Journal of Visualization, Vol. 10, No. 1(2007)111-121

Adaptive Model Simplification in Real-Time Rendering for Visualization

Ng, K.-W.* and Wong, Y.-P.*

* Faculty of Information Technology, Multimedia University, Jalan Multimedia, 63100 Cyberjaya, Selangor, Malaysia. E-mail: kwng@mmu.edu.my, and ypwong@mmu.edu.my

Received 30 June 2006
Revised 8 September 2006

Abstract : In this paper, we propose four different geometric measures to identify appropriate triangles to be simplified in 3D complex model. Each measure yields different weight on the same surface and produces a unique simplified model that worth to be analyzed. The proposed measures involve consideration on the resulting of the surfaces collapse, the high peak and low peak of the triangles mesh, the irregular triangle shape, the capacity and boundary view on the triangles mesh. The chosen triangle is to be collapsed based criterion on Half-edge Collapse Transformation method. From the empirical results, one of the proposed measures presents almost excellence in all the criteria mentioned above. The empirical results include the quality of the surface models (visualization purpose), the efficiency of the measures and the overall appearance preservation of the simplified models. The proposed measures are then to be compared to three existing measures. From the analyzed results, we combine the measures to adapt to the user's response for generating the user-desired simplified models.

Keywords : Visualization, Computer Graphics, Level of Details, Virtual Reality.

1. Introduction

In Computer Graphics, triangle is a popular drawing primitive used to create 3D models. The realism of a model depends a lot on the number of triangles used to construct the model. The more triangles and smaller the triangles are used, the smoother and more realistic the model is. Over the last decade, advances in Model Acquisition, Computer Aided Design and Simulation Technologies have resulted in massive databases of complex polygonal models. Models that are composed of hundreds of million of polygons are in common place to achieve realism and high visual fidelity of the modeled objects. However, the complexity of these models grows faster than the ability of the advanced graphics hardware to render them in real time. Level of Detail Technique attempts to narrow this discrepancy between the model complexity and the hardware performance. In real life, when an object appears relatively small or far away from the viewer, some of the detail object should be unseen or too small to be noticed. These small details can be approximated and replaced by larger polygons without deteriorating the appearance of the model viewed from far. This can also alleviate the burden of the rendering pipeline and thus maintain acceptable frame rates. The method to render fewest numbers of polygons while ensuring the closest model quality to the original model is called Simplification Method (see Fig. 1).

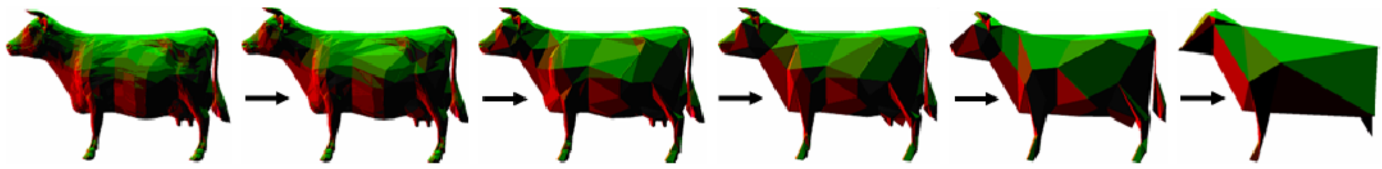


Fig. 1. Different levels of detail of the cow models. The original model (leftmost) slowly reduces its polygons to 40 %, 20 %, 10 %, 5 % and 1 %; yet preserve the overall appearance of the model shape.

In next section, we will discuss the background and some popular methods using vertex, edge and triangle operators. Emphasis will be put on the edge operator because it is computation efficient, model appearance preservative and does not need extra triangulation algorithm to patch the resulting hole. In third section, we will present our proposed algorithm. In section four, we will discuss our proposed measures. This section will give the ideas of how the measures evaluate the triangles mesh and how they preserve the appearance of the original model. Section five will show the empirical results for each proposed measures. The comparison results will also be shown in this section. Section six will discuss our design methodology as to adapt to the user's response. Different weight will be assigned to each measure to achieve a balance scale for the triangles mesh. Last section will be the conclusion of this paper and the references.

2. Background

Generally, simplification methods can be classified into four major categories: Vertex Removal Methods, Vertex Clustering Methods, Edge Collapse Methods and Regional Collapse Method.

Vertex Removal Method is the earliest existing method. Each of the vertices in the polygonal mesh is a potential candidate to be removed. After removing a vertex, a hole will be created. The triangulation algorithm is then being applied to patch the resulting hole. Decimation Algorithm (Schroeder et al., 1992) is a popular Vertex Removal Method. Schroeder (1997) re-modified the method to yield a guaranteed reduction level and enable the modification of the topology model when necessary to achieve the desired result. The work (Schroeder et al., 1992) was also improved by a few research teams (Ciampalini et al., 1997), (Klein et al., 1996) and (Ng et al., 2003). Vertex Removal Method is good at reducing iso-surfaces containing millions of polygons, but the extra triangulation work has slowed down the whole simplification process.

Vertex Clustering Method is a completely different method from Vertex Removal Method. When an object is far away from the viewer, the vertices are mapped on the same pixel. Only one point will be appeared on the image at the pixel. The other hidden vertices will be eliminated by hidden surface removal algorithm. Rossignac and Borrel (1993) is first to propose Vertex Clustering Method. They used a regular, uniform 3D grid on the model and collapsed the vertices within each cell of the grid to a single most important vertex within the cell. Low and Tan (1997) used Floating-cell Clustering Method in which the vertex with the highest weight will be the centre of a new clustering-cell. Unfortunately, Vertex Clustering Method consumes a lot of memory and inefficient for animated objects.

Edge Collapse Method is a popular topic in the recent years. It is a special case of Vertex Removal Method. Edge Collapse Method merges two vertices (say v_1, v_2) at both end of an edge (e_{12}) into a single vertex (v). Hoppe (1996) constructed the progressive mesh to collapse the edges. All the edges that are to be collapsed are evaluated according to their effect on the energy function. Ronfard and Rossignac (1996) conceptualized the display of very complex scenes as where many objects are seen with a range of varying levels of detail. They collapse the edges that are co-planar. Garland and Heckbert (1997) used the work done by Ronfard and Rossignac to be the starting point of their simplification algorithm (QSlim). They measured the squared distance from the collection of planes associated with triangles incident on a vertex and stored them as a symmetric 4×4 matrix which one matrix per vertex. Garland and Heckbert combine the Clustering Operation and Vertex-pair Contraction Operation to re-mesh the resulting holes. However, their method is memory consuming. Lindstrom and Turk (1998) used Edge Collapse Operation to place the new vertices in a manner that helps preserve the location and the shape of the original surface. One drawback of their approach is that it is sensitive to modifications of the mesh connectivity that do not affect the geometry. Hussain et al. (2004) used Half-edge Collapse Operation to simplify the triangles mesh. Their MELOD method took the average areas of the original mesh and the resulting mesh to maintain the highly irregular triangles mesh. They also proposed FMLOD method (Hussain et al., 2003a) to compute the average length from the original mesh; and FMSLOD method (Hussain et al., 2003b) to compute the area of the triangle that is adjacent to the original mesh and the resulting mesh. These methods excel in different triangular mesh. Hussain et al. also compared their methods to some popular simplification methods above. From the results shown in their journal paper (Hussain et al., 2003b), their generated triangles mesh shows as high quality as QSlim method and Ciampalini et al.'s method (based on Mean Geometric Error); whereas Lindstrom and Turk's method produces the most original surfaces. In terms of efficiency, Hussain et al.'s method shows the second fastest after QSlim method. In our survey, so far QSlim method gives the shortest computation time among all the existing simplification methods. However, our proposed methods will be improving on Hussain et al.'s methods in the aspect of quality model and computation time. This is because their method is less memory consuming.

Lastly, the Regional or Triangles Collapse Method is implemented by removing all the adjacent triangles of a triangle and the triangle itself. Hamann (1994) arranged each set of triangles based on curvature weight of the vertices. Triangles that were close to co-planer surface regions were prime candidate for removal. Gieng et al. (1997) presented a hierarchical set of triangle meshes to blend different levels of detail in a smooth fashion through a set of Triangle Collapse Operations. Wu et al. (2001) used statistical idea to introduce terms involving mean and variance of dihedral angles to detect flat neighborhoods which were free from rough features. Triangle Removal Method comes across a lot of restrictions compared to other simplification methods. Thus, it is less popular to be exploited. For extra reading, readers may refer to Oliver et al. (2006) and Miyachi et al. (2005) papers.

3. Proposed Algorithm

Generally, researchers who work on simplification method would share a few common steps in their simplification algorithm. The steps involve evaluation of the input data, elimination of the appropriate data, checking for the criteria and reconstruction of the model with fewer triangles. Below is our proposed algorithm:

(1) Input data. Filter the noise and classify the vertices into two categories.
(2) Accept the user's input parameters
(3) Evaluate the vertices with different methods (proposed in next section).
(4) Update the data structure (with the computed results) and link the associated vertex to its adjacent vertices.
(5) Select the smallest cost and eliminate the associated vertex. Re-update the data structure.
(6) Check the termination criteria. If not satisfy, go to step 3.
(7) Re-assemble the vertices and triangles to form the surfaces.

We first filter the noise while reading the input data to speed up the evaluation process. Here, the noise is referred to standalone vertices and edges in the model. Only triangles are maintained in the storage. In second step, the user needs to key in his desired triangles mesh. A list of choices will be provided. In classification process, we categorize the vertices into Centre vertices and Boundary vertices. Each Centre vertex is surrounded by a complete triangles-fan. The vertices with incomplete triangles-fan are classified as Boundary vertices. Each category will be evaluated with different methods (see next section). The computed cost will be updated in the priority queue (with the smallest cost on top) and the associated vertex will be linked to all the adjacent vertices to ease the process of updating the priority queue. In termination step, the process will only be ended if the user-input percentage for polygonal simplification is achieved.

4. Proposed Methods

As mentioned in the abstract, our proposed methods will take into consideration of the resulting of the surfaces collapse, the high peak and low peak of the triangles mesh, the irregular triangle shape, the capacity and boundary view to the triangles mesh.

4.1 Distance Collapse of Level of Detail Method (DCLOD)

Our first proposed method is given the name *Distance Collapse of Level of Detail* method (DCLOD). For a Centre vertex, the collapsed of a vertex will eliminate one edge and two triangles. Subsequently, the original orientation of the triangles-fan (pyramid-like) will become flat and join to the other Centre vertex (Fig. 2). We classify this orientation into high peak or low peak of the triangles mesh (though it could also be shallow valley or deep valley). We treat all the alteration as peak curvature. See Fig. 3 for the illustration.

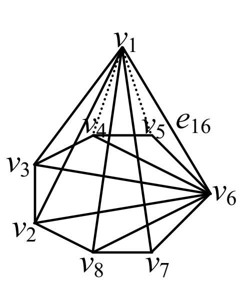


Fig. 2. The original triangles-fan centred at v_1 and the resulting flat triangles-fan centred at v_6.

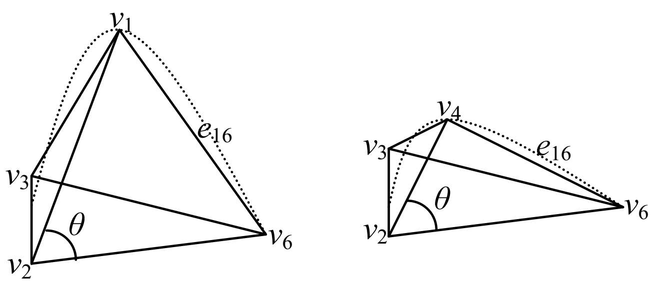


Fig. 3. Vertex v_1 and v_4 located at the peak of the curvature of both triangles mesh. High curvature (left) and Low curvature (right).

The triangles mesh in Fig. 3 is to be seen as the dotted curvature when view from far. Employing the curvature equation such as Bezier Curve, Parametric Equation and etc for calculating the length of the curve is slow and inaccurate to the original mesh. We take the summation of the average length of the triangle t_{132} (refer Fig. 3(left)) and the length of the edge e_{16}. This measure is important because higher the curvature, sharper the bend of the triangles mesh be. Any unexpected change on the sharp bend will seriously harm the original shape of the model. Therefore, low curvature will normally be selected to remove the Centre vertex. To strengthen the weight above, we multiply the length to the dihedral angle, θ for each triangle mesh. The complete formula for the Centre vertex is (Eq. 1):

$$Centre\ Vertex = \sum_{i=0}^{i=n-2}(\theta_i \times |0.5(u_1 + u_2) + u_3|) \tag{1}$$

where $u_1 = v_1\text{-}v_2$, $u_2 = v_1\text{-}v_3$ and $u_3 = v_6\text{-}v_1$, $\theta_i = 1 - n_1 \cdot n_2$. The n_1 and n_2 are the unit normal vectors for the triangles t_{132} and t_{632} respectively. n is the number of adjacent triangles. The formula for measuring the θ is borrowed from Hussain et al. (2003a).

Since the Boundary vertex is an incomplete triangles-fan, the formula for such vertex is alike the Centre vertex formula above but with more weight be added. This is because a slightly change on the boundary mesh will easily be notified. See Fig 4 for triangles mesh at Boundary vertex v_1. Vertex v_1 can either be collapsed onto v_2 or v_9. We will choose the smaller Φ value and remove the associated edge. The Boundary vertex formula is (Eq. 2):

$$Boundary\ Vertex = 50\Phi|v| + 2\sum_{i=0}^{i=n-2}(\theta_i \times |0.5(u_1 + u_2) + u_3|) \tag{2}$$

where Φ is the angle shown in Fig. 4. Vector, $v = v_1$-v_2. The constant values (50 and 2) are to magnify the importance of the Boundary vertex as to be the model outline.

4.2 Triangle Centre of Level of Detail Method (TCLOD)

When the simplification process goes beyond certain levels of detail, the generated triangles mesh tend to be long or bilateral shape. This phenomenon will spoil the smoothness of the surface model. See Fig. 5(left) below, if v_1 is to be collapsed onto v_6, the edges e_{12} and e_{13} will be sheared to be e_{62} and e_{63} respectively (which become longer triangle). Same thing happens to Fig. 5(right) (fat triangle shown in red dotted arrows). These edges create ugly look to the original surfaces. To avoid this to be happen, we introduce an arrow grows from the centre of the base-triangle.

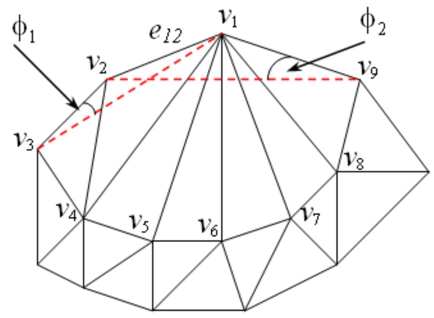


Fig. 4. Boundary vertex v_1.

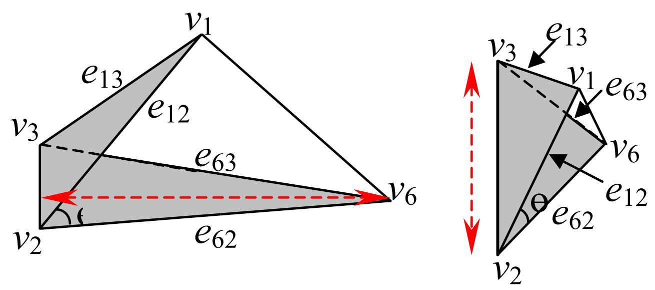


Fig. 5. The long triangles (left) and fat triangles (right) mesh.

Compare Fig. 6(left) and 6(right) below, let the distance $d_1 = d_2$, the position of v_1 in both figures is at the same height from the base-triangle. The size of the base-triangle t_{263} in both figures is also the same, thus, $c_{t1} = c_{t2}$. In simplification process, we will collapse the edge e_{16} in Fig. 6(right) because the angle θ_2 is smaller than θ_1. So that, the created new triangle will not be obviously long as compared to the one happens in Fig. 6(left).

The distance d_i in Fig. 6 can also be served as partially height from the base-triangle t_{263} to the candidate vertex v_1 (the model characteristics) to prevent creating large changes to the original surfaces. The Centre vertex and Boundary vertex formulas are shown below:

$$\textit{Centre Vertex} = \sum_{i=0}^{i=n-2} (\theta_i \times |v_1 - c_t|) \tag{3}$$

$$\textit{Boundary Vertex} = 50\Phi|v| + 2\sum_{i=0}^{i=n-2} (\theta_i \times |v_1 - c_t|) \tag{4}$$

where $c_t = (v_1 + v_2 + v_3)/3$. The rest of the variables are defined the same as in DCLOD method.

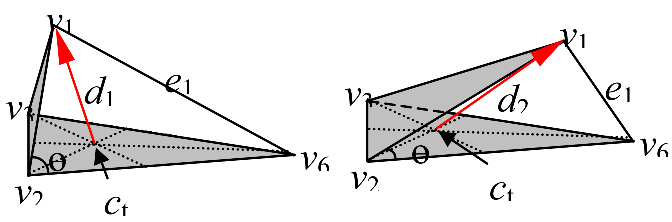


Fig. 6. The Triangle Centre c_t on the base-triangle and the distant d from the Triangle Centre to the candidate vertex v_1.

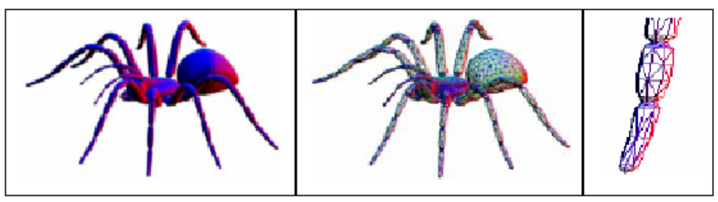


Fig. 7. The Spider model in polygonal mode (leftmost), wire-frame mode (middle) and the triangles mesh of the spider's side-leg (rightmost).

4.3 Volume of Level of Detail Method (VolLOD)

The above methods are good at tackling co-planer surfaces and long triangles in the polygonal mesh. Using such methods to simplify the polygonal mesh that consists of long and slim components is difficult to maintain the characteristics of the model and the model shape. Please take note of the word "long component" in this context as it does not mean long triangle. It is long on the model parts (such as the long legs of the spider model), but the triangles mesh of the components are made up of small and fat triangles. See Fig. 7 above, the spider model is made up of long and slim legs (including the palps: two beards attached to the head). If we apply DCLOD method or TCLOD method to this model, the spider legs will first be simplified because the triangles mesh are flat and make up of many fat triangles.

Since the spider legs are the main characteristics to the model, the dihedral angle at these meshes will be large. However, solely depend on the dihedral angle will bias to those triangles mesh with smaller angle. Hence, we incorporate a few surfaces in a single measure. When the dihedral angle at long components is large, the volume of the components will automatically be large and vice versa. Therefore, we take the volume with respect to the dihedral angle to characterize the long and slim components of the meshes. We use dot product and cross product to find the volume for the Centre vertex and Boundary vertex below:

$$Centre\ Vertex = \sum_{i=0}^{i=n-2} (\theta_i \times |(u_1 \times u_2) \cdot w|) \tag{5}$$

$$Boundary\ Vertex = 50\Phi|v| + 2\sum_{i=0}^{i=n-2} (\theta_i \times |(u_1 \times u_2) \cdot w|) \tag{6}$$

where $u_1 = v_6 - v_2$, $u_2 = v_3 - v_2$ and $w = v_1 - v_2$. The volume computation above is a parallelogram-sided box which is six times to the actual volume of the triangle-pyramid above. We do not take 1/6 for the calculation because it makes no difference since we apply the same measure to all the vertices. Furthermore, multiplying 1/6 to the value above will slow down the computation process.

4.4 Probability of Viewing of Level of Detail Method (PVLOD)

PVLOD method solves mainly the vertices lying on the model's silhouette. We borrow the idea from Low and Tan (1997) and extend it on Half-edge Collapse Transformation method.

The probability of the vertex v_1 to be viewed at silhouette model is Cos (θ /2) (refer to Fig 8a and 8b below). Instead of using the trigonometry function, we convert it to Taylor polynomials of order 4 (will be shown later). Unfortunately, the formula will repeatedly apply the simplification process on the flat triangles-fan and thus drastically change the original shape of the model. Therefore, we introduce extra calculation to stabilize the extreme by weakening its asset but somehow allow partially preserving the sharp surfaces. We multiply the formula by a distance measured from the candidate v_1 orthogonal to the base-triangle. The complete PVLOD formula for Centre vertex and Boundary vertex are:

$$Centre\ Vertex = \sum_{i=0}^{i=n-2} d[Cos(X_i)] \tag{7}$$

$$Boundary\ Vertex = 50\Phi|v| + 2\sum_{i=0}^{i=n-2} d[Cos(X_i)] \tag{8}$$

where $d = |v \cdot (n / |n|)|$, n is the unit normal vector of the associated triangles mesh.

$v = v_1 - u_1$, $Cos(X_i) = 1 - (X_i^2/2) + (X_i^4/24)$, $X_i = \theta/2$

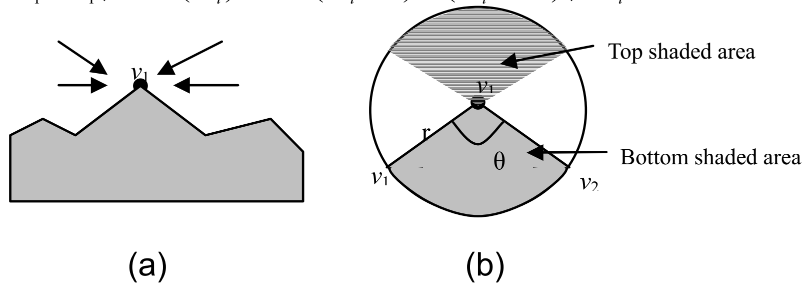


Fig. 8. Vertex v_1 rested at the silhouette in (a). Vertex v_1 situated at the center of the sphere in (b).

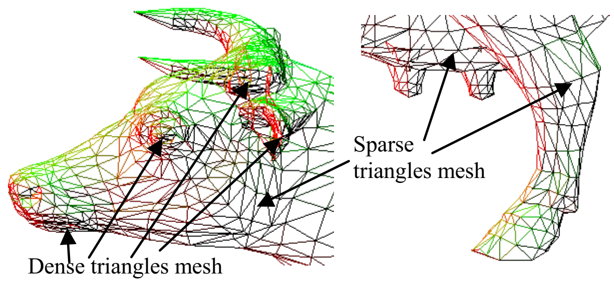


Fig. 9. Dense and sparse triangles mesh of the cow model.

5. Comparison and Results

Our simplification process ran on 1.8 GHz Intel Pentium 4 machine with Windows XP, 256 Mb RAM and Matrox Millennium G200 AGP graphics card. In this paper, we choose to use cow and Stanford Bunny models to be our main experimental models, as they are composed of dense and sparse triangles mesh (for cow model, refer Fig. 9) and Boundary mesh (for Bunny model). Furthermore, they are also consisting of difference shape and components. These models can easily present the strength of our measures and the quality of the simplified models. We use Metro tool (developed by Cignoni et al., 1998b) to evaluate the quality of the surface models.

Besides cow and Bunny models, we have also applied our measures on the Spider, Horse, Cat, Beethoven, Tri-Cycles, Porsche, Dragon and Happy Buddha models. However, due to limited space in this paper, we are not able to show all their results here. Readers may refer to the author's website at http://pesona.mmu.edu.my/~kwng/research.htm (under the title "Simplified models") for the samples of the simplified models.

Figure 10 and Fig. 12 shows the results based on Mean Geometric Error of the simplified cow and Bunny models using our proposed measures and the three existing measures (discussed in Section 2). From Fig. 10, DCLOD method produces the largest errors (which mean the roughest surfaces). This is followed by TCLOD, FMLOD, FMSLOD, VolLOD, MELOD and PVLOD method; whereas in Fig. 12, the results in the order of roughest to smoothes is DCLOD, FMSLOD, MELOD, VolLOD, TCLOD, FMLOD and PVLOD method. These experiment results show that PVLOD method outperforms all the other methods; whereas DCLOD method gives the poorest result.

Figure 11 and Fig. 13 shows the time taken (in second) to simplify cow and Bunny models. Obviously, PVLOD method takes the shortest time to generate the simplified models in both the figures. It is then followed by FMSLOD, MELOD, VolLOD, FMLOD, DCLOD and TCLOD method in Fig. 11; whereas in Fig.13, it is followed by FMLOD, FMSLOD, MELOD, VolLOD, TCLOD and DCLOD method.

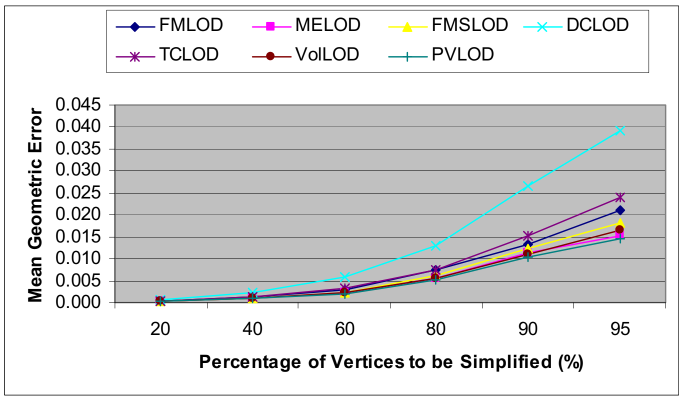


Fig. 10. Mean Geometric Error versus Percentage of Vertices to be simplified for cow model.

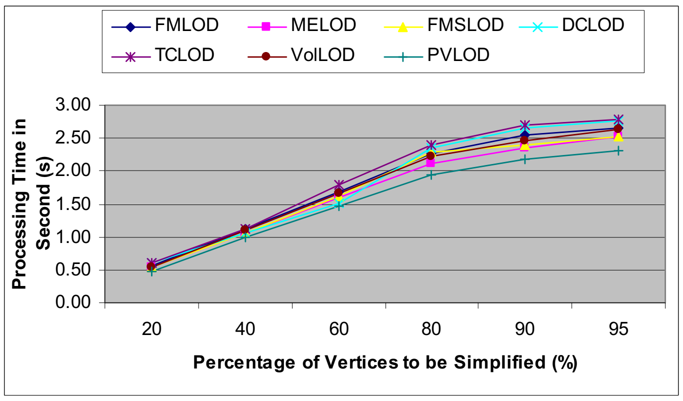


Fig. 11. Processing Time versus Percentage of Vertices to be simplified for cow model.

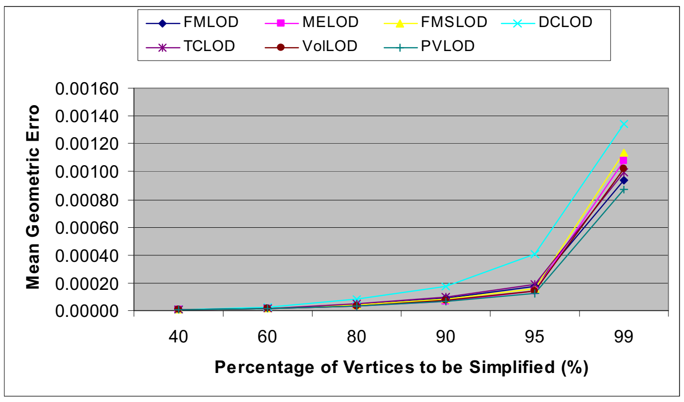


Fig. 12. Mean Geometric Error versus.

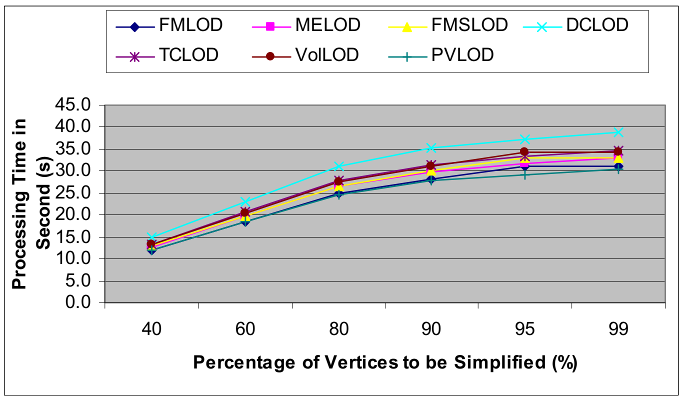
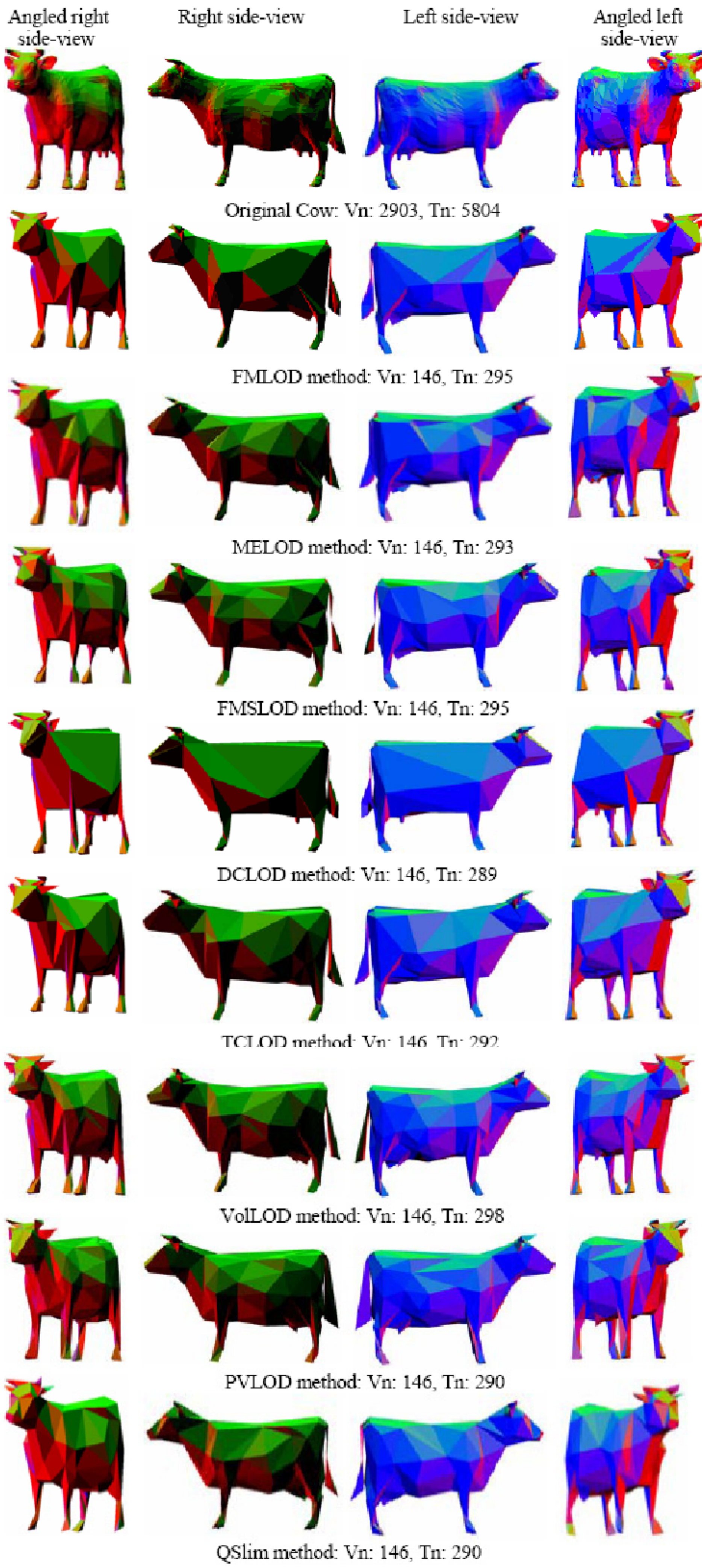


Fig. 13. Processing Time versus Percentage of Vertices to be simplified for Bunny model.

Figure 14 shows the simplified cow models at 95 percent lost from the total vertices. In this figure, the simplified cow models are presented in four different views: angled right side-view, right side-view, left side-view and angled left side-view. The abbreviate Vn stands for number of vertices and Tn stands for number of triangles. In the figure, the first row is the original cow model. The following three rows are generated with the existing methods. The next four rows of the simplified models are generated with our proposed methods. The last row of the simplified models is generated with QSlim method.

From our observation, the body cow on FMLOD method is quite rough and its horns are blunted. The left horn on MELOD method is missing. The tail is broken on FMSLOD method. On DCLOD method, the body cow is uneven and rough. However, the horns and its udders are



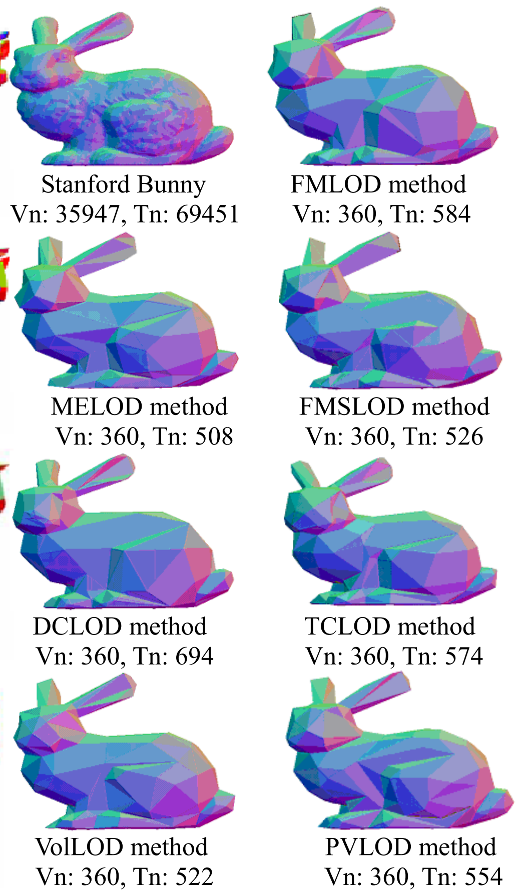


Fig. 15. Simplified Bunny models at 99 % of the vertices lost.

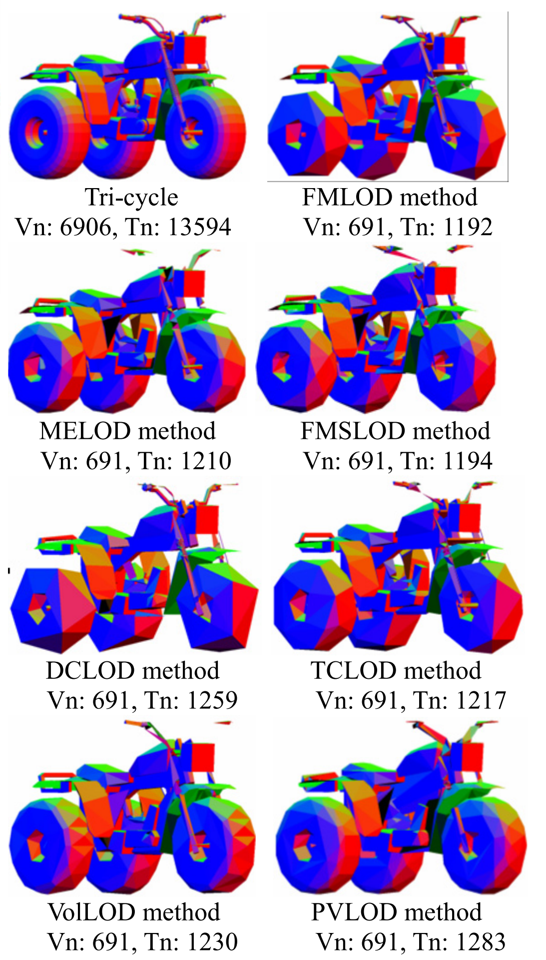


Fig. 16. Simplified Tri-cycle models at 90 % of the vertices lost.

Fig. 14. Simplified Cow models at 95 % of the vertices lost.

preserved well. The body cow on TCLOD method is slightly better than DCLOD method. The tail on VolLOD method is straight and the horns are short. The simplified cow model by PVLOD method looks good but the tail is thick. Lastly, the simplified cow by QSlim method lost both its horns and the body cow is slightly rough.

In Fig. 15, the generated Bunny ears by DCLOD, TCLOD and FMLOD methods look as good as the original Bunny's ears. Other characteristics for this model are slightly hard to be compared visually because all its components (legs) stick together with the Bunny body. Besides these models, we have also shown the simplified Tri-cycle models (a non-living object) for visualization purpose. Readers may examine the smoothness of the tyres and the number of triangles being simplified in Figure 16 with different simplification methods.

6. Adaptive Simplification

After having a full understanding on the performance and the strength of each of the proposed simplification methods, we combine the four methods to adapt to the user-input parameters for generating the desired simplified model. There are two important points to be considered in order to design a program to adapt to the user's response.
(1) What could be the user-input parameters?
(2) How are the methods to be combined?

To answer the first point, from the survey papers (David, 2001; Cignoni et al., 1998; Garland and Heckbert, 1997) and the discussion with some researchers in this field, the user will generally look for three different results (solutions) in any simplification methods.

The first result falls on the quality model. The user may like to have a simplified model that is smooth and displays a realistic look of the surface model. Some important characteristics model has to be sacrificed in order to maintain the smoothness of the model surfaces.

The second result falls on the characteristics model. Some users may keen to preserve the important features of a model. This is significant in the field of Medical Visualization whereby the users focus only on certain imperative features. The rationality of the whole model may not be the first priority to them. Therefore, the simplified model may look coarse but the important characteristics of the model are preserved well.

The third result falls on the speed of the simplification process. The user may concern more on the processing time than the quality model or the characteristics model. Given that the model preserves the minimum required characteristics model, the user may want to quickly generate the simplified model with constant frame-rate. Thus, the pace is to be the centre point for the user's favor. We match the criteria above to our proposed methods as shown in Fig. 17 below (top portion).

For second point, since we allow the user to input his desired weights for each method, we need to ensure that the methods do not dominate one and the others when combining them into one. We have introduced an error adjustment (ratio) for each method based on their generated weight of dominant. The error adjustment is shown in Table 1 below. These ratios will be multiplied to the computed costs before the associated vertices be passed to the elimination step.

We do not display the generated weights of each method in this paper because the generated empirical results are very long and it is not the central here. Next, the user may also combine the four proposed methods with different percentages to generate the blended model. Example of the program is shown in Fig. 17 (the middle and last portions). The output of the combination methods is shown in Fig. 18. Though the body cow is slightly rough, overall the important characteristics of the cow model are preserved nicely.

7. Conclusion

In this paper, we have proposed four simplification methods to reduce the complexity of the polygonal mesh. From the output models shown in Fig. 14, 15 and 16 (and many more in our website), our proposed DCLOD method excels at preserving almost all the importance features of a model; TCLOD method excels at creating non-irregular triangles shape; VolLOD method excels at maintaining long components of a model; and PVLOD method excels at creating smooth surfaces on the curvature mesh. Also, we have successfully created a program which adaptable to the user-desired model shape. The user may have full control over the methods and may input any percentage to obtain different levels of detail from the input polygonal mesh.

Table 1. Ratio for error adjustment on each method.

Method	DCLOD	TCLOD	VolLOD	PVLOD
Ratio	1	15	16000	2

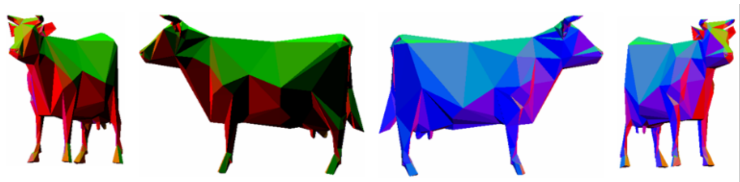


Fig. 18. Simplified cow models with 90 % of the vertices lost.

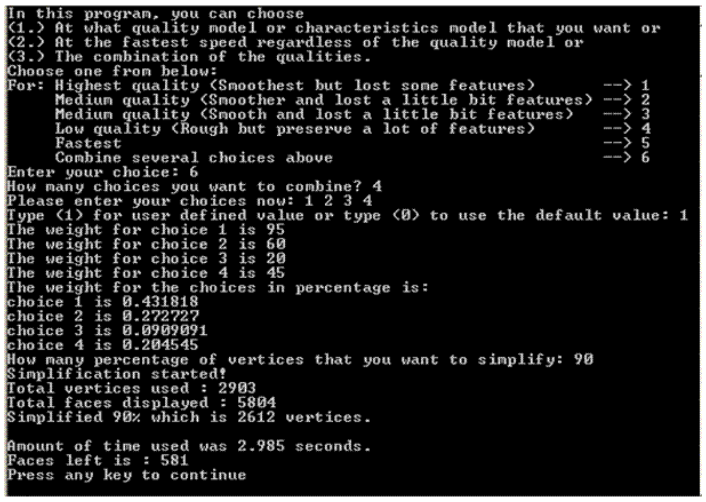


Fig. 17. Options for selecting the criteria for the desired output simplified model.

Acknowledgments

We would like to thank Associate Professor Muhammad Hussain and Associate Professor Peter Lindstrom for their comments. Also, many thanks to the anonymous reviewers for improving the quality of this paper.

References

Ciampalini. A., Cignoni, P., Montani, C. and Scopigno, R., Multiresolution Decimation based on Global Error, The Visual Computer, 13 (1997), 228-246.

Cignoni, P., Montani, C. and Scopigno, R., A Comparison of Mesh Simplification Algorithms, Computers and Graphics, 22-1 (1998a), 37-54.

Cignoni, P., Rocchini, C. and Scopigno, R., Metro: Measuring Error on Simplified Surfaces, Computer Graphics Forum, 17-2 (1998b), 167-174.

David, L., A Developer's Survey of Polygonal Simplification Algorithms, IEEE Computer Graphics and Applications, Tutorial paper, (2001).

Garland, M. and Heckbert, P. S., Surface Simplification Using Quadric Error Metric, Proc. SIGGRAPH '97, (1997), 209-216.

Gieng, T. S., Hamann, B., Joy, K. I., Schussman, G. L. and Trotts, I. J., Smooth Hierarchical Surface Triangulations, Proc. IEEE Visualization '97, (1997), 379-386.

Hamann, B., A Data Reduction Scheme for Triangulated Surfaces, Computer Aided Geometric Design, 11 (1994), 197-214.

Hoppe, H., Progressive Meshes, Proc. Siggraph '96, ACM SIGGRAPH, (1996), 99-108.

Hussain, M., Okada, Y. and Niijima, K., A Fast and Memory-Efficient Method For LOD Modelling of Polygonal Models, Proc. GMAG '03, IEEE Computer Society (London), (2003a), 137-142.

Hussain, M., Okada, Y. and Niijima, K., Fast, Simple, Feature-Preserving and Memory-Efficient Simplification of Triangle Meshes, International Journal of Image and Graphics, 3-4 (2003b), 1-18.

Hussain, M., Okada, Y. and Niijima, K., Efficient and Feature-Preserving Triangular Mesh Decimation, Journal of WSCG, 12-1 (2004), 167-174.

Low, K. L. and Tan, T. S., Model Simplification Using Vertex-Clustering, Symposium on Interactive 3D Graphics ACM SIGGRAPH, (1997), 75-82.

Lindstrom, P. and Turk, G., Fast and Memory Efficient Polygonal Simplification, Proc. IEEE Visualization '98, (1998), 279-286.

Klein, R., Liebich, G. and StraBer, W., Mesh Reduction with Error Control. In Yagel, R. editor, Visualization 96, ACM, (1996),

Miyachi, H. and Sakamoto, N., Data Reduction by Applying an Image-Based Modeling and Rendering Technique to CG Models, Journal of Visualization, 8-4 (2005), 331-338.

Ng, K. W., Wong, Y. P., and Ho, S. N., Improvement in Decimation of Triangle Meshes for Level of Detail, International Conference on Geometric Modelling and Graphics, IEEE Computer Society (London), (2003), 123-128.

Oliver M. K. and Helio P., A Comparative Evaluation of Metrics for Fast Mesh Simplification, Computer Graphics Forum, 25(2) (2006), 197-210.

Rossignac, J. and Borrel, P., Multi-resolution 3D Approximations for Rendering Complex Scenes, Geometric Modelling in Computer Graphics, Eds. B. Falcidieno and T. L. Kunii, (1993), 455-465, Springer Verlag, Genova, Italy.

Ronfard, R. and Rossignac, J., Full Range Approximation of Triangular Polyhedra, Computer Graphics Forum, Proc. Eurographics '96. 15(3), (1996), 67-76.

Schroeder, W. J., Zarge, J. A. and Lorensen, W. E., Decimation of Triangle Meshes, Computer Graphics, ACM SIGGRAPH, 26(2) (1992), 65-70.

Schroeder, W. J., A Topology Modifying Progressive Decimation Algorithm, IEEE Visualization, (1997), 205-212.

Wu, J. H., Hu, S. M., Tai, C. L. and Sun, J. G., An Effective Feature-Preserving Mesh Simplification Scheme based on Face Constriction, Proc. Pacific Graphics, (2001).

Author Profile

Ng Kok-Why: He received his B.Sc. (Math) in 2001 from Universiti Sains Malaysia (USM) and M.Sc. (IT) (by research) in Computer Graphics in 2006 from Multimedia University (MMU), Malaysia. At present, he is pursuing his Ph.D (IT) and giving lecture in Faculty of Information Technology in Multimedia University, Malaysia.

Wong Ya Ping: He received his B.Sc. (Math.) and M.Sc. (Computer Science) in 1992 and 1998 respectively from Universiti Sains Malaysia. He is now lecturing in Multimedia University, Malaysia. His current research interests include Computational Science, Computer Graphics, Virtual Reality, Computer Vision and Intelligent Computing.