Feature-Preserving and Memory-Efficient Simplification of Polygonal Meshes

Article · January 2005

DOI: 10.2316/Journal.205.2005.4.205-4060

Muhammad Hussain محمد حسين

From: "journals" <journals@iasted.org>
To: <mhussain@i.kyushu-u.ac.jp>
Sent: Friday, August 20, 2004 6:20 AM
Subject: Paper Status - Paper 205-4060

Paper Number: 205-4060
Title: FEATURE PRESERVING AND MEMORY EFFICIENT SIMPLIFICATION OF POLYGON MESHES
Authors: Muhammad Hussain, Yoshihiro Okada, and Koichi Niijima

Dear Dr. Hussain,

I am very pleased to inform you that the above-mentioned paper has been accepted for publication in the International Journal of Modelling and Simulation. This paper has passed peer review, and final approval by a member of the Editorial Board.

In order to begin processing your paper, we require the following items:

1. A double-spaced MS word document of your final paper. If your paper is in Latex (.tex) format, please also submit your final paper in pdf for copy editing.
2. An electronic copy of your figures (.eps, .ps, .jpg, etc.). Please ensure that your figures are labeled according to the figure number, i.e. the file for figure 1 is labeled "figure 1.eps".
3. To avoid applicable extra page charge, please try to confine your final paper to within 16 pages (single-column, double-line spacing with regular 12 pt font) inclusive of all sections, equations, tables, figures, and photos. Extra page charge may apply if exceeded the limit.

After necessary changes have been made to your paper, please send the final paper as an e-mail attachment to journals@iasted.org within 3 months. At this time, please also include a brief biography for each author (if possible, include photos with your bios. Please send us photo that is hi-res (300 dpi), black and white, and no less than 3.5 cm in width). An information sheet with our "format requirements" can be found on our web site at www.actapress.com/journals/format.htm. Please also include a manuscript checklist, which can be found at http://www.actapress.com/journals/checklist.htm.

Your final paper will be copy edited by our staff and formatted for publication using LaTeX. When processing of your paper is complete, a galley proof will be sent to you to check over. I will schedule your paper to be published in the next issue, if possible. Page charges may apply: for papers exceeding eight (two-column) printed pages, including illustrations, there is a mandatory page charge of US $100 per page, which is a prerequisite for publication.

Thank you very much for helping us maintain the consistent high quality of this publication.

Sincerely,

Estella Chan
Publisher - Journals
ACTA Press / IASTED
#80, 4500-16th Ave. NW
Calgary, AB T3B 0M6
Canada
phone: (403) 288-1195
fax: (403) 247-6851

e-mail: journals@iasted.org
web site: http://review.iasted.com/journal
web site: http://www.actapress.com

FEATURE PRESERVING AND MEMORY EFFICIENT SIMPLIFICATION OF POLYGONAL MESHES

M. Hussain, Y. Okada, and K. Niijima

Graduate School of Information Science and Electrical Engineering,

Kyushu University, Fukuoka, Japan.

{mhussain, okada, niijima}@i.kyushu-u.ac.jp

Abstract

We propose a new automatic edge collapse simplification algorithm, which is not only fast and memory efficient but also automatically preserves visually imprant features of a mesh. In an edge collapse algorithm, a sequence of edge collapse transformations determined by an optimal greedy approach is applied until an approximating mesh of required size or of given tolerance is obtained. The way how to measure the geometric error introduced as a result of an edge collapse transformation plays a crucial role in determining the priority ordering of such transformations. We introduce a new idea to measure this error based only on current simplified mesh and error accumulation. The proposed technique for measuring geometric error is not only simple to implement, but is also memory efficient. The presented algorithm consumes less memory and takes less execution time than most of the published edge collapse *based algorithms. Results and numerical comparisons show that our algorithm generates simplified meshes of good visual fidelity, which are comparable with those by other methods.*

Key Words

Triangular Meshes, Surface simplification, Level of detail, Edge-collapse, Multiresolution modeling

1 Introduction

Automatic simplification of large polygonal meshes is an important problem in CG. Recent advances in scanning technologies have given rise to highly detailed polygonal meshes whose has surpassed the development in rendering systems; it has become hard to interactively display and navigate such large models. The solution lies in employing polygonal meshes with varying detail in different application contexts and simplification techniques form its backbone.

The importance of simplification techniques has motivated intense research in this area with different goals: geometric simplification, topology simplification, and view-dependent simplification. This paper focuses on geometric simplification. During the past years, many simplification algorithms, addressing geometric simplification, have appeared in the graphics literature [16, 15, 6, 14, 8, 4, 5, 1, 9, 11, 17, 13]. Most of the algorithms are based on iterative approach, where small local geometric change is introduced according to some optimality criterion. The optimality criteria is usually based on one of the two types of measures of error: *local error,* which is measured by comparing the small affected patch of a mesh with the current simplified mesh, and *global error,* which is based on the comparison with original mesh. The algorithms proposed in [15, 16] follow the local approach for measuring geometric deviation. Algorithms presented in [6, 1, 4, 8] are based on global measure of error; while they produce high quality simplifications, they require the geometric history to be carried along the partly simplified mesh, which

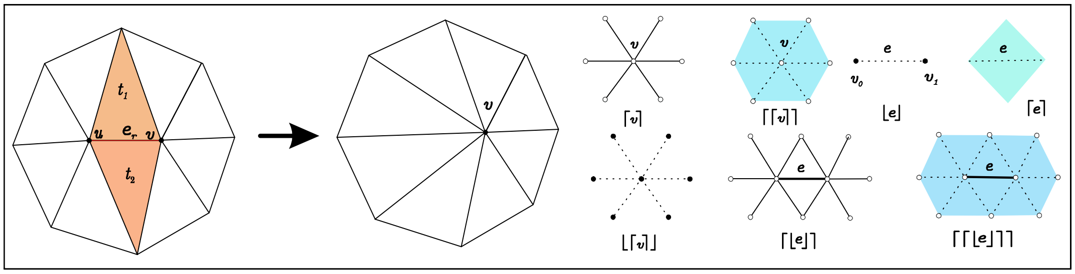

Figure 1. (Left) Edge collapse operation. Edge e_r is substituted with v and triangles t_1 and t_2 are eliminated. (Right) Simplex operators.

renders the iterative process memory consuming and very slow. Our proposed algorithm is based on half-edge collapse; it is neither based on a local nor a global approach; although, the measure of geometric fidelity is based only on partly simplified mesh, but we accumulate the error introduced by an edge collapse to the cost of each of the half-edges having their starting vertex on the terminating vertex of the collapsed edge. So it is not only memory efficient but also faster than almost all those iterative methods, which use geometric history to execute optimality criteria and even Memoryless Simplification [14], which is also a memory efficient algorithm, and yields approximations of good visual fidelity.

The remainder of this paper is organized as follows. In Section 2, we give an overview of some of the related iterative simplification algorithms. Section 3 outlines our simplification algorithm. Error metric used in our algorithm has been detailed in Section 4. Collapse of edges on the boundary has been discussed in Section 5. In Section 6, we present the empirical and numerical results and compare it with some other published algorithms. Section 7 concludes the paper.

2 Related Work

Most of the iterative simplification methods can be classified into three categories with respect to the topological operator they adopt: *vertex decimation*, *edge collapse*, and *face decimation*. Of all these simplification operations, edge collapse is the most attractive one because of its simplicity and robustness. Since the proposed algorithm is based on edge collapse and we center our attention on edge collapse algorithms and give an overview of the most related recent methods. For thorough survey of simplification algorithms, please consult [7, 3].

Hoppe's algorithm [8] for constructing progressive meshes is the pioneer of the class of edge collapse algorithms. Garland and Heckbert [5] measure the squared distance from the collection of planes associated with triangles incident on a vertex and store it as a symmetric 4×4 matrix, one matrix per vertex. While their approach is fast and gives high quality approximations, it is not memory efficient. For each vertex it stores ten floats. Lindstrom and Turk [14] uses linear constraints, based primarily on the conservation of volume, to define a memoryless version of quadric error metric [5]; it produces good quality simplifications and is fairly efficient, particularly in memory consumption, but it is slower than *QSlim*. Hussain et. al [10] also proposed a memory efficient method; the proposed algorithm improves this method in three ways: accumulation of error is more appropriate, measure of error is computationally more efficient, and the quality of boundary preservation is better.

3 Simplification Algorithm

In this section, we outline our algorithm and briefly highlight its main features. First of all, to fix the ideas a brief description of terminology and notation is in place.

A polygonal mesh is specified by a pair (P, K), where P is a set of n point positions $P = \{\mathbf{v}_i \in R^3 \mid 1 \leq i \leq n\}$ and K is an abstract simplicial complex which contains all the topological information. In other words, P and K describe geometry and topology of a polygonal mesh. The complex K is the set of subsets of $\{1, 2, 3, \cdots, n\}$, which are known as *simplices*; *vertex* is a 0-simplex, *edge* is a 1-simplex and *face* is a 2-simplex. We represent 0-simplex or vertex by v with its geometric counterpart as a $3D$ vector $\mathbf{v}$; an edge e or 1-simplex is a subset $\{v_0^e, v_1^e\}$; an oriented edge is represented by an ordered pair (v_0^e, v_1^e) and is denoted by $\vec{e_{01}}$. A triangle t or 2-simplex is a set of oriented edges i.e. t = $\{\vec{e_{01}}, \vec{e_{12}}, \vec{e_{20}}\}$ or t = $\{(v_0^{e_0}, v_1^{e_0}), (v_1^{e_1}, v_2^{e_1}), (v_2^{e_2}, v_0^{e_2})\}$ or simply t = (v_0, v_1, v_2).

According to the definition of simplex operators $\lfloor\rfloor$ and $\lceil\rceil$ as defined in [14], $\lceil v\rceil$, $\lceil\lceil v\rceil\rceil$, $\lfloor\lceil v\rceil\rfloor$, $\lfloor e\rfloor$, stand for edges incident on v, triangles incident on v, neighboring vertices of v, vertices of e respectively and, $\lceil\lfloor e\rfloor\rceil$ and $\lceil\lceil\lfloor e\rfloor\rceil\rceil$ represent the sets of edges and triangles sharing at least one vertex with e, see Figure 1.

3.1 Outline of the Algorithm

The proposed method, like most related algorithms, is a simple greedy procedure. It is based on half-edge collapse and a new measure of accuracy detailed in Section 4. It takes the original mesh as input and executes the following steps and yields progressive mesh representation as output.

- Compute the cost of collapse for each half-edge in the original triangular mesh using the proposed error metric (Section 4) and put them in priority ordering with least cost edge first.

- Choose the half-edge $\vec{e_{01}}$ = $(v_0,\ v_1)$ with minimum cost and substitute it with v_1. During this operation triangles $\lceil e_{01}\rceil$ become singular and are discarded. The remaining edges $\lceil\lfloor e_{01}\rfloor\rceil - \{e_{01}\}$ and triangles $\lceil\lceil\lfloor e_{01}\rfloor\rceil\rceil - \lceil e_{01}\rceil$ incident upon v_0 and v_1 are updated such that all occurrences of v_0 are replaced with v_1.

- Re-evaluate the cost of the edges $\lceil\lfloor e\rfloor\rceil - \{e\}$ after the collapse of the edge e.

- Take maximum of the cost of collapse of the half-edge $\vec{e}_{01}$ = $(v_0,\, v_1)$ and that of the half-edge $\vec{e}_{1i}$

$= (v_1, v_i)$ emanating from v_1; this maximum value will be the cost of the half-edge $\vec{e}_{1i} = (v_1, v_i)$. Update priority ordering of the edge collapse transformations.

Note that the new cost is evaluated against the current partly simplified mesh, global effect of geometric error is achieved by accumulating the error described in the last step. This step is one of the differentiating factor between this algorithm and the one proposed in [10].

The choice of a particular topological operator has no significant effect on the results [12]; what matters is the way how to measure the accuracy. We employ *half-edge collapse* as a topological operator. In this case, the vertices of the simplified mesh form a subset of the original vertices; this makes progressive transmission of meshes more effective and is crucial for integrated level of detail extraction. Apparently, it seems that half-edge collapse transformation results in triangles with bad aspect ratio, but we found empirically considering a wide range of meshes that the triangles are of good quality everywhere except a few triangles at regions of high curvature and these can be avoided if our error measure is coupled with the aspect ratio test proposed in [6], but it will be at the cost of a little bit degeneration in accuracy; *QSlim* [5] and *MS* (Memoryless Simplification) [14] also exploit this criteria to avoid sliver triangles.

The main characteristics of the algorithm are as follows:

- Visually important features are preserved in a better way e.g. see Figure 7.
- The algorithm is memory efficient because it needs not to store any kind of geometric history.
- In respect of execution time, it is the fastest one after *QSlim*.
- The approximations generated by this algorithm are comparable with those created by the state-of-the-art algorithms in terms of maximum geometric error.

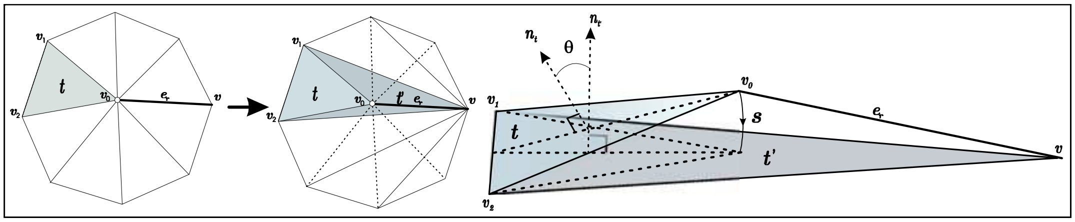

Figure 2. (Left) When edge e_r is collapsed, triangle $t = (v_0, v_1, v_2)$ is mapped to triangle $t' = (v, v_1, v_2)$.(Right) When triangle t is mapped to triangle t', the vertex v_0 traverses the arc s if only rotation of t is considered.

4 Error Metric

The criterion for the evaluation of approximation error is based on an intuitive observation. The collapse of an arbitrary edge e_r removes triangles $\lceil e_r \rceil$ and transforms the remaining triangles $\lceil\lceil v_0 \rceil\rceil - \lceil e_r \rceil$, see Figure 2 for geometric description. Analogous to the length of a circular arc, the partial geometric error introduced by the transformation of a typical triangle $t = (v_0, v_1, v_2)$ to $t^{'} = (v, v_1, v_2)$, see Figure 2, is

$$Q_t = l_t \theta$$

where

$$l_t = \frac{1}{2}(|\mathbf{a}| + |\mathbf{b}|), \mathbf{a} = \mathbf{v}_0 - \mathbf{v}_1, \mathbf{b} = \mathbf{v}_0 - \mathbf{v}_2,$$

and θ is the dihedral angle between the triangles t and t'. To avoid the computational overhead incurred by the evaluation of θ, we approximate θ by $1 - n_t \cdot n_{t'}$, where n_t and $n_{t'}$ are unit normals to the triangles t and t', as shown in Figure 2.

The cost of the transformation $\vec{e}_r(v_0, v) \rightarrow v$ is the sum of partial errors contributed by each of the triangles in $\lceil\lceil v_0 \rceil\rceil - \lceil e_r \rceil$, $i.e.$

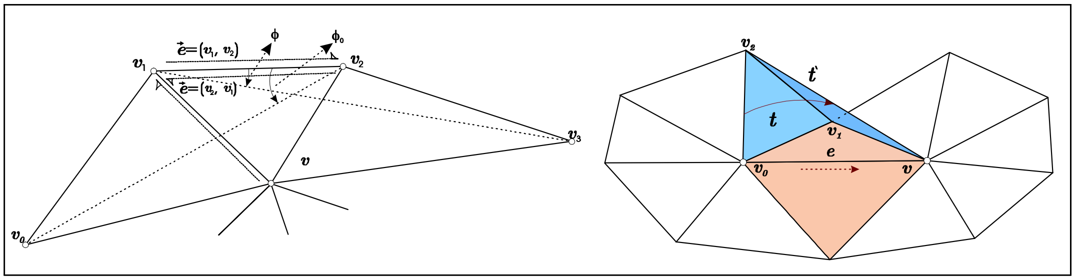

Figure 3. (Left) Boundary simplification: v **is an interior vertex and** $v_0, v_1, v_2,$ **and** v_3 **are boundary vertices. (Right) Collapse of edge** e **will cause the triangle** $t = (v_0, v_1, v_2)$ **to fold over the mesh.**

$$Cost(\vec{e}_r) = \sum_{t \in \lceil\lceil v_0 \rceil\rceil - \lceil e_r \rceil} Q_t.$$

Note that the edge collapse transformations $\vec{e}_r(v_0, v) \to v$ and $\vec{e}_r(v, v_0) \to v_0$ have different effects on geometric error; we collapse edge $e_r\{v_0, v\}$ by applying the transformation that results in minimum error.

5 Boundary Simplification

Boundary half-edges can be categorized into two main types: (1) the half-edges having one of the starting and terminating vertices on the boundary e.g. $\vec{e} = (v, v_1)$, note Figure 3 and (2) the half-edges having both vertices on the boundary e.g. $\vec{e} = (v_1, v_2)$, see Figure 3. Each case is dealt with separately.

The half-edge collapse transformation $\vec{e}\,(v_s, v_t) \mapsto v_t$ substitutes the half-edge with terminating vertex; so collapse of the half-edge having terminating vertex on the boundary needs not special treatment. However if starting vertex is on the boundary, then collapse of the half-edge will deform the boundary, so the collapse of this half-edge is not allowed.

The half-edge having both end vertices on the boundary must be dealt tactfully; it will obviously collapse to a vertex along the boundary. Now the problem is how to guide the greedy approach so that

model	model size	*FPME*	*QSlim*	model	model size	*FPME*	*QSlim*
Fandisk	12,946	2.4sec	1.5sec	Hand	654,666	78sec	39.9sec
Bunny	69,451	8.15sec	4.0sec	Dragon	871,414	103.5sec	53.6sec
Male	605,902	75.3sec	48.2sec	Blade	1,765,388	247.4sec	125.1sec

Table 1. Time taken in seconds to reduce to one face. Model size is in number of faces.

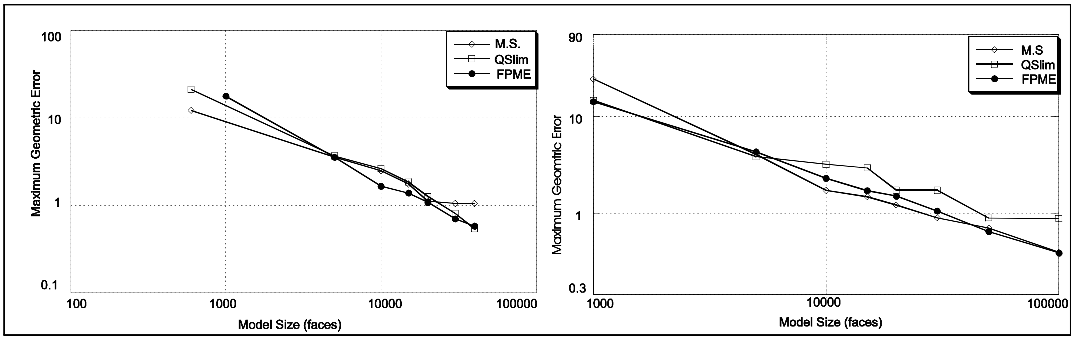

Figure 4. Maximum geometric error for (left) bunny and (right) hand models.

it is not entrapped in a local minimum; to this end some heuristics are employed. A similar part of the mesh is assumed on the exterior side of the half-edge as is on the interior side to bring it in line with the interior edges. Note Figure 3, edge $e = \{v_1, v_2\}$ may be collapsed either to v_1 or v_2, but to achieve better results this must be collapsed to v_2. To achieve this end we penalize the costs of collapse of the half edges $\overrightarrow{e} = (v_1, v_2)$ and $\overrightarrow{e} = (v_2, v_1)$ with the length of the edge weighted by ϕ and ϕ_0 respectively. So the cost of collapse of the half edge $\overrightarrow{e} = (v_2, v_1)$ whose both vertices are on boundary is

$$Cost(\vec{e}) = \lambda \phi s \parallel v_1 - v_2 \parallel + 2 \sum_{t \in \lceil\lceil v \rceil\rceil - \lceil e \rceil} \mathbf{Q}_t$$

where $\phi = 1 - u_1 \cdot u_2$, u_1 and u_2 being the unit vectors along the edges $\vec{e} = (v_1, v_2)$ and $\vec{e} = (v_1, v_3)$ as is shown in Figure 3, and $s = \min_{t \in \lceil\lceil v_1 \rceil\rceil} \{s_t\}$, s_t being the quality measure of the triangle t i.e. $s_t = \frac{4\sqrt{3}\triangle t}{l_1^2 + l_2^2 + l_3^2}$ [6]; this quantity causes the thin triangles to be eliminated along the boundary. Here λ is a parameter used to control the quality of boundary preservation; it can assume any value greater than zero. During

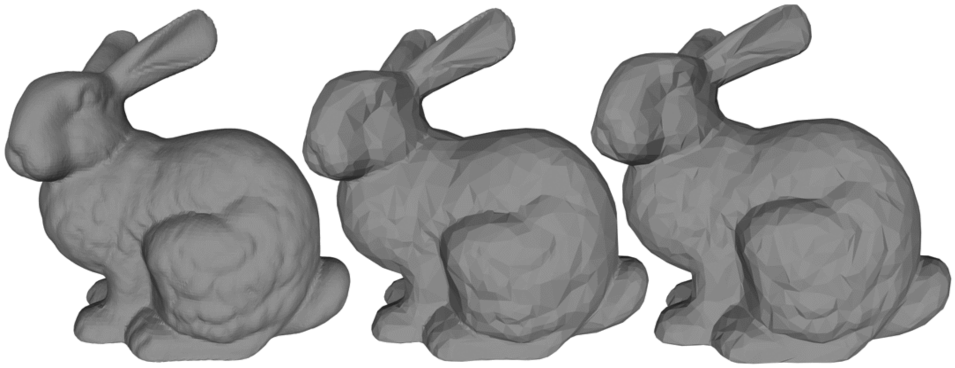

Figure 5. (Left to right) Original bunny model with 69451 faces, its reduced versions each having 5000 faces generated by FPME and QSlim.

our experiments we found that feasible results can be found using the value of λ in the range of 1 to 50, in our experiments we have used $\lambda = 10$. Closer the value of λ to 50, the tighter the boundary is preserved.

6 Experimental Results and Discussion

We tested *FPME*, implementation of the proposed algorithm, using a wide range of triangular meshes and achieved good results. To evaluate the proposed technique, we make comparison with two state-of-the-art methods *QSlim* [5] and *MS* (Memoryless Simplification) [14].

Table 1 lists the execution times, reported on 800MHz Intel PentiumIII machine with 384 MB of main memory, taken by *QSlim* and *FPME* to simplify various models to one face. Notice that *FPME* is about two times slower than *QSlim*. From the results reported in [14] (see table 1), it follows that *MS* is about 5 times slower than *QSlim*; so we can safely conclude that *FPME* is about 3 times faster than *MS*.

For thorough evaluation of the quality of approximations generated by *FPME*, we employ maximum geometric error measure and compute it using version 2.5 of Metro tool [2]. We tested our algorithm

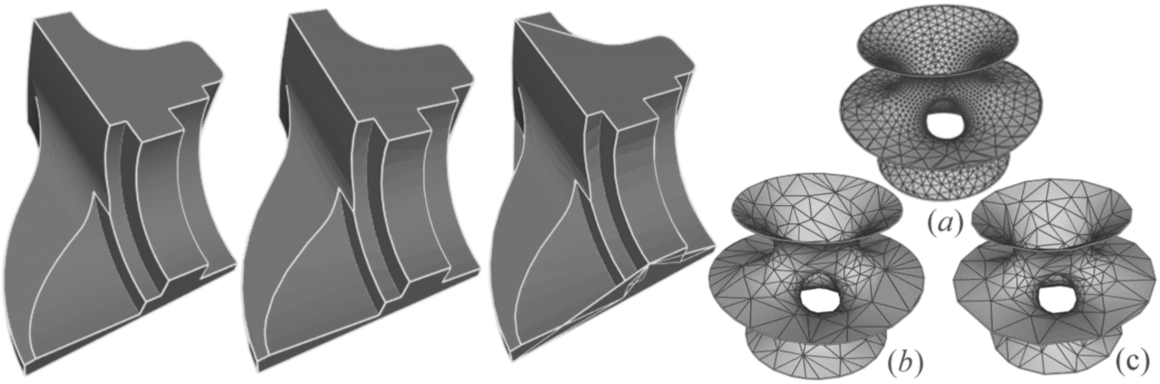

Figure 6. (Left to right) Original fandisk model, its simplified versions each having 496 faces produces by FPME and QSlim, (a) original hypersheet model with 3837 faces and its reduces versions created by FPME with (b) $\lambda = 25$ **(c)** $\lambda = 4$.

on most of the publicly available models and found similar results, due to space limitations we only present the test results for bunny and hand models because of their complex structures. Graphs shown in Figures 4 illustrate the maximum geometric error between the original and the simplified models created by *QSlim*, *MS* and *FPME*.

FPME is memory efficient like *MS*; it needs not to store any kind of geometric history. It only accumulates the cost of collapse, which does not consume extra memory like storing the geometric history. *QSlim* consumes at least 40 bytes per vertex to store a form of geometric history.

Folds may occur when an edge to be collapsed is surrounded by a very concave polygon, see Figure 3. The proposed error measure has the potential to automatically prevent the occurrence of folds; in case of a fold, it assigns greater value to θ and thus prevents fold. Figure 6 depicts that *FPME* automatically prevents folds whereas *QSlim* creates folds and it needs extra heuristics to prevent the folds.

Boundary is one of the main geometric features, which have effect on the visual appearance of the model, and must be preserved properly for good visual fidelity. The proposed idea is capable of preserv-

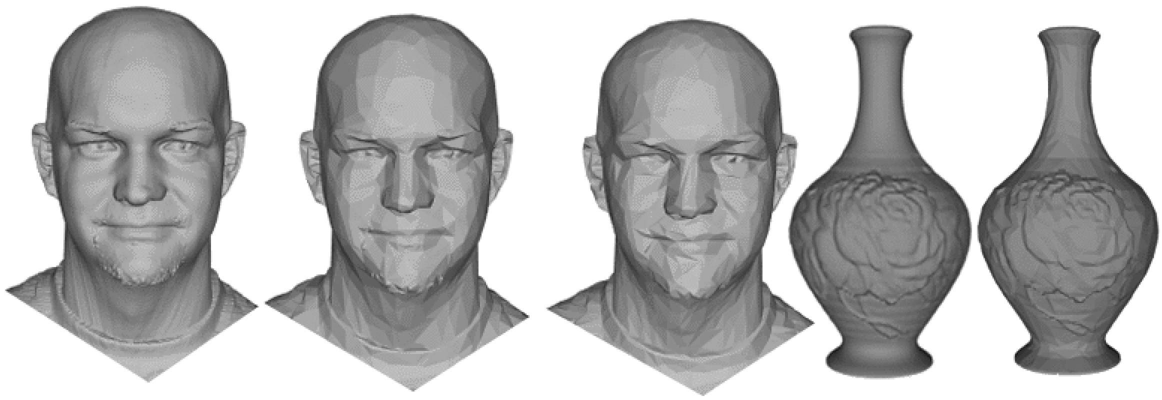

Figure 7. (Left to right) Original male model with #faces 605902, its simplified versions each with 4887 faces generated by FPME, and QSlim, original vase model with 135680 faces, and its reduced version with 4792 faces approximated by FPME.

ing the boundary with varying degrees of tightness as is demonstrated using hypersheet model shown in Figure 6. Note that the triangles along the boundary are relatively well-shaped.

Feature lines or lines of discontinuity are sharp edges whose two incidents faces describe a dihedral angle of measure less than some threshold. These lines reflect the overall geometric appearance of a model and are visually very important. The proposed error measure automatically preserves lines of discontinuity. Note fandisk models, original and simplified, shown in Figure 6.

High frequency detail resolves visually important features of a model. Again, the dihedral angle exploited by our measure of geometric deviation helps preserve high frequency detail automatically. Observe male models shown in Figure 7; it is obvious that *FPME* preserves eyes and lips finely whereas *QSlim* can't. Also observe the contours on the left feet of bunny model (Figure 5) and the vase models (Figure 7).

FPME can efficiently simply very large models. Observe male model and its simplified version (0.8 % of the original model) depicted in Figure 7; despite this drastic simplification, eyes, eyebrows, lips

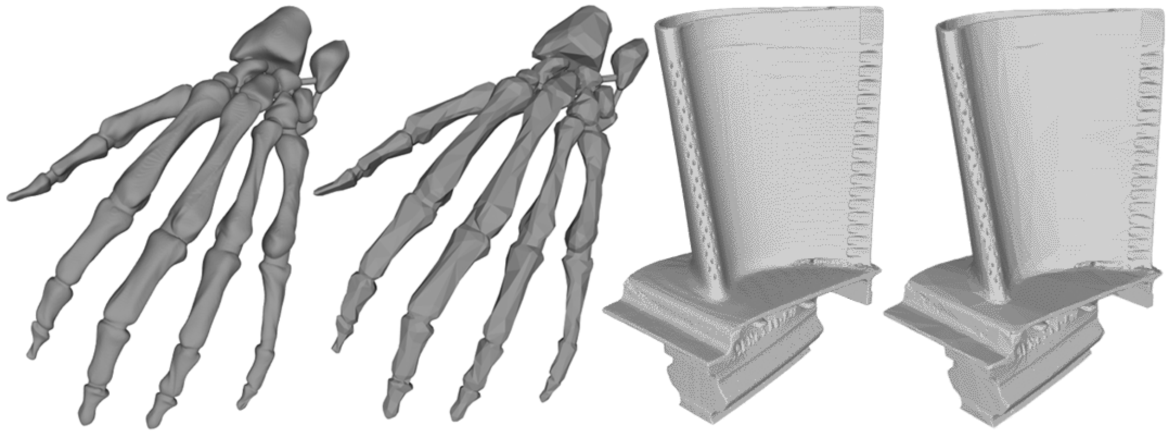

Figure 8. (Left to right) original hand model with 654,666 faces, its reduced version with 5000 faces, original blade model with 1,765,388 faces and its simplified version with 18524 faces.

and other visually important features are finely apparent. Figure 8 shows the simplified model of hand after 99.24% reduction. It can be seen that major features of the original models still remain in spite of being highly simplified.

Turbine blade model shown in Figure 8 consists of about 1.8 million triangular faces. Simplification of this model is a challenging task because of its sheer size, complicated topology with a large number of tiny holes, and complex geometry with many sharp edges.

7 Conclusion

A new method for geometric simplification of triangle meshes has been presented; it is memory efficient and faster than most of the state-of- the- art algorithms. It preserves automatically not only the lines of discontinuity and high frequency detail of a geometric model but also prevents automatically the creation of folds in a mesh. It can simplify huge models in relatively short time. The quality of simplifications is good and essential features are preserved even after significant reduction. This can be used for applications which require visual fidelity, not tight error bound, and the set of vertices of the simplified version to be a proper subset of original vertices.

References

[1] A. Ciampalini, P.Cignoni, C. Montani, and R. Scopigno. Multiresolution decimation based on global error. *The visual computer*, 13, 1997, 228-246.

[2] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum*, 17(2), 1998, 167-174.

[3] P. Cignoni, C. Montani and R. Scopigno. A comparison of mesh simplification algorithms. *Computer & Graphics*, 22(1), 1998, 37-54.

[4] J. Cohen, A. Varshney, D. Manocha, G. Turk, H. Weber, P. Agarwal, F. Brooks, and W. Wright. Simplification Envelopes. *Proc. SIGGRAPH'96*, 1996, 119-128.

[5] M.Garland and P.S.Heckbert. Surface simplification using quadric error metric. *Proc.SIGGRAPH'97*, 1997, 209-216.

[6] A. Guéziec. Locally toleranced surface simplification. *IEEE Transactions on Visualization and Computer Graphics*, 5(2), 1999, 168-189.

[7] P. Heckbert and M. Garland. Survey of surface simplification algorithms. Technical report, Carnegie Mellon University-Dept. of Computer Science, 1997.

[8] H. Hoppe. Progressive meshes. *Proc.SIGGRAPH'96*, 1996, 99-108.

[9] M. Hussain, Y. Okada, and K. Niijima. A fast and memory-efficient method for LOD modeling of polygonal models. *Proc.GMAG03*, IEEE Computer Society Press,London, 2003, 137-142.

[10] M. Hussain, Y Okada, and K. Niijima. Fast, simple, feature-preserving and memory efficient simplification of triangle meshes. *International Journal of Image and Graphics*, 3(4), 2003, 1-18.

[11] M. Hussain, Y. Okada, and K. Niijima. LOD modeling of polygonal models based on multiple choice optimization. *Proc. MMM*04, Brisbane, Australia, 2004, 203-210.

[12] L. Kobbelt, S. Campagna, and H. P. Seidel. A general framework for mesh decimation. *Proc. Graphics Interface'98*, 1998, 311-318.

[13] Kim, S. J., Kim, C. H., and Levin, D. Surface simplification using a discrete curvature norm. *Computers and Graphics*, 26, 2002, 657-663.

[14] P. Lindstrom and G. Turk. Fast and memory efficient polygonal simplification. *Proc. IEEE Visualization'98*, 1998, 279-286.

[15] M. Algori and F. Schmitt. Mesh simplification. *Computer Graphics Forum* 15(3), 1996.

[16] W. J. Schroeder, J. A. Zarge, and W. E. Lorenson. Decimation of triangle meshes.*Computer Graphics*, 26(2), 1992, 65-70.

[17] J. Wu, and L. Kobbelt. Fast mesh decimation by multiple-choice techniques. *Proc.Vision, Modeling, and Visualization 2002*, Erlangen, Germany, 2002.

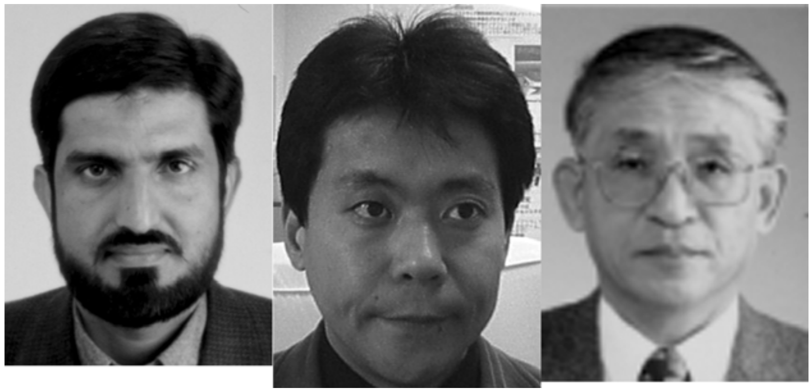

Figure 9. (Left to right) M. Hussain, K. Okada, and K. Niijima.

Muhammad Hussain: He received his M.Sc. degree in applied Mathematics and M.Phil. degree in Computational Mathematics specializing in CAGD, both from University of the Punjab, Lahore, Pakistan, in 1990 and 1993 respectively. In 1995, he joined the Department of Mathematics, Punjab University, Lahore, Pakistan as a lecturer where in addition to teaching graduate courses, he participated in a project on CAGD. In 2003, he received Ph.D. degree in Computer Science from Kyushu University, Japan. He is currently working as a JST researcher at the Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan. His current research interests include geometric modeling, image-based modeling and virtual reality.

Yoshihiro Okada: He received his B.Sc., M.Sc. and doctoral degrees in Engineering from Electrical Engineering Department, Faculty of Engineering, Hokkaido University, Japan, in 1988, 1990 and 1993 respectively. He was an Associate Professor at Computing Center, Kyushu University, Japan from 1999 to 2000. He has been an Associate Professor in the Department of Informatics, Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan since April, 2000. His current research interests include 3D graphics, software architecture, network collaboration technology and human-computer interface.

Koichi Niijima: He received the B.S. degree in 1967, the M.Sc. degree in 1969, and the Ph.D. degree in 1977 in Mathematics from Kyushu University, Fukuoka, Japan. In 1971, he was a research Assistant at Kyushu University, and from 1974 to 1981, he was an Associate Professor at Fukuoka Women's University. He worked from 1981 to 1988 as an Associate Professor of Mathematics at Kyushu University, and from 1988 to 1996, he was a Professor of Control Engineering at Kyushu Institute of Technology. Since 1996, he has been with Kyushu University, where he is currently a Professor in the Department of Informatics. His current research interests include wavelets, in particular lifting wavelets, and their applications to image analysis and signal processing and geometric modelling.