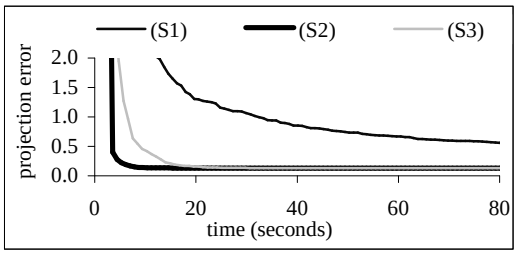


Figure 3. Dependence of the projection error on time.

Difference between projection errors in the whole iteration process of algorithms **S2** and **S3** is rather small, but in most cases, the final results obtained by **S3** are better (see Table 1)**.**

Dependence of the projection error on the "magic factor"

While examining the dependence of projection error on the value of a , artificial data No1 are analyzed with different values of a (0.1; 0.11;…;1.45; 1.5). In Fig. 4, the mean value of the projection error is presented. It is shown that dependence on the "magic factor" a is less using the Seidel-type coordinate descent method (**S2**). Applying noise (**S3**) does not influence the dependence on a .

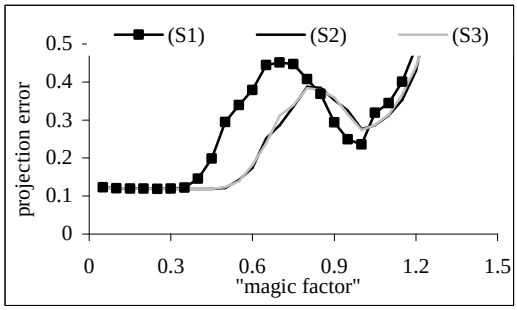


Figure 4. Dependence of the projection error on the value of "magic factor".

Strategies of E_s minimization			Data
(S1)	(S2)	(S3)	
0.1209452	0.1201307	0.1203316	artificial (No1)
0.0710200	0.0711317	0.0695346	clusters
0.0058476	0.0045259	0.0040088	Iris
0.0243263	0.0257550	0.0256691	Wood
0.0112111	0.0113962	0.0049657	HBK

Table 1. Projection errors

Analysis of mappings

The minimal projection errors have been found for each real data set. After examining the artificial data No1, the average values of minimal projection errors have been calculated over 100 experiments ($\alpha = 0.25$) (Table 1). The largest distortion of projection has been obtained using by algorithm **S1** in all the cases. The smallest projection error has been obtained using algorithm **S3** in most cases.

In Figures 5-6, the mappings of the real data are presented. They have been obtained using algorithms **S1** and **S3** that gives the smallest projection error with the real data (see Table 1). The figures show that, if a smaller projection error is obtained, the preserving data structure is more precise. When analyzing the iris data, three flower types are separated more precisely by using algorithm **S3** (see Fig. 5b); when investigating the HBK data, classical Sammon's mapping (**S1**) is able to separate point groups (see Fig. 6a), but all the three groups are separated more exactly when algorithm **S3** is used (see Fig. 6b).

4. CONCLUSIONS

In this paper, the new opportunities for minimizing the multidimensional data projection error E_s (1) are suggested and examined experimentally. Classical Sammon's mapping **S1** is compared with two new algorithms **S2** and **S3**.

Smaller projection errors are usually obtained by using algorithms **S2** and **S3** compared with the errors by classical Sammon's algorithm **S1**. Another advantage of the new algorithms is that small errors are obtained after a smaller number of iterations and

sooner. Therefore, the projections of multidimensional data on a plane are more faithfully; lower dependence of the projection quality on the value of a is obtained.

Applying noise to the 2nd derivatives in first iterations of the projection error minimization process (algorithm **S3**) speeds up the moving of the initial two-dimensional points from the line. This improves the quality of visualization, because the smaller projection errors are obtained.

Finally, the discovered new ways for minimizing the projection error allow a better perception of the Sammon-type projection and make a basis for further research.

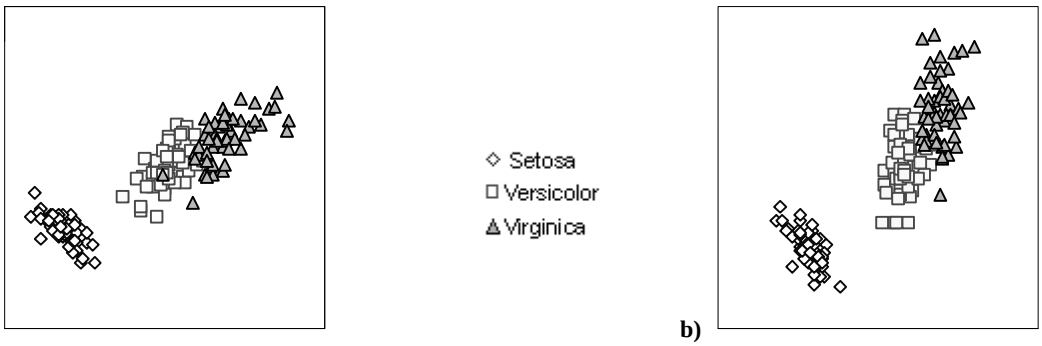


Figure 5. Projections of the iris data: a) S1 ($E_s = 0.0059$); b) S3 ($E_s = 0.0040$).

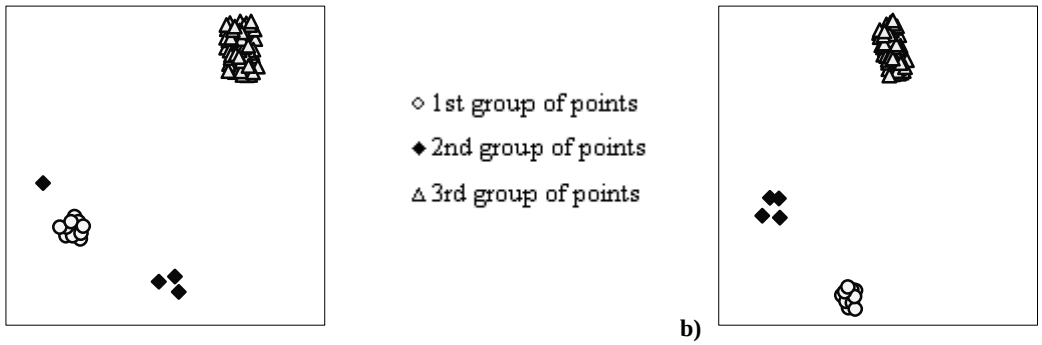


Figure 6. Projections of the HBK data: a) S1 ($E_s = 0.0112$); b) S3 ($E_s = 0.0050$).

5. REFERENCES

[Bor97a] Borg, I., and Groenen, P. Modern Multidimensional Scaling: Theory and Applications, Springer, New York, 1997.

[Bis81a] Biswas, G., Jain, A.K., and Dubes, R.C. Evaluation of projection algorithms, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 3(6), pp. 701-708, 1981.

[Dra66a] Draper, N. R., and Smith, H. Applied Regression Analysis. John Wiley and Sons, New York, 1966.

[Dze01a] Dzemyda, G. Visualization of a set of parameters characterized by their correlation matrix. Computational Statistics and Data Analysis, Vol. 36(10), pp. 15-30, 2001.

[Fis36a] Fisher, R. A. The use of multiple measurements in taxonomic problems. Annals of Eugenics, No.7, pp.179-188, 1936.

[Fle97a] Flexer, A. Limitations of self-organizing maps for vectors quantization and multidimensional scaling. In M.C.Mozer, M.I.Jordan and T.Petsche(eds.), Advances in Neural Information Processing Systems 9, Cambridge, MA: MIT Press/Bradford Books, pp.445-451, 1997.

[Haw84a] Hawkins, D. M., Bradu, D., and Kass, G. V. Location of several outliers in multiple regression data using elemental sets. Technometrics, No.26, pp.197–208, 1984.

[Kar03a] Karbowski, A. Direct method of hierarchical nonlinear optimization - reassessment after 30 years. Proceedings of III International Conf. on Decision Support for Telecommunications and Information Society (DSTIS 2003), Warsaw, pp.15-30, 2003.

[Kas97a] Kaski, S. Data Exploration Using Self-Organizing Maps, PhD thesis, Helsinki University of Technology, Department of Computer Science and Engineering, 1997, http://www.cis.hut.fi/~sami/thesis/

[Sam69a] Sammon, J. W. A nonlinear mapping for data structure analysis. IEEE Transactions on Computers, No.18, pp.401-409, 1969.

[Tay03a] Taylor, P. Statistical methods. In M. Berthold and D. J. Hand (eds.), Intelligent Data Analysis: an Introduction, Springer-Verlag, pp. 69-129, 2003.

Managing Data Flow in Interactive MR Environments

Patrick Dähne Helmut Seibert
Computer Graphics Center (ZGDV)
Fraunhoferstraße 5
D-64283 Darmstadt, Germany
{patrick.daehne,helmut.seibert}@zgdv.de

ABSTRACT

In this paper the concept and design of a software framework which provides a transparent data flow for interactive Mixed Reality (MR) applications is discussed. The design was affected by our demands on platform independency, simplicity, network transparency, maximum performance and availability of runtime debugging facilities. Our software framework tries to simplify the development of MR applications by using the concept of a data flow graph. The developer builds such a graph from a library of small software components that communicate via the edges of the graph.

Keywords

Mixed Reality, Augmented Reality, Virtual Reality, Device Management, Tracking, Interaction.

1. INTRODUCTION

In our daily work on various Mixed Reality applications, we made the experience that we usually spend most of our time on implementing code that handles communication with hardware devices and software components provided by our project partners, instead of concentrating on the application itself. This paper describes the result of our efforts to minimize the work we have to do when implementing the "infrastructure" of our application.

The base of our MR applications is our self-developed rendering framework "Avalon" [3] that utilizes VRML/X3D. An interesting feature of VRML is that it does not only allow to describe graphical objects and animations, but also to specify the interactions that are possible with these objects. In fact, VRML allows to write huge parts of applications by using standard VRML nodes, JavaScript or Java, instead of doing tedious C++ compiler sessions.

Unfortunately, VRML has quite limited capabilities to integrate hardware devices and to communicate with other software components. In this paper we describe how we extended our rendering framework by an sophisticated device management system. The fundamental idea is to create a library of small software modules that handle specific tasks, for example operating devices or transforming data values provided by devices, and to assemble these modules into a data flow graph.

WSCG SHORT papers, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. RELATED WORK

The main research focus in the area of Mixed Reality has been the determination of the user's position and orientation (tracking), appropriate interaction methods for MR applications, graphical problems like the correct illumination of virtual objects and the occlusion of virtual objects by real objects, and the development of powerful and yet lightweight wearable computers. But recently, the development of MR software frameworks gained more and more attention.

An early example of a software framework is COTERIE [6]. COTERIE is a set of packages written in Modula-3 that allow to create "Distributed shared objects", i.e. data objects that are transparently shared by processes on different machines in the network.

Another well-known software-framework is Studierstube [9]. Studierstube is based on the concept of a distributed scene graph (Distributed Open Inventor), and a

data flow framework called OpenTracker [8] that handles different kinds of trackers.
DWARF [2] is using a concept similar to our own concept described in this paper. AR applications are build by using services that form a distributed data flow graph. CORBA is used for all network communication.
Tinmith [7] is using an object store based on Unix file system semantics. Applications can register callbacks on these objects to get notifications when an object changes. Objects are automatically distributed between different processes on the network.
Other software frameworks that focus on the development of VR applications are VR Juggler [4] and DIVERSE [1]. VRPN [11] and IDEAL [5] are device management systems that concentrate on the flexible and network-transparent access to VR devices.

3. SYSTEM DESIGN

Our concept of an MR framework is influenced by the design of VRML. In VRML, each node of the scene graph is a small status machine that receives events by inslots, changes its status according to the event, and sends events to outslots. Outslots and inslots are connected by ROUTEs. This architecture in fact establishes a second graph besides the scene graph, a data flow graph that contains large parts of the application logic.
The advantage of VRML over applications written in C++ or any other compiled language is that it is completely system independent. The same VRML world can be viewed on different operating systems, on low-end desktop PC's and high-end graphic workstations that operate immersive stereoscopic projection systems.
For our MR framework, we chose an approach that is strongly influenced by the VRML data flow graph and therefore has similar attractive properties. The main task of our MR framework is to handle input and output devices efficiently, because this is an area that is not handled by VRML. But besides simply allowing access to devices, our MR framework should also handle all preprocessing of the data provided by the devices. For example, it should not only allow the application to grab video frames from a web cam, but also to determine the position and orientation of the user by using these video frames (i.e. video based tracking).
Our intention is to create a library of small software modules that are specialized on simple tasks. Some software modules act as device drivers, i.e. they produce or consume data streams. Others act as filters, i.e. they transform incoming data streams. The modules are nodes of a data flow graph, and they receive data from and send data to other modules via the edges of the graph. The application developer should be able to build as much as possible of his application by simply assembling these prefabricated software modules into a data flow graph, allowing him to concentrate on the application logic.

3.1. Nodes

Nodes are the core component of the data flow graph. There are four types of nodes:

1. Nodes that produce data. These are usually device drivers of input devices, nodes that replay data streams stored on a hard disk, timers etc.

2. Nodes that transform data. Examples are nodes that transform coordinate systems, or video tracking systems that take video frames and transform them to position and orientation values.

3. Nodes that consume data. These are usually device drivers of output devices, nodes that store data streams on a hard disk, etc.

4. Nodes that do not deal with data in any way. This sounds strange at a first glance, but our system currently already has three nodes of this kind: The "Network" node that makes the system network transparent, the "Web" node that provides a user interface to the framework, and the "Inline" node that allows to integrate subgraphs stored in configuration files.

To create a new node, the developer has to derive a new class from the abstract node base class. Nodes can be linked into the application executable (mostly used for the basic, application-independent set of standard nodes provided by the system), or they can be loaded as plugins during runtime (application-specific nodes).
An important design decision that has to be made by the developer of a new node is whether the node uses its own thread or not. Device driver nodes for input devices usually always need to have an own thread, because they have to listen to the port the device is connected to. All other types of nodes do not necessarily need to have their own thread. For example, filter nodes that transform position values into another coordinate system usually directly calculate the new values when they are notified that new data is available. This means that they are in fact driven by the threads of the device driver nodes that provided the incoming data values. On the other hand, filter nodes that have to perform complex calculations, like video trackers, usually get their own threads.

3.2. Outslots and Inslots

Outslots and inslots are the means used by nodes to exchange data values. Outslots are used to send data to other nodes, and inslots are used to receive data from other nodes. Both outslots and inslots are typed, i.e. you have to specify what kind of data can be send to ceived from the slot when you create it. When a

node writes data values into an outslot, the outslot automatically transfers copies of these data values to all inslots it is currently connected to. For efficiency reasons, smart pointers are used to transfer large amounts of data, i.e. copying data values into inslots usually just requires to copy a pointer.

3.3. Routes

When creating nodes of the data flow graph, the application developer has to specify a unique name for each of these nodes, e.g. “Joystick 1” and “Joystick 2” for two nodes that operate joysticks attached to the system. Each outslot and inslot of a node has a unique label as well, e.g. “Button #1” for the first button of a joystick or “X-Axis” for the x-axis. As a result, each slot in the system can be identified by a unique label consisting of the node name and the slot name, e.g. “Joystick 1/Button #1” for the first button of the first joystick or “Joystick 2/X-Axis” for the x-axis of the second joystick. To connect an outslot to an inslot, the application developer simply has to create a so-called “Route” that maps the label of an outslot to the label of an inslot. Of course, only slots sending and receiving the same data type can be connected by routes.

3.4. Configuration File

The configuration file allows to store the whole data flow graph on hard disk and to restore the graph when starting the application again. The format of the configuration file is XML due to the fact that this is a well-known, established standard for storing information.
Even though it is possible to create these XML files by hand, this approach is not recommended. Instead, the application developer uses the integrated user interface of the device management system to create a data flow graph that fulfills his needs. This user interface also allows to save the current status into a configuration file.
There are two ways to restore a saved graph from a configuration file:

1. The user simply loads the configuration by using the integrated user interface.

2. There is a special “Inline” node that allows to integrate configuration files into other configuration files, exactly the same way as the VRML “Inline” node allows to integrate several VRML subgraphs into one scene graph, or the C “include” preprocessor statement allows to integrate several pieces of source code into one file.

3.5. Network Transparency

The concept of a data flow graph makes it simple to introduce network transparency into the device management system. Nodes of the graph communicate via edges, so the most elegant solution is to allow edges to be created between nodes on different machines.
Our system does not support network transparency directly. Instead, it is made available by a special node called “Network”. We did this to achieve a clear separation between the device management and the network code.
When the Network node starts operation, it automatically connects to all other Network nodes on the network. We are using a technique called “Multicast DNS” [12] to automatically discover all Network nodes available on the network without the need of any configuration or central servers. After connecting, the Network nodes create local proxy nodes for all nodes available on remote machines. These proxy nodes can be used to create edges between nodes on different machines.
Each data value that gets transferred via a network connection needs to be transformed into a system-independent byte stream (serialization). For each data type, a corresponding codec has to be implemented that performs this serialization. Such codecs already exist for the default data types provided by the system, but the application programmer has to implement and register codecs for all application-specific data types. Most codecs simply write the data values into the byte stream, but more sophisticated codecs are possible that compress the transferred data, e.g. when transfering video frames.

3.6. User Interface

Our device management system has an integrated Graphical User Interface (GUI) that allows to modify the data flow graph during runtime. During our work on several MR projects it soon became obvious that it is quite often impossible to do development and debugging directly on the machine the application is running on. Instead, we need means to control our application from other machines in the network.
Our solution is a small web server that provides a user interface build from HTML pages. This web server is not part of the device management system, instead it is implemented as a node that can be added to the data flow graph. The web server solution allows us to control the device management system from any device that has a simple web browser installed and that is connected via network to the machine the application is running on. The interface allows to inspect the current state of the system, to add and to remove nodes and routes from the graph, to change parameters, and to save the current state to or to restore it from a configuration file.

4. APPLICATION EXAMPLE

In this section we will discuss the solution for the task of combining tracking systems which is necessary in many Mixed Reality applications. The MEDARPA Project focused on the development of a flexible medaugmented reality system supporting minimal in-

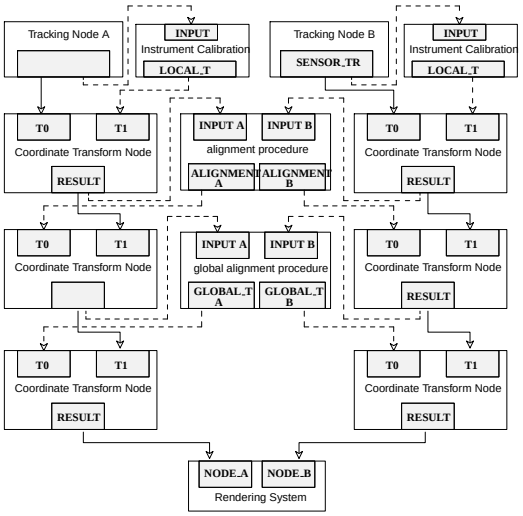


Figure 1. Graph for a composed tracking system. Dashed arrows depict event based data flow, solid arrows continuous data flow.

vasive interventions. Medical Image data and navigation support as augmentation are given on a transparent display which is mounted on a swivel arm. 6DOF tracking of the physicians head, the transparent display, patient and the instrument for the intervention is needed in order to provide the augmentations. An optical tracking system is used for the tracking of the physicians head as well as for the display position and orientation. For the tracking of the instrument an electromagnetic tracking system was chosen. These two tracking systems were combined as discussed in [10] by applying several consecutive transformations to all measurements to get the output of all tracking systems to a common coordinate system. A 6DOF transform consists of a rotation component and a translation component which are represented in a 4×4 transformation matrix and referred as T in the following. Usually the sensor of a tracking system cannot be placed exactly at the location which is to be tracked e.g. the tip of an instrument, the local transform T_l^{sens} allows to specify the location of the instruments tip in the sensor coordinate system. The alignment transform T_{tr}^{al} is needed to get the frame of reference defined by each tracking system aligned to a common frame of reference by mapping the output of one system to the frame of reference of the other system. A final transformation T_{al}^{w} allows to change the frame of reference if this is required by a special application.

For each measurement T_{sens}^{tr} reported by the tracking system, the transformation

$T = T_l^w = T_{al}^w \cdot (T_{tr}^{al} \cdot (T_{sens}^{tr} \cdot T_l^{sens}))$

needs to be calculated. The resulting transformation T can then directly be routed into the corresponding VRML Transform nodes of the scenegraph which is loaded on the Avalon rendering system. The resulting application dataflow is shown in figure 1.

5. CONCLUSION

In this paper we presented the framework we use for MR applications. It is currently implemented as a C++ library on several operating systems (Windows, MacOS X, Linux, IRIX and SunOS) and as a Java package (written in pure Java, therefore running on all systems where a Java virtual machine is available). We are currently developing MR applications based on the framework, and the experiences we gain from these practical trials propel the further advancement of the system. Our current research focus is the identification, design and implementation of new nodes that further ease the development of MR applications.

References

[1] L. E. Arsenault and J. Kelso. The DIVERSE Toolkit: A Toolkit for Distributed Simulations and Peripheral Device Services. In *VR 2002*, 2002.

[2] M. Bauer, B. Bruegge, G. Klinker, A. MacWilliams, T. Reicher, S. Riss, C. Sandor, and M. Wagner. Design of a component-based augmented reality framework. In *Proceedings of ISAR 2001*, 2001.

[3] J. Behr, P. Dähne, and M. Roth. Utilizing X3D for Immersive Environments. In *Proceedings of Web3D 2004*, pages 71–78, 2004.

[4] C. Cruz-Neira, A. Bierbaum, P. Hartling, C. Just, and K. Meinert. VR Juggler - An Open Source Platform for Virtual Reality Applications. In *Procs of 40th AIAA Aerospace Sciences Meeting and Exhibit '02*, 2002.

[5] T. Fröhlich and M. Roth. Integration of Multidimensional Interaction Devices in Real-Time Computer Graphics Applications. In *Computer Graphics Forum 19*, pages C–313 – C–319, 2000.

[6] B. MacIntyre and S. Feiner. Language-level support for exploratory programming of distributed virtual environments. In *Proceedings of UIST '96*, pages 83 – 95, 1996.

[7] W. Piekarski and B. H. Thomas. An object-oriented software architecture for 3D mixed reality applications. In *Proceedings of ISMAR 2003*, 2003.

[8] G. Reitmayr and D. Schmalstieg. An Open Software Architecture for Virtual Reality Interaction. In *Proceedings of VRST 2001*, 2001.

[9] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavari, L. M. Encarnação, M. Gervautz, and W. Purgathofer. The Studierstube Augmented Reality Project. *Presence*, 11, 2002.

[10] B. Schwald, H. Seibert, and M. Schnaider. Composing 6 DOF Tracking Systems for VR/AR. In *Proceedings of Computer Graphics International 2004*, pages 411–418, 2004.

[11] R. M. Taylor, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. VRPN: A Device-Independent, Network-Transparent VR Peripheral System. In *Proceedings of VRST 2001*, 2001.

[12] Multicast DNS, IETF draft. http://www.multicastdns.org/.

A new formulation of differential radiosity and a rendering application

D. Arquès V. Biri O. Derpierre
Gaspard Monge Institute
University of Marne-la-Vallée
6, cours du Danube
77 000 SERRIS, FRANCE
{arques,biri,oderpier}@univ-mlv.fr

ABSTRACT

Among all the existing realistic lighting methods, the radiosity method is the only one that gives precisely an analytic solution of diffuse light exchanges. The gradient of this solution have been studied but not often used in a context of pure rendering. We present in this article a method to render a surface using the radiosity contour levels. First, we define a differential formulation of the radiosity equation which leads us to a new expression of the gradient of radiosity. We deduce from this general equation a simpler equation of this gradient in the case of a planar surface lighted by a light source reduced to a point. Then we present our method to render planar surfaces using a radial mesh that follows the contour levels of the radiosity. This method is shown to improve the quality of the rendering and decrease the number of vertices used for rendering.

Keywords

Differential radiosity, Radiosity gradient, Global illumination, Rendering.

1. INTRODUCTION

The generation of photo-realistic images using just a scane description requires a precise and realist lighting model. The radiosity method, initially proposed by Goral et al. [Gor84] uses a faithful model of diffuse light exchanges. This method belongs to the global illumination methods that strive to describe the light interactions as precisely as possible. Though this approach tends to be less widely used today, it is still the only one that gives a true analytic solution of the illumination solution on the surfaces forming the virtual scene. Moreover the use of radiosity is still perfectly adequate in the case of architectural scenes where its ability to handle multiple light sources and indirect lighting allows the generation of truly amazing images.

Netherveless, the initial approach of Goral et al. requires to mesh precisely the surfaces composing the scene and to consider a constant radiosity function on every single single generated face. Therefore the computer graphics community had to elaborate different methods to solve these problems. Hierarchical methods were form the first of these answers. These methods use meshes that adapt themselves to the radiosity function and to the shadows cast on objects which reduces considerably the number of generated faces. A second answer was found in progressive algorithms that are able to compute quickly a reconstruction of the illumination function by considering first the main light transfers. But the two answers were mainly algorithmic solutions to an analytic problem. Research based on the analytic formulation of the radiosity equation finally gave birth to the higher order radiosity methods that separate the radiosity function from its geometric support by projecting it on higher order function bases.

We follow the same strategy focusing on the radiosity function. We propose a new equation of the radiosity function gradient but instead of using it to build oracles used in the refinement process of a hierarchical method, as proposed by Holzshuch et al. [Hol95], we use it for rendering. Coupled with the property of unimodality of the radiosity function we propose a meshing method using directly the shape of the radiosity function as opposed to using the geometry support of that function.

Conference proceedings ISBN 80-903100-7-9
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. A NEW FORMULATION OF THE RADIOSITY GRADIENT

Let F denote the set of faces describing a scene. The exchanges of radiosity Bi between a face i belonging to F and the rest of the scene are described by the equation [Gor84] where E_i is the emittance of the face i and ρ_i its reflectivity.

$$B_i = E_i + \rho_i \sum_{j \in F} B_j F_{i,j} \tag{2.1}$$

In the case of unoccluded polygonal surfaces, we can define the oriented contour of the surface A_i as the set β_i of its oriented edges. The form factor $F_{i,j}$ between two polygonal surfaces A_i and A_j can be expressed :

$$F_{i,j} = \frac{1}{2\pi A_i} \sum_{\vec{k} \in \beta_i} \sum_{\vec{l} \in \beta_j} \underbrace{\int_{\vec{k}} \int_{\vec{l}} \ln\left(\left\|\overrightarrow{M_i M_j}\right\|\right) dM_i \, dM_j}_{L_{\vec{k},\vec{l}}} \tag{2.2}$$

Our objective is to find an equation describing the gradient of the radiosity function. So we consider an oriented square discretization of all the surfaces describing the scene and we study the difference of radiosity between two adjacent faces i and i' compared to the radiosity of every other face j in the scene. Starting from equation (2.1) and using the form factor on polygonal contours (2.2) we obtain the following equation :

$$B_i = E_i + \frac{\rho_i}{2\pi A_i} \sum_{\vec{k} \in \beta_i} \sum_{j \in F} \sum_{\vec{l} \in \beta_j} B_j L_{\vec{k},\vec{l}} \tag{2.3}$$

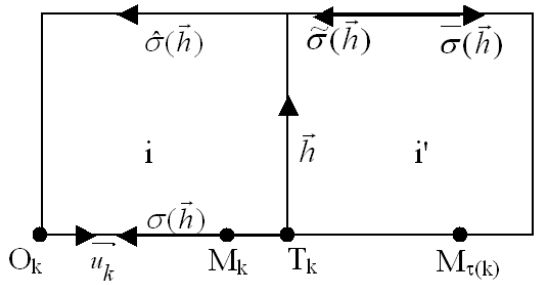


Figure 1 : Notation of a face *i* and its neighbor *i'*

We first subtract the radiosity B_i' to the radiosity B_i where i' is the neighboring face of i along the edge h. Every internal edge of the scene i is counted twice in the sum (2.3), once positively and once negatively. We group this terms together to get :

$$\begin{aligned} dB_{\vec{h}} &= B_i - B_{i'} \\ &= dE_{\vec{h}} + \frac{\rho_i}{2\pi A_i} \sum_{\vec{k} \in \beta(i)} \sum_{\vec{l} \in B} L_{\vec{k},\vec{l}} \, dB_{\vec{l}} - \frac{\rho_{i'}}{2\pi A_{i'}} \sum_{\vec{k} \in \beta(i')} \sum_{\vec{l} \in B} L_{\vec{k},\vec{l}} \, dB_{\vec{l}} \end{aligned} \tag{2.4}$$

We then separate the edges of the discretization that form the contour of the surfaces from the internal edges. Let B_e denote the first set of edges and B_i the second one. Equation (2.4) can then be rewritten as follows:

$$dB_{\vec{h}} = dE_{\vec{h}} + \sum_{\vec{l} \in B = Bi \cup Be} dB_{\vec{l}} \left(\sum_{\vec{k} \in \beta(i)} \frac{\rho_i}{2\pi A_i} L_{\vec{k},\vec{l}} - \sum_{\vec{k} \in \beta(i')} \frac{\rho_{i'}}{2\pi A_{i'}} L_{\vec{k},\vec{l}} \right)$$

Using the following notations :

$$\begin{cases} \overrightarrow{O_{\vec{k}} T_{\vec{k}}} = dk \, \vec{u}_{\vec{k}} \\ \overrightarrow{O_{\vec{l}} T_{\vec{l}}} = dk \, \vec{u}_{\vec{l}} \\ M_{\vec{k}} = O_{\vec{k}} + \lambda \overrightarrow{O_{\vec{k}} T_{\vec{k}}} \\ M_{\vec{l}} = O_{\vec{l}} + \lambda \overrightarrow{O_{\vec{l}} T_{\vec{l}}} \end{cases}$$

we get, after some computations [Der04] :

$$dB_{\vec{h}} = dE_{\vec{h}} + \sum_{\vec{l} \in B} dB_{\vec{l}} \, S_{\vec{l},\vec{h}} \tag{2.5}$$

with :

$$S_{\vec{l},\vec{h}} = \sum_{\vec{k} \in \beta(i)} \frac{\rho_i}{2\pi A_i} \int_0^1 \int_0^1 \log\left(\frac{\left\|\overrightarrow{M_k M_l}\right\|}{\left\|\overrightarrow{M_{\tau(k)} M_l}\right\|} \right) \left(\vec{u}_{\vec{k}} . \vec{u}_{\vec{l}}\right) dk \; dl \; d\lambda \; d\mu \tag{2.6}$$

We study then what happens to equation (2.6) when the distances dk and dl decrease toward 0. By grouping the terms of both the "vertical" and the "horizontal" edges, we get the following equation [Der04] :

$$S_{\vec{l},\vec{h}} = \frac{\rho_i}{2\pi d_{hl}^2} dk \, dl \left[\left(\vec{u}_{\vec{h}}^{\perp} . \vec{u}_{\vec{l}} \right) + 2 \left(\vec{u}_{hl} . \overrightarrow{u_{\tilde{\sigma}^{-1}(\vec{h})}} \right) \left((\vec{u}_{\vec{l}} \wedge \vec{u}_{hl}) . \vec{u}_n \right) \right]$$

Let d_{hl} denotes the distance between the points M_k and M_l, $\vec{u}_{hl}$ the normalized direction between these points, and $\vec{u}_n$ the normal to face i on edge h. We can now express the gradient of the radiosity [Der04] in the equation (2.7) that follows :

$$\begin{aligned} \overrightarrow{grad \; B_i} = \overrightarrow{grad \; E_i} &- \frac{\rho_i}{2\pi} \sum_{j \in F} \iint_{A_j} \frac{1}{d_{hl}^2} \left(\overrightarrow{Grad \, B_j}^{\perp} \wedge \vec{u}_n \right) dA_j \\ &- \frac{\rho_i}{\pi} \sum_{j \in F} \iint_{A_j} \frac{1}{d_{hl}^2} \left(\vec{u}_{hl} - \left(\vec{u}_{hl} . \vec{u}_n\right) \vec{u}_n \right) \left(\left(\overrightarrow{Grad \, B_j} \right)^{\perp} \wedge \vec{u}_{hl} \right) . \vec{u}_n \Big) dA_j \\ &- \sum_{j \in F} \int_{C_j} \frac{B_j \rho_i}{2\pi d_{hl}^2} \left[\left(\vec{u}_{\vec{l}} \wedge \vec{u}_n \right) + 2 \left(\vec{u}_{hl} - \left(\vec{u}_{hl} . \vec{u}_n\right) \vec{u}_n \right) \left((\vec{u}_{\vec{l}} \wedge \vec{u}_{hl}) . \vec{u}_n \right) \right] dl \end{aligned}$$

We also check that this equation is still valid even in the case of a discretization using parallelograms instead of squares. We get, instead of expressions in [Hol95], a gradient contained in the plane since we consider the radiosity as a two dimension function. But equations are close especially in the simple case that we will study in the next section. This confirms the two different approaches.

The gradient being orthogonal to the contour levels, it can prove interesting to consider the cross product of the previous equation with the normal to the surface to obtain the tangent to the contour levels :

$$\overrightarrow{grad\,B_i}^{\perp} = \overrightarrow{grad\,E_i}^{\perp} - \frac{\rho_i}{2\pi}\sum_j \iint_{surface} \frac{1}{d_{hl}^2}\left(\overrightarrow{GradB_j}^{\perp} \wedge \vec{u}_n\right) \wedge \vec{u}_n \, dA_j$$

$$- \frac{\rho_i}{\pi}\sum_j \iint_{surface} \frac{1}{d_{hl}^2}\left(\vec{u}_{hl} \wedge \vec{u}_n\right)\left(\left(\vec{u}_{hl} \wedge \vec{u}_n\right).\overrightarrow{GradB_j}^{\perp}\right) dA_j$$

$$- \frac{\rho_i}{2\pi}\sum_j \int_{bord} \frac{B_j}{d_{hl}^2}\left[\left(\vec{u}_l \wedge \vec{u}_n\right) \wedge \vec{u}_n + 2\left(\vec{u}_{hl} \wedge \vec{u}_n\right)\left(\left(\vec{u}_{hl} \wedge \vec{u}_n\right).\vec{u}_l\right)\right] dl$$

(2.8)

3. GRADIENT IN A SIMPLE CASE

We will show here an interesting use of the radiosity gradient in a simple case : a disc lighting a plane. This simple case allows us to compute in any point of the plane an analytic expression of the radiosity and also an analytic expression of the radiosity gradient.

Analytic equation of radiosity

We consider now the simple case of a disc of center *O* emitting light and illuminating a surface contained in an infinite plane *P* (cf. figure 2). Moreover, we will consider that the radiosity of the disc will be constant and equal to its emittance.

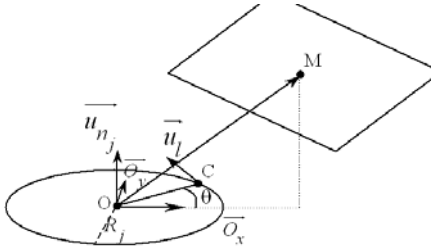


Figure 2 : Our study case : A source disc lighting a plane

We will also consider that the radius of the disc is small compared to the distance disc – plane, a property that we will use to compute difficult integrals. Finally, we do not consider any occlusion between the disc and the plane.

The case of a disc emitting light is nothing new. It was abundantly studied and multiple analytic solutions, approximate or not, were formulated [Wal89]. The contribution of the face j to the radiosity of face i is computed according to the approximation given by Wallace et al.:

$$Bi = \frac{\rho i B j A j}{n}\sum_{k=1}^{n} \delta k \frac{\cos(\theta kj)\,\cos(\theta ki)}{\pi\, rki + \frac{Aj}{n}} \qquad (3.1)$$

where r_{ki} is the distance from a source point to *M*, θ_{kj} (respectively θ_{ki}) is the angle between the normal to the receiver (respectively to the emitter) and the vector between this two points, and δ_k has 1 for value if the points see each other and 0 else.

Analytic equation of the gradient

We now try to use equation (2.8) to compute a simple analytic equation of the gradient of radiosity at any point of the plane, knowing that radiosity is constant on the emitter:

$$\overrightarrow{grad\ B_i}^{\perp}(M) = -\frac{\rho_i}{2\pi}\int_C \frac{B_j}{d_{hl}^2}\left[\left(\vec{u}_l \wedge \vec{u}_{n_i}\right) \wedge \vec{u}_{n_i} + 2\left(\vec{u}_{hl} \wedge \vec{u}_{n_i}\right)\left(\left(\vec{u}_{hl} \wedge \vec{u}_{n_i}\right).\vec{u}_l\right)\right] dl$$

After some manipulations and a polynomial expansion of order 2 in R_j/OM, we wet a simple analytical expression of the radiosity gradient on any point *M* of a plane lighted by a disc.

$$\overrightarrow{grad\ B_i}^{\perp}(M) = \frac{\rho_i L_j}{OM^4}\left[\left(\overrightarrow{OM}.\vec{u}_{n_i}\right)\left(\vec{u}_{n_j} \wedge \vec{u}_{n_i}\right) - 4\left(\vec{u}_{n_i}.\vec{u}_{n_j}\right)\left(\overrightarrow{OM} \wedge \vec{u}_{n_i}\right) + 4\frac{\left(\overrightarrow{OM} \wedge \vec{u}_{n_i}\right).\left(\overrightarrow{OM} \wedge \vec{u}_{n_j}\right)}{OM^2}\left(\overrightarrow{OM} \wedge \vec{u}_{n_i}\right)\right] \qquad (3.2)$$

4. RENDERING USING RADIOSITY GRADIENT

We use here our analytic expression of radiosity and its gradient in any point of a plane illuminated by a source disc. We present in this section an algorithm using this gradient that improves and speeds up the rendering of such a surface.

Unimodality of Radiosity

Radiosity is a function from $\Re^2$ to $\Re$. The idea our rendering method is to use the image space instead of the definition space as it is done in classical rendering algorithms, including the ones based on the radiosity method.

We decide to discretize the image space so that it follows closely the various values radiosity can take on a surface (cf. figure 3). This approach, though it can seems similar to the one presented in [Dre93], differs in the exact use of the gradient and the use of a radial discretization scheme.

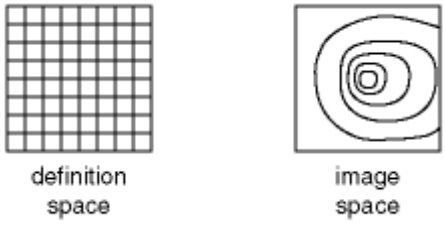


Figure 3. definition space discretization vs. image space discretization

Moreover we know, thanks to the unimodality properties of the radiosity solution, that if we consider the case of a single light source then the contour levels of the radiosity function are imbricated curves. Therefore, any line going from the maximum of radiosity toward any direction of the plane intersects every contour level of the radiosity

function whose value lies between this maximum and 0.

Discretization of the image space

To be able to discretize the image space we have to find where the maximum of the radiosity function lies. The position of this maximum is given by the following expression:

$$B(x) = \frac{h\ x\ \sin(\phi) + h^2\ \cos(\phi)}{(x2+h2)2} \qquad (4.1)$$

where h is the distance from source point O to plane P, x is the distance from point M to the projection of O on P, and ϕ is the angle between the direction of the source point and the normal to plane P.

The maximum of equation (5.1) is given by the following formula :

$$\frac{1}{6}\ \frac{h(-4+2\sqrt{4+3\tan(\phi)^2})}{\tan(\phi)} \qquad (4.2)$$

Once we know this position, we choose first the set of radiosity values we want to represent, and then a set of directions sampling uniformly all the directions. Then for every direction d and for every value of radiosity B_k, we find the intersection point $X_{k,d}$ between the line $[Od)$ and the contour level B_k (cf. figure 4) using a variation of the classical gradient algorithm.

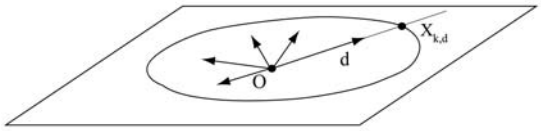


Figure 4. Intersection of a direction and a line of contour

Once we have these points, we have to restrict the rendering of the lighted plane to the boundaries of the real surface. This is done using OpenGL's stencil test to restrict the drawing to the lighted area.

5. RESULTS

We implemented the previous algorithm allowing us to discretize more adequately the illuminated surfaces. This discretization which follows the contour levels of the radiosity function, allows us to render faithfully this function at a lower cost in terms of vertices compared to a classical square discretization scheme.

Our technique following as closely as possible the contour levels of the radiosity function, we can expect visual results that are more faithful than those given by a square discretization. Figure 5 shows the results given by a 10x10 square discretization and those given by our results using 20 radial subdivisions and 5 contour levels. As expected we get a real improvement in terms of visual aspect.

Figure 5 also allows us to compare the results generated by our technique using a radial discretization scheme with 20 radial subdivisions and 5 contour levels and those generated by a 30x30 square discretization. With such discretization levels we achieve comparable results. As expected, thanks to our radiosity-following discretization scheme, we get good visual results while using fewer vertices.

6. DISCUSSION AND CONCLUSION

This article presents a new rendering technique based on a radial discretization scheme which follows the contour levels of the radiosity function. To achieve this, we present first an equation of the radiosity gradient in a simple case of a source disc lighting a plane. Then a technique similar to the gradient method is used to compute the positions of our vertices. Our method generates a radial mesh that is closer to the radiosity function and is therefore less costly in terms of vertices, allowing us to build images that are both more realistic and faster to compute.

We plan to use this rendering technique in a fast global illumination algorithm to compute coarse radiosity solution. Moreover, the equations presented in this article do not constraint in any way the formulation of the radiosity function, and therefore we want to apply this technique to higher order bases methods.

7. REFERENCES

[Der04] O. Derpierre, Details of the mathematical computation of the gradient : http://www-igm.univ-mlv.fr/~oderpier/recherche/radiosite_dif/index.html

[Dre93] G. Drettakis, E. Fiume, *Accurate and Consistent Reconstruction of Illumination Functions Using Structured Sampling*, in Eurographics'93 conf.proc., Computer Graphic Forum, vol. 12 (3), pp. 273-284, September 1993.

[Gor84] C.M. Goral, K.E. Torrance, D.P. Greenberg, B. Battaile, *Modeling the Interaction of Light between Diffuse Surfaces*, in ACM Siggraph'84 conf.proc., Computer Graphics, vol. 18 (3), pp. 213-222, July 1984.

[Hol95] N. Holzschuch, F.X. Sillion, *Accurate Computation of the Radiosity Gradient for Constant and Linear Emitter*, in 6th Eugraphics Workshop on Rendering conf.proc., Rendering Techniques'95, pp. 186-195, June 1995.

[Wal89] J.R. Wallace, K.A. Elmquist, E.A. Haines, *A Ray Tracing Algorithm for Progressive Radiosity*, in ACM Siggraph'89 conf.proc., Computer Graphics, vol. 23 (3), pp. 315-324, July 1989.