Feature-preserving Simplification of Polygonal Surface based on Half-edge Contraction Manner

Myeong-Cheol Ko, Yoon-Chul Choy

Multimedia/Graphics Lab. Dept. of Computer Science, Yonsei Univ.
134 Shinchon-Dong, Seodaemun-Gu, Seoul 120-749, Korea
{zoo,ycchoy}@rainbow.yonsei.ac.kr

Abstract. Polygonal surface models are generally composed of large amount of complex polygonal patches. In real-time applications, such as VR or 3D simulation system focusing on the real-time interactivity, various simplified versions of those models are often used according to the performance of the system. The goal of surface simplification is to take a complex polygonal model as input and generate a simplified model, which is an approximation of the original model, as output without the loss of geometric properties if possible. In this paper, we present a surface simplification algorithm, which can excellently preserve the characteristic features of the original model, even after drastic simplification process. To decrease the amount of real-time transmission of surface data, the proposed algorithm is based on the half-edge collapse manner. The half-edge based scheme is more efficient in memory usage and useful in real-time rendering applications, which require progressive transmission of large quantity of surface data compared to other methods. We present numerical and visual comparisons showing that the proposed algorithm results in higher quality approximations of original models than other algorithms in the literature with excellent shape preservation even after drastic simplification process.

1 Introduction

With the introduction of reverse engineering in the fields of 3D modeling [1], there has been increased need for complex and high detailed models, which are generally created by 3D laser scanner, in many computer graphics applications. However, the full complexity of such models is not always required and desirable in applications, such as simulation and virtual reality systems. Therefore, it is useful to have various simpler versions of original complex models according to its uses in applications.

In polygonal surface simplification, the goal is to take a complex polygonal model as input and automatically generate a simplified model as output without a loss of geometric properties of the original model if possible. The simplification algorithms start with an original model, iteratively remove elements from the model in each simplification step until the desired level of approximation is achieved. To decide the order of elements for removal during the simplification process, most existing algorithms use error metric based on distance optimization. That is, the order is determined so that the distance between the meshes before and after removal operation is minimized. However, it is difficult to define exactly the local geometric

characteristics of surface using just the distance metric, which is intrinsically scalar and the degree of loss for geometric information caused by simplification cannot be guaranteed [4]. Since the distinguishing surface features are mostly concentrated in small areas with small scalar values, such as area, length and volume, their decimation costs based on distance metric are often low. This means that the distinguishing features can be removed in the earlier stage of simplification process (Fig.1). In other words, the error metric cannot retain the distinguishing features, and consequently, the detailed local shapes of original surface are lost as the simplification ratio increases.

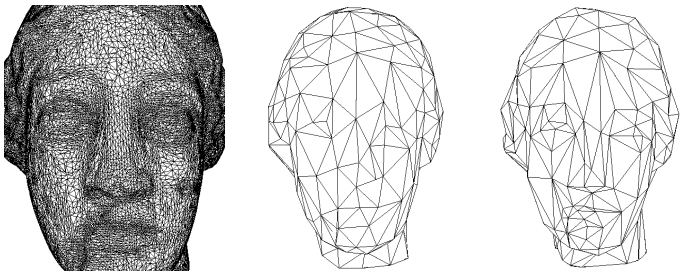


Fig. 1. Distribution of the geometric features and degree of preservation of the features in the drastic simplification ratio. In here, the simplified models are composed of only 0.5% of the original model. (a): result from the traditional method based on distance optimization. (b): result from the method, which considers orientation component as the additional factor besides the scalar value. Characteristic features are well preserved in (b), and it is much closer to the original

The surface orientation is one of the important features representing the characteristics of local surface and it is independent from the amount of scalar, such as distance measure. So, by considering the orientations of local elements, we can reconsider whether the elements should be preserved or not even though they are the elements with small scalar values.

In this paper, we present an error metric that describes and reflects the geometric features of surface and the geometric changes before and after simplification. To define the error metric, we introduce the orientation component of local surface as an additional property. In addition, a surface simplification algorithm based on half-edge collapse manner utilizing the error metric is implemented. The half-edge based scheme is the method that repositions a new vertex after the edge collapse by merging one of the two vertices to the other in previous edge having directionality. It is more efficient in memory usage and useful in real-time applications, which require progressive transmission of large number of surface data and instant rendering compared to existing methods adopting optimal positioning scheme. In many cases, according to our observation, when the original model is simplified less than some degree of simplification ratio, there is almost no visual difference. Moreover, applications for real-time rendering might demand higher rate of simplification to decrease the complexity of a scene. So, the proposed algorithm is concentrated on the preservation of geometric features of the original model even after performing drastic level of simplification.

2 Related Works

Most existing simplification algorithms, which are generally based on greedy strategy, locally perform a sequence of simple topological operations in each simplification step, to remove and update certain geometric elements. Previous works can be classified depending on the type of the topological operations. Possible topological operations are vertex decimation, edge decimation, triangle decimation and vertex clustering [3][6]. Out of these methods, the following two classes are broadly relevant to our work. We explain the characteristics of the algorithms in each class by focusing on the error metric employed in each algorithm.

Vertex Decimation Method. This method removes a vertex with a decimation cost lower than the predefined threshold in each simplification step. In [5], the distance between a given vertex and the average plane composed of its surrounding vertices is used for error metric. The Hausdorff distance between the simplified and the original mesh is used as error metric in [8]. And in [7], the curvature of local surface around a given vertex is used as error metric. In [17], for the planes of triangles adjacent to a given vertex in current mesh, the maximum distance from the corresponding planes of the intermediate mesh, and its accumulation are used for error metric.

These approaches need a robust re-triangulation method, as in [15], to fill the resulting hole caused by the removal of a vertex. And they mainly focus on the connectivity of a surface rather than the geometric characteristics. So, the quality of the approximation largely depends on the re-triangulation operation. In addition, the intermediate surface models, which are generated from the simplification process, do not have any direct hierarchical relationship with each other. Consequently, when these models are applied to the rendering applications based on the hierarchical LOD (Level Of Detail), it is difficult to switch views smoothly and continuously.

Edge Decimation Method. This is a simplification method based on iterative edge contraction, which replaces an edge with a vertex. Neighboring triangles, which contain both vertices of the edge, are removed from the mesh and remaining triangles are adjusted so that they are appropriately linked to the new vertex. In [9] and [12], energy function using the sum of the squared distance between the sample vertices from the original mesh and the simplified mesh is used for error metric. The quadric error metric is used in [14]. This metric is derived from the quadratic equation representing the measure of sum of squared distance between vertex and associated planes. And in [10], the degree of change of geometric properties, such as area and volume, between successive meshes is used for error metric.

In these approaches, each algorithm has to determine the positioning policy for a new vertex to replace the edge after contraction. The new vertex may or may not exist in the original mesh. The error metrics used in each algorithm provide the clues for optimal positioning of the generated vertex. In optimal approach, the generated vertex is not a subset of the original mesh in most cases.

After an analysis of existing studies, we decided to use the half-edge collapse scheme, which is one of the edge-based approaches, as the topological operation.

Since this method does not create a new vertex, it is memory efficient. And in most cases, this method preserves the original shape well, although it does not use the optimal placement policy.

In addition, as shown previously, most existing algorithms generally use the distance as the main factor of the error metric. However, it is difficult to describe the degree of deviation from the previous mesh after simplification operation using just the distance metric, which is intrinsically a scalar. So we define an error metric by introducing additional components. The concrete meaning of the half-edge collapse scheme and the proposed error metric are discussed in section 3 and 4, respectively.

3 Simplification Framework

The goals of this paper are to retain the characteristic features of the original model even after drastic simplification process, and to generate a simplified model that is applicable to the incremental multi-resolution model supporting progressive network transmission [2][12]. We describe the structural characteristics of the proposed algorithm for satisfying the goals.

3.1 Notation

Before describing our algorithm, we will briefly introduce some notations. We assume that a polygonal surface model is simply a set of triangular polygons in three-dimensional Euclidean space R^3. An arbitrary polygonal surface model $M=\{V,T\}$ is a set of vertex set $V=\{v_1,v_2,v_3,\ldots,v_m\}$ and triangle set $T=\{t_1,t_2,t_3,\ldots t_n\}$. M_i is a mesh in the i^{th} stage of iterative simplification process and normally means the current mesh. And M_{i+1} is the successive mesh of M_i. An edge $\overline{e}$ is denoted by a set of vertices $\{u,v\}$, where $u,v \in V$. Every edge has a direction. A directional edge $\vec{e}$ is denoted by a set of ordered vertex pairs (u,v). This means that the vertex u will be merged into v after collapse operation. P(u) is a set of planes of the triangles that meet at vertex u and $P(\overline{e})$ is, generally two, a set of planes adjacent to the edge $\overline{e}(u,v)$. Then, $P_i(\vec{e})$ and $P_{i+1}(\vec{e})$ are a set of $\{P(u)-P(\overline{e})\}$ in mesh M_i and M_{i+1}, respectively.

3.2 Half-Edge Contraction

Generally, in the edge contraction-based simplification, there are three cases of placement policies of new generated vertex v' after the edge e(u,v) is contracted, as shown in Fig.2.

First, the midpoint scheme (b) creates a new vertex at the midpoint of the two vertices of an edge. This is intuitive and unbiased to the positions of the two vertices. However, the drawback is the volume of the original object becomes smaller as simplification steps proceed, especially for a convex surface. The optimal point scheme (c) generates a new vertex at the optimal position on the contour curve connecting two vertices of an edge. This scheme can create a high-quality

approximation, but finding an optimal position costs both a great deal of time and extra memory space to store the new vertex. The endpoint approach (d) is the method that places a new vertex by merging one of the two endpoints of the edge to the other. Since no new vertex is created, the endpoint approach gives no additional memory burden and rapid calculation is possible. Moreover, in most cases, the original shape is well preserved [11], although the method does not use the optimal placement policy.

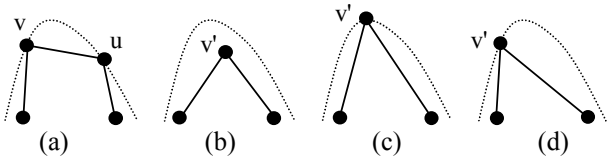
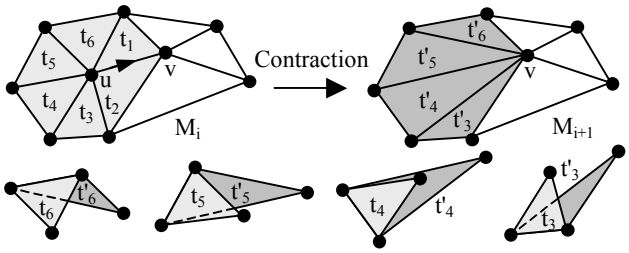


Fig. 2. Vertex placement policies after edge contraction, in two-dimensional space. (a): original submesh, (b): midpoint, (c): optimal point, (d): endpoint

Our algorithm is based on the half-edge contraction scheme, which has the same meaning with the endpoint approach except some implementation details. In terms of implementation of the half-edge contraction, we allow every edge to have directionality. By using the directional edges, we can easily calculate the approximation errors and intuitively process the topological reconstruction of the local surface after the contraction (Fig.3). Moreover, when the half-edge contraction scheme is applied to the rendering applications based on the hierarchical LOD, successive views can be switched smoothly and continuously by utilizing the directional edges.



$$P(u) = \{t_1, t_2, t_3, t_4, t_5, t_6\}, P(\vec{e}(u,v)) = \{t_1, t_2\},$$
$$P_i(\vec{e}(u,v)) = \{t_3, t_4, t_5, t_6\}, P_{i+1}(\vec{e}(u,v)) = \{t_3', t_4', t_5', t_6'\}$$

Fig. 3. Half-edge contraction and geometric variations of local surface before and after the operation. When the directional edge $\vec{e}(u,v)$ is contracted, $P(\vec{e}(u,v))$ and the start vertex u are removed from the current mesh M_i and $P_i(\vec{e}(u,v))$ are adjusted to $P_{i+1}(\vec{e}(u,v))$ in M_{i+1}. With the use of directional edge, we can estimate the degree of geometric deviation after the contraction by reconstructing the local planes in next step temporarily

Another attractive point of the half-edge contraction scheme is that the progressive transmission of meshes [12] in the network environment can be performed very effectively, since the set of vertices in the simplified mesh is always a proper subset of the original mesh. This means that, in every moment, instant rendering of the scene

is possible with no need to wait for the whole data to be transmitted (Fig.4). In the applications, which take seriously the real-time interactivity, this can be a critical problem.

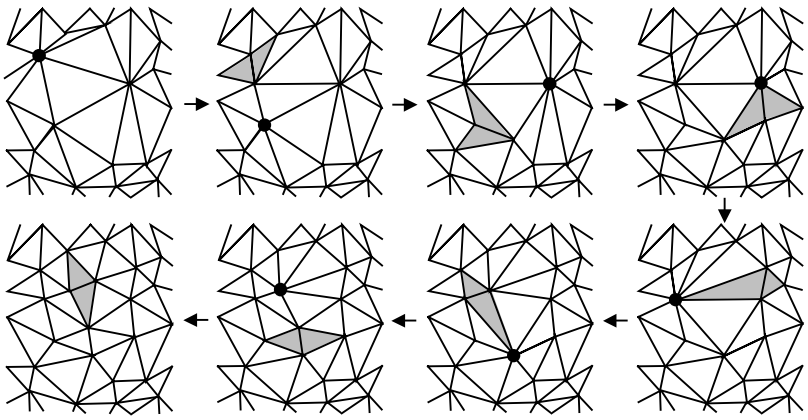


Fig. 4. Progressive refinement of meshes. Filled circle means that the vertex will be split into two vertices (edge) in the next stage. In the half-edge contraction scheme, successive mesh can be generated immediately after receiving one more vertex, while two more vertices in optimal and midpoint approaches. Because, in those approaches, the vertex to be split cannot be reused in the successive mesh as it is not a subset of the mesh

3.3 Memoryless Simplification

Simplification algorithms perform iterative decimation operation in each simplification step depending on the decimation cost calculated from certain criteria. Local geometric changes caused by the removal of elements in each simplification step alter the decimation cost of adjacent elements. Usually, some form of geometric history is kept in the simplified model to reflect the changes and guide the calculation of decimation cost in future steps. This means that an over-estimated error from the previous simplification step will continue to accumulate in future. A straightforward solution to the accumulation of over-estimated errors during the simplification process is to re-compute the decimation cost every time, after each decimation operation, namely the memoryless approach. This imposes a huge computational complexity on the algorithm. Fortunately, however, only a limited number of elements will be affected, i.e. the neighbors of decimated element and their associated neighbors. Consequently, on each update, it is sufficient to recalculate the decimation cost for these neighbors. The memoryless scheme has the following advantages. First, the memory requirements for such an algorithm are smaller than those that require storing a geometric history. Second, memoryless algorithm is typically faster than those that must query a geometric history in order to recalculate the decimation cost [10]. Finally, memoryless simplification improves the accuracy of the results [9]. The proposed algorithm is based on the memoryless approach.

4 Error Metric

In the simplification process, for the assessment of geometric similarity between M_i and its descendent M_{i+1}, we need some means of quantifying the notion of similarity. Error metric is a measure that represents the degree of deviation, or error, of approximation from the original model.

$$E(M_i, M_{i+1}) = G_{planes}(M_i, M_{i+1}) + H_{planes}(M_i) \ . \quad (1)$$

The proposed algorithm defines a geometric error metric that exploits the locality of mesh changes before and after simplification. That is, because the geometric changes in the simplification method based on iterative edge reduction always happens in adjacent areas of the edge, we can estimate the degree of deviation after collapse operation, by appropriate description of the local surface. Also, we observe the degree of geometric variation (G) between M_i and M_{i+1} and geometric characteristic (H) of current mesh M_i as the factors for causing errors during the simplification process. Eq.1 is the combination of those two concepts. The proposed algorithm assigns the decimation cost for every edge calculated from Eq.1 to corresponding edges. We discuss the detailed meaning and sub elements of the factors in the following sections.

4.1 Geometric Variations

As the factors for geometric variations G occur when an edge is collapsed, we consider the amount of distance and changes of surface orientation between meshes before and after simplification. They are factors for detecting the degree of variation in the magnitude of scalar and vector of each local surface, and they work complementary to each other. That is, the degree of variation of surface orientation before and after simplification is independent from that of scalar. Therefore, it is possible to control overall decimation cost by assigning higher cost to the element when the probability of reduction is high due to little change in scalar even when the degree of variation for orientation is high, and vice versa.

Distance. Distance between an arbitrary vertex v and plane t can be calculated using Eq.2. $\vec{v}$ is a vector between v and an arbitrary vertex on plane t, n_t is a normal vector of t, and $\cdot$ is an inner product of two vectors.

$$d(t, v) = \left| \vec{v} \cdot n_t \right| \Big/ \left\| n_t \right\| \ . \quad (2)$$

Then, the distance between M_i and M_{i+1} can be calculated by the sum of distance between plane set $P_i(\vec{e})$ in current mesh M_i and an end vertex v of the directional edge $\vec{e}(u, v)$ in descendent mesh M_{i+1} (Eq.3) (Fig.5). In here, it must be noted that $D(P_i(\vec{e}), v)$ is the sum of distance-to-plane measurement. This is similar to [14] but the difference is that, in this case, only P(u), which is the superset of $P_i(\vec{e})$ and not

$P(u) \cup P(v)$ is considered. Also, the reason why the sum of distance is used rather than maximum or average distance is to avoid sensitive response to geometric noise.

$$D(P_i(\vec{e}), v) = \sum_{t_i \in P_i(\vec{e})} d(t_i, v) \cdot \quad (3)$$

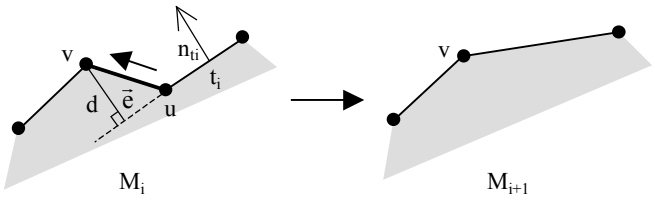


Fig. 5. Calculation of distance between M_i and M_{i+1} during the directional edge $\vec{e}(u, v)$ is contracted in two-dimensional space, where, $t_i \in P_i(\vec{e})$ and n_{ti} is a normal vector for t_i

Orientation. Describing the geometric variation of local surface before and after simplification using just the distance measure is not sufficient. In Fig.6 (a), decimation costs based on the distance metric of every vertex (u_1~u_4) are the same. This is because the distance between vertex v to planes, p1~p4, each containing one of the vertexes are d. But, the amount of deviation before and after simplification are different as shown in (b) and (c). When the orientation variation of planes is considered, this problem can be solved. In other words, when decimation costs of (b) and (c) based on distance metric are the same but the orientation variation of (b) before and after simplification is comparatively larger than (c), higher decimation cost can be assigned to (b).

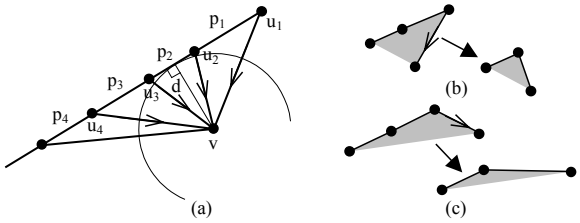


Fig. 6. Cases when the decimation costs based on distance metric are the same, where each arrow means collapse operation

In the mesh M_{i+1}, remaining planes after collapse operation of edge $\vec{e}(u, v)$ are $P(u)$ with $P(\vec{e})$ excluded, which is $P_{i+1}(\vec{e})$. So, estimation of orientation variation only considers the variation among previous planes, $P_i(\vec{e})$ from M_i (Eq.4).

$$O(P_i(\vec{e}), P_{i+1}(\vec{e})) = \sum_{t \in P_i(\vec{e}),\, t' \in P_{i+1}(\vec{e})} \left(\left| 1 - n_t \cdot n_{t'} \right| / 2 \right) \cdot \quad (4)$$

Normal of plane, which is a vector, can be especially sensitive to geometric noise. So, similar to the case of distance metric explained previously, to lower the impact of noise, summation operator is used in Eq.4.

4.2 Geometric Characteristics

As the sub components of H, which is another component of our error metric for detecting geometric features in current mesh M_i, we define both local curvature and edge's length. Since the geometric features of mesh are constructed with small sized elements, which are concentrated in small areas, they have trivial quantity of G. The component of orientation variation O, explained in the previous section, may be used for preserving the features with small scalar value. This approach, however, computes the same decimation cost for those surface areas having different geometric characteristics (Fig.7). So, we must introduce an additional component, such as local curvature of current mesh M_i, to distinguish the areas.

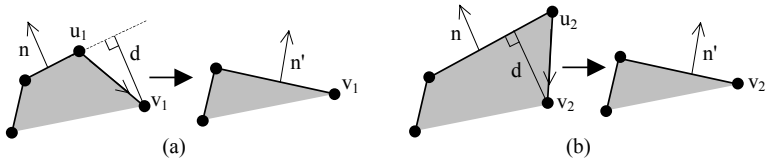


Fig. 7. Cases when the decimation costs based on G of error metric are the same but region curvatures are different

The local curvature C is calculated by the sum of inner product between the plane sets P(u) and $P(\vec{e})$, which are adjacent to start vertex u of edge $\vec{e}$ and the edge itself respectively (Eq.5).

$$C(P(u), P(\vec{e})) = \sum_{t \in P(u)} \underset{t' \in P(\vec{e})}{\text{Min}} \left(|1 - n_t \cdot n_{t'}| / 2 \right) \cdot \quad \textbf{(5)}$$

Eq.5 is similar to the curvature element of edge cost formula in [13]. We, however, use sum instead of max. Roughly speaking, short edges are relatively less important, since they make a low impact on the local surface of the mesh. Consequently, they are assigned with a low decimation cost. This means that the length of the edge should be considered as the additional component of error metric. We completed the final form of our error metric by multiplying the edge's Euclidian length $\|u - v\|$ to the summation of Eq.3, 4 and 5.

5 Experimental Results

For the measurement of numerical accuracy of the simplified model, we use a public tool, namely Metro [16]. Metro is a tool that evaluates the difference between surfaces, i.e. triangulated mesh and its simplified representation. The fundamental

operation of Metro is to calculate both the distances from sampled points of one model to the surface of second model and vice versa. It returns numerical results, such as maximal, mean and mean squared errors.

Fig.8 represents the comparison graph of each simplified result from seven simplification algorithms including our method by using Fandisk model. The Fandisk model is composed of 12,946 triangles and characterized by comparatively uniform size of triangular patches, and apparent distinction of the differences in surface curvature between low and high curvature regions. At the simplification ratio of 99%, the best results are given by the Mesh optimization, and QSlim and the proposed method follow. The Mesh optimization, however, shows low accuracy when the simplification ratio is less than 90%. In addition, this method keeps a large amount of points sampled from original surface to calculate the degree of deviation from the original model in each simplification step. Consequently, it has high cost in memory usage and time for calculating the optimal location of the new vertex [18]. Therefore, we can assume that the results from QSlim, our method and JADE are all fairly acceptable.

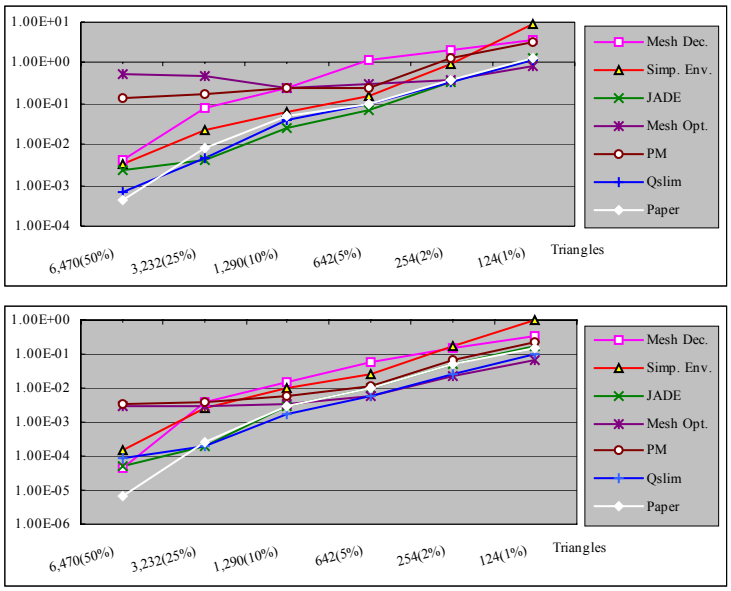


Fig. 8. Maximal (up) and average (down) errors of each simplification method on the Fandisk model. All errors are measured as the percentage of the bounding box diagonal. In here, 'Paper' means the proposed algorithm

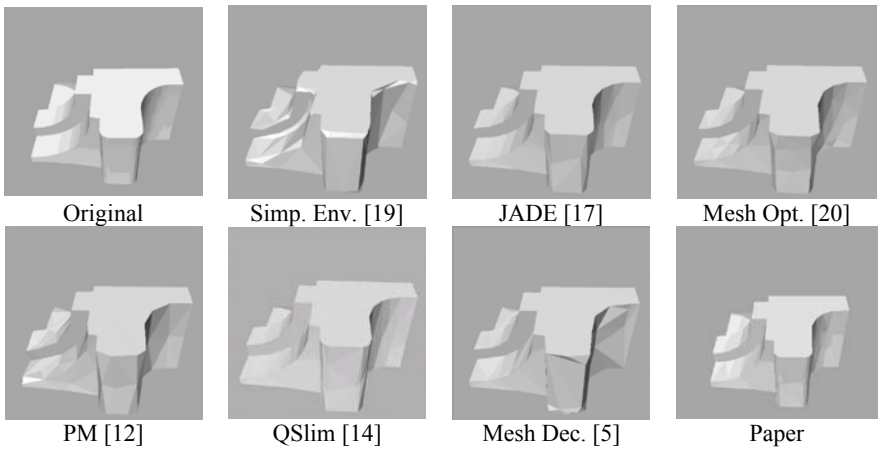
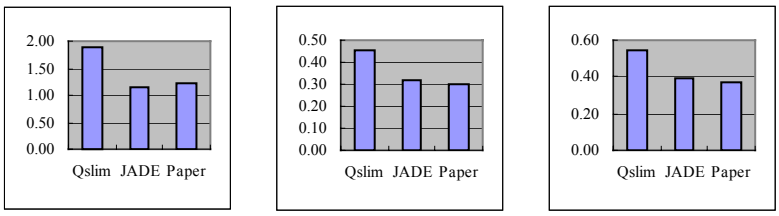


Fig. 9. Visual results from each simplification methods for the Fandisk model.

Fig.9 represents the visual results. In the results of Mesh decimation, QSlim, JADE and ours, we can observe that there are almost no visual differences. Another two surface models used in accuracy comparison, are Venus and Cow. These models are characterized by various sizes of triangular patches, and sophisticated surface curvature (Fig.11). In this case, we compared the results of the proposed algorithm with two previous methods, QSlim v2.0 [14] and JADE v2.1 [17]. They are publicly available and representative methods for vertex and edge decimation methods respectively. Fig.10 shows the numerical results of the three methods by using the Venus model, at the simplification ratio of 99.7%. The proposed algorithm obtains the minimal mean and mean square errors. The best result in terms of maximal error is given by JADE, which is based on the global error management approach. Similar to the previous Mesh optimization, however, this method costs high in memory usage, since it keeps the history of the removed vertices for the global error management. It is noticeable that QSlim has the lowest overall accuracy, though it adopts the optimal vertex placement policy.



(a) Maximal errors (b) Mean errors (c) Mean square errors

Fig. 10. Maximal, mean and mean square errors for the Venus model

Fig.12 and 13 represent the visual results of the simplified models for Cow and Venus obtained from the three algorithms, respectively. In both cases, it is noticed that our algorithm retains the best high curvature region of the original model at the

same simplification ratio. Fig.14 represents the distribution of the approximation errors with gray scale. Lighter color indicates more errors occurred in the area and darker color indicates less error. In the results of the QSlim, large approximation errors are distributed broadly, since it uses the optimal vertex positioning policy. That is, in most cases, since the new generated vertices are not the subset of the original model, approximation errors occurred in almost all areas. While large approximation errors occurred in high curvature regions, such as brow, eyes, nose, lips, in JADE, the proposed algorithm preserves the areas excellently.

The numerical results do not guarantee the accuracy or the quality of the simplified model. Because the most important measure of fidelity is not geometric but perceptual [21]. This means that whether or not the simplified model looks like the original. In terms of the perceptual fidelity, the visual results indicate that the proposed algorithm works well with good shape preservation.

Fig. 11. Original Cow and Venus models. Each of them are composed of 5,804 and 100,000 triangles, respectively

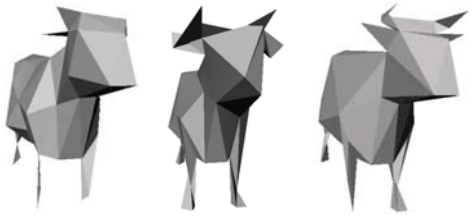


Fig. 12. Visual results for the Cow model with 150 triangles

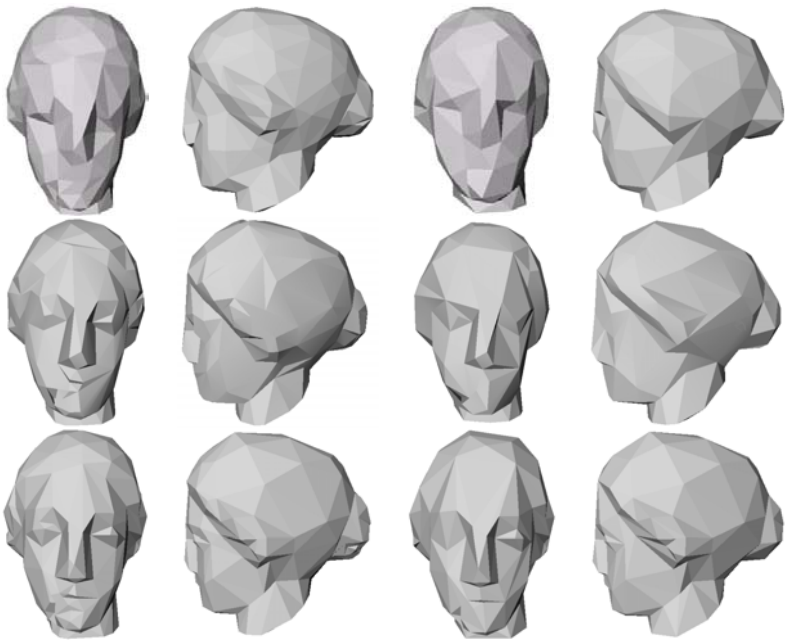


Fig. 13. Visual results for the Venus model. Top: QSlim, middle: JADE, bottom: proposed algorithm, left two: 500 triangles, right two: 300 triangles

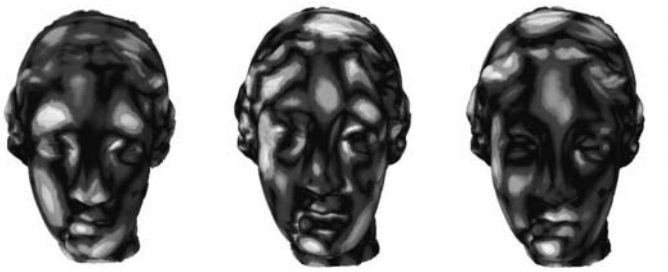


Fig. 14. Error distributions. The simplified Venus model is composed of about 300 triangles. Left: QSlim, middle: JADE, right: proposed algorithm

6 Conclusion

There is no unique simplification method generating the best results for every surface model. This means that it is desirable to select an appropriate method according to the various usages in the applications.

The goals of this paper are to retain the characteristic features of the original model even after drastic simplification process, and to generate various simpler versions of

original complex model, which are applicable to the real-time rendering applications. To satisfy the goals, we proposed an error metric and implemented a surface simplification algorithm utilizing the error metric. In addition, we proposed the topological operation based on the half-edge collapse manner.

The experimental results indicate that the proposed algorithm works well when there is a need to preserve comparatively high curvature regions in the surface model, which is constructed with various local curvatures. That is, the proposed algorithm preserves the details of the original mesh and retains the overall shape even after drastic simplification process. And the half-edge collapse scheme is memory efficient and can be effectively applied to the real-time applications, which require progressive transmission of the large number of surface data.

Acknowledgements

We wish to thank Cyberware, Inc., for the Venus model, Avalon archive for the Cow model, used in the experiments of the algorithm, respectively.

References

1. T. Varady, R. R. Martin, J. Cox: Reverse engineering of geometric models - an introduction. Computer-Aided Design, 29(4), (1997) 255-268
2. D. To, R. Lau and M. Green: A Method for Progressive and Selective Transmission of Multi-Resolution Models. Proc. ACM VRST, (1999) 88-95
3. J. Cohen: Interactive Walkthroughs of Large Volumetric Datasets: Concepts and Algorithms for Polygonal Simplification. Proc. SIGGRAPH 2000 Course Notes, (2000)
4. J. Cohen: Advanced Issues in Level of Detail: Measuring Simplification Error. Proc. SIGGRAPH 2001 Course Notes, (2001)
5. W. J. Schroeder, J. A. Zarge, W. E. Lorensen: Decimation of triangle meshes. Proc. SIGGRAPH'92, 26(2), (1992) 65-70
6. D. Luebke: A Survey of Polygonal Simplification Algorithms. UNC Department of Computer Science Technical Report, (1997) #TR97-045
7. G. Turk: Re-tiling Polygonal Surfaces. Proc. SIGGRAPH'92, (1992) 55-64
8. R. Klein, Gunther Liebich, and W. Straßer: Mesh reduction with error control. Proc. Visualization'96, (1996) 311-318
9. H. Hoppe: New quadric metric for simplifying meshes with appearance attributes. Proc. IEEE Visualization'99, (1999) 59-66
10. P. Lindstrom, G. Turk: Evaluation of Memoryless Simplification. IEEE Trans. Visualization and Computer Graphics, 5(2), (1999) 98-115
11. M. DeLoura: Game Programming Gems. Charles River Media, (2001)
12. H. Hoppe: Progressive meshes. Proc. SIGGRAPH'96, (1996) 99-108
13. Stan Melax: A simple, Fast, and Effective Polygon Reduction Algorithm. Game Developer, (1998) 44-49
14. M. Garland and P. Heckbert: Surface Simplification Using Quadric Error Metrics. Proc. SIGGRAPH'97, (1997) 209-216
15. K. J. Renze and J. H. Oliver: Generalized Unstructured Decimation. IEEE Computer Graphics and Applications, 16(2), (1996) 24-32

16. P. Cignoni, C. Rocchini and R. Scopigno: Metro: measuring error on simplified surfaces. Computer Graphics Forum, 17(2), (1998) 167-174
17. A. Ciampalini, P. Cignoni, C. Montani, and R. Scopigno: Multiresolution Decimation based on Global Error. The Visual Computer, Springer International, 13(5), (1997) 228-246
18. P. Cignoni, C. Montani and R. Scopigno: A Comparison of Mesh Simplification Algorithms. *Computers & Graphics*, Pergamon Press, 22(1), (1998) 37-54
19. 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
20. Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald and Werner Stuetzle: Mesh optimization. Proc. SIGGRAPH'93, (1993) 19–26
21. David P. Luebke: A Developer's Survey of Polygonal Simplification Algorithms, *IEEE Computer Graphics &Applications*, 21(3), (2001) 24-35