Volume and Normal Field Based Simplification of Polygonal Models*

MUHAMMAD HUSSAIN
Department of Computer Science
College of Computer and Information Sciences
King Saud University
Riyadh, 11451 Saudi Arabia
E-mail: mhussain@ksu.edu.sa

In automatic decimation of polygonal models, the measure of geometric fidelity plays the key role. Among the existing measures, volume based error measure results in better quality approximations whereas the one based on normal field variation better preserves salient features. We exploit both volume and normal field to develop a more reliable and efficient two phase greedy algorithm. In the first phase, the priority of each vertex is defined using the normal field variation across its one-ring neighborhood. In the second phase, the desired number of vertices is removed according to their priorities. Once a vertex is a candidate for removal, it is eliminated by collapsing an outgoing half-edge that is selected by using the measure of geometric fidelity based on volume loss. Subjective and objective comparisons validate that the proposed algorithm not only has better speed-quality trade-off but also consumes less memory space and keeps visually important features even after drastic simplification in a better way than the similar state-of-the-art best algorithms. This method is useful for applications where computing coordinates and/or attributes other than those attached to the original vertices is not allowed by the application and the focus is on both speed and quality of LODs.

Keywords: polygonal models, polygonal simplification, LOD modeling, multiresolution modeling, half-edge-collapse

1. INTRODUCTION

Recent advances in technology and the quest of realism have given rise to highly complex polygonal models for encoding 3D information. Despite the enhancement in graphics acceleration techniques and network bandwidth, it is hard to process, transmit and render such models because their complexity has gone beyond the throughput of current graphics systems. The solution of this problem is simplifying a mesh to get rid of redundant information and to create mesh instances with different levels of detail. Based on the application requirements, existing techniques can be classified into three categories: quality simplification [1, 2], fast simplification [3, 4], and simplification with best speed-quality trade-off [5-9]. The proposed algorithm is concerned with the class where the focus is on better quality of approximations and less simplification time, and re-computing the vertex attributes such as texture coordinates, colors, etc. is not allowed or there is no straightforward method of interpolation.

So far, QSlim [6] and Memoryless Simplification (MS) [7] are considered as the best algorithms in respect of speed-quality trade-off. QSlim has relatively high memory

Received January 25, 2011; accepted June 29, 2011.
Communicated by Nancy M. Amato.
* This work was supported by the Research Center of College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia. The authors are grateful for this support.
overhead and MS is memory efficient, but both the algorithms can’t preserve important shape features, especially after drastic reduction of polygon count. For example, when David model consisting of more than seven million triangular faces is simplified to 40,000 faces using QSlim and MS, eyes of David model are not properly preserved, see Fig. 1. The normal field based algorithm (FMLOD) [8] produces LODs of acceptable quality and is relatively better in preserving visually important features but it is not as fast as QSlim and MS. Volume based error measure [7] results in better quality approximations whereas the one based on normal field deviation [8] better preserves visually important features. The proposed algorithm – VolSIMP – exploits both volume loss and normal field variation in a novel way so that the better quality LODs are created in less time and the detail features are effectively retained. Unlike FMLOD, QSlim and MS that use one phase greedy approach where decimation proceeds by selecting and collapsing half-edges according to their cost, VolSIMP employs two phase greedy procedure; in the first phase, vertices are ordered according to their importance, and the required number of vertices is removed in the second phase by collapsing the least significant respective incident half-edges. The priority of a vertex is defined using the normal field variation across its one-ring neighborhood. After a vertex is liable to be removed according to its priority, it is eliminated by collapsing one of the outgoing half-edges; to decide which of the outgoing half-edges is to be collapsed, a measure of geometric deviation is used that is based on volume loss caused by collapse. A quantitative comparison reveals that VolSIMP generates better quality LODs in terms of symmetric Hausdorff distance in less time than FMLOD, QSlim and MS. Its memory over-head is small like MS; it is comparable with FMLOD in preserving significant shape features automatically but better than QSlim and MS. Initially the idea was present in [10].

The remainder of this paper is organized as follows. Section 2 is devoted to related work. In section 3, we discuss the new measures of geometric distortion. Section 4 describes the two phase simplification algorithm. Quality and efficiency of the algorithm is discussed in section 5. Section 6 concludes the paper.

**Notations:** To fix the ideas and for the sake of compactness, we represent one-ring neighborhood of the vertex \( v \) by \( N_v \), the set of triangular faces in \( N_v \) by \( NF_v \), the set of half-edges incident on \( v \) by \( NHE_v \) and the set of vertices in \( N_v \) by \( NV_v \).

**2. RELATED WORK**

The importance of mesh simplification motivated the research on this problem and a large number of algorithms exist. For a thorough survey, an interested reader is referred
to [11-15]. In the sequel, we give an overview of the related simplification algorithms.

Though QSlim [6] was proposed in 1997, it still has better quality-speed trade-off and serves as a benchmark for new proposals. This algorithm employs quadric error metric (QEM). In spite of having better quality-speed tradeoff, it can’t automatically preserve surface discontinuities and visually important features, in particular, at very low levels of detail. Also, it is not memory efficient; for each vertex it adds a memory overhead of at least 40 bytes. Memoryless simplification (MS) [7] uses a memoryless version of QEM, which is derived from volume constraints, and generates good quality LODs but it also does not effectively retain detail features of a model after drastic simplification. The algorithms by Kim et al. [16], Yoshizawa et al. [17], Lee et al. [18] and Yan et al. [2] incorporate additional heuristics to QEM for tackling its drawbacks. Although these algorithms preserve salient features, the execution time increases drastically.

Normal field has also been used for simplification; it has been used either to define an error measure for driving the decimation procedure [8, 19-21] or to perform clustering/sampling [4, 22, 23]. FMLOD [8] uses the deviation between current and original normal field to decide whether to collapse a half-edge. It overestimates geometric error because the normal field deviation over supporting sub-mesh of a vertex is computed considering the normal vectors of the surviving faces after collapse and their counterparts in the original mesh, which do not form contiguous sub-mesh; some of the original faces incident to the vertex may be eliminated during the decimation process and others may be incident on it. Ramsey et al. [20] use normal field variation along with a threshold value for selecting edges for collapse. It does not generate LODs of good quality because of significant volume loss. GeMS [21] uses an error metric that is a multiple of maximum normal deviation and local volume loss caused by half-edge collapse.

All these methods select half-edges according to their significance and collapse them. In contrast, VolSIMP selects vertices according to their importance, and removes them by collapsing the least significant respective incident half-edges.

The algorithm by Brodsky et al. [22] clusters faces using normal field variation; each cluster is replaced by its representative vertex and the model is re-triangulated. This approach is fairly efficient in running time but the constructed LODs are of poor quality, for example, when Bunny model (#faces 69,451) is simplified to 1600 the mean Hausdorff distance is 0.155, which is 0.071 in case of QSlim. Cohen-Steiner et al. [23] introduced an error metric that is based on normal variation, and employed it in their global non-linear optimization technique for finding the best $n$ polygon subsampling of the detailed mesh. TopStoc [4] is based on stochastic sampling and topological clustering, and uses normal field variation for defining the probability of survival of a vertex. This method is computationally very efficient but constructs LODs of poor quality.

3. ERROR MEASURES

VolSIMP selects vertices according to their importance for removal; after selection, a vertex is removed by collapsing one of its outgoing half-edges. In this section, first the measure of importance of a vertex is elaborated, then detail of the error measure, which finds the outgoing half-edge resulting in minimum volume loss, is presented.
3.1 Normal Field-Based Error Measure for Vertex Selection

Normal field of a surface model plays fundamental role in its appearance and it has been used for constraining the geometric distortion in different geometry processing tasks [4, 23-25]. The Poincar-Wertinger-Sobolev inequality implies that minimizing the normal field distortion ensures the minimization of the geometric deviation [24]. Normal field variation truly represents the importance of a vertex. In view of this evidence in support of the strength of normal field, we use the normal field variation across one-ring neighborhood of a vertex for defining its importance.

Surface curvature at a vertex determines its importance and characterizes small detail features of a surface. One common measure of surface curvature is normal curvature, which is defined for infinitely small region around a point and is a good description of surface characteristics. However this measure does not work well in case of larger surface regions with multiple scales of curvatures. In this case, the only choice is to use discrete approximations of differential curvatures, which are usually prone to suboptimal results. For better results, local approximations must adapt to anisotropy asymptotically [23]. One such approximation can be defined as normal field variation across one-ring neighborhood of a vertex as follows:

$$\text{VC}(v) = \sum_{t \in NF_v} \int_{s \in \Delta_t} \left\| n_t(s) - n_v \right\|^2 ds,$$

(1)

where $n_t$ and $n_v$ are, respectively, the unit face and vertex normal vectors of the face $t$ and the vertex $v$. This approximation better captures the anisotropy of a surface [23].

Alternatively, the normal field variation over $NF_v$ can be defined more simply as follows:

$$\sum_{t \in NF_v} \int_{s \in \Delta_t} n_t(s) ds = \sum_{t \in NF_v} \Delta_t n_t,$$

where $n_t$ and $\Delta_t$ are, respectively, the unit face normal and the area of the triangular face $t$. According to triangular inequality

$$\left\| \sum_{t \in NF_v} \Delta_t n_t \right\| \leq \sum_{t \in NF_v} | \Delta_t n_t |.$$

Since $| \Delta n_v | = \Delta_v$, so

$$\left\| \sum_{t \in NF_v} \Delta_t n_t \right\| \leq \sum_{t \in NF_v} \Delta_t | n_t | \quad \text{or} \quad \sum_{t \in NF_v} \Delta_t - \sum_{t \in NF_v} \Delta_t n_t \geq 0.$$

(2)

In case $v$ is flat \textit{i.e.} all faces in $NF_v$ are coplanar, and then left-hand-side of inequality (2) is zero, otherwise it is greater than zero depending on how much the vertex $v$ departs from being flat. This expression defines a measure of significance of a vertex as follows:
\[ VC(v) = \sum_{n \in NF_v} \Delta_n - \left\| \sum_{n \in NF_v} \Delta_n n \right\| , \]  \hspace{1cm} (3)

where the summation is over all faces in \( NF_v \). Note that \( VC(v) = 0 \) when a vertex is flat and it gets larger values according to its level of significance. This discussion leads to the following proposition:

**Proposition 3.1**

(a) \( VC(v) \geq 0 \).

(b) A vertex is not significant iff \( VC(v) = 0 \).

(c) Level of significance of \( v \) is directly proportional to \( VC(v) \).

The value of \( VC(v) \) is used as the priority of a vertex. Now consider that the Eq. (1) can be written as follows:

\[ VC(v) = \sum_{n \in NF_v} \int (n(s) - n_v)(n(s) - n_v)^T ds. \]

Using the fact that \( n(s) \) and \( n_v \) are unit vectors, and the approximation that \( n(s) \) is same over the triangle \( t \), this equation takes the form:

\[ VC(v) = 2 \left( \sum_{n \in NF_v} \Delta_n - \sum_{n \in NF_v} \Delta_n n_v^T \right). \]  \hspace{1cm} (4)

Note that the second term on the right hand side of the Eq. (3) \( i.e. \left\| \sum_{n \in NF_v} \Delta_n n \right\| \) is the magnitude of the projection of the neighborhood \( NF_v \) on the plane defined by the vertex \( v \) and \( n_v \), the unit vector along the vertex normal \( n_v = \sum_{n \in NF_v} \Delta_n n \). The second term on the right hand side of the Eq. (4) \( i.e. \sum_{n \in NF_v} \Delta_n n_v^T \) is another form of the magnitude of the projection of the neighborhood \( NF_v \) on the plane defined by the vertex \( v \) and \( n_v \). This discussion leads to the following conclusion:

**Proposition 3.2:** The normal field based error metrics defined by the Eqs. (1) and (3) are equivalent.

Cohen et al. [23] use this metric for defining shape proxies whereas we use it as a measure of geometric deviation for selecting vertices for removal. According to Cohen et al. [23] it better captures the anisotropy of a surface, and so helps better preserve salient features of a surface after simplification. The form of this error measure given by Eq. (3) is computationally more efficient because its evaluation involves less number of operations.

### 3.2 Volume-Based Error Measure for Half-Edge Collapse

Once a vertex is liable to be removed according to its cost (3), it is eliminated by collapsing one of its outgoing half-edges (see Fig. 2 (b)). Half-edge collapse is in fact
chosen as a strategy for filling the hole created by vertex removal. Other optimal ways for hole filling can be adopted but with drastic increase in computational cost. This decimation operation is simple to implement, is easy to invert and requires less information for inversion, so it is suitable for applications like progressive transmission and visualization of 3D information across networks. It can also simplify when there are vertex attributes that have no straightforward method of interpolation. For instance, many meshes come with vertex normals, texture coordinates, colors, etc., and half-edge collapse frees one from having to re-compute such attributes at the position of the new vertex.

Fig. 2. (a) Half-edge collapse \( h_i: (v, v_i) \rightarrow v \) and the volume loss caused by a typical triangle \( t(v, v_1, v_2) \); (b) Half-edges with \( v_0 \) as origin.

A vertex can be removed by collapsing any one of the outgoing half-edges (see Fig. 2 (b)); we eliminate it by collapsing the half-edge \( h_i \in NHE_v \) that causes minimum local geometric distortion; this half-edge is termed as optimal half-edge and is denoted by \( h_o \).

For finding \( h_o \) we employ a measure similar to the one given in [7] that ensures minimum volume loss. A half-edge collapse \( h_i(v, v_i) \rightarrow v \) causes each triangle \( t \in NF_v \) to sweep out a tetrahedron, see Fig. 2 (a). The volume of this tetrahedron represents the volume loss due to the movement of \( t \) as a result of half-edge collapse and is indicative of the geometric deviation. As such the geometric deviation introduced due to face \( t(v, v_1, v_2) \) is defined as follows

\[
TC(t) = \frac{1}{6} [(v_1 - v) \times (v_2 - v) \cdot (v_i - v)]^T.
\]

In view of this, the geometric error introduced due to half-edge collapse \( h_i(v, v_i) \rightarrow v_i \) is defined as follows,

\[
HC(h_i) = \sum_{t \in NF} TC(t).
\]  \hspace{1cm} (5)

For each half-edge \( h_i \in NHE_v \), we compute \( HC(h_i) \) and choose the one as optimal half-edge \( h_o \) for which \( HC(h_o) \) is minimum. Note the difference between our idea of using volume measure and that of Lindstrom [7]; Lindstrom uses it for global ordering of edges whereas we employ it for local decision of determining the optimal half-edge once a vertex is chosen for removal.
4. TWO PHASE GREEDY SIMPLIFICATION ALGORITHM

The proposed algorithm employs the greedy framework similar to the one introduced in [26]. It involves two degrees of freedom to be fixed: to select a vertex for decimation and to determine the corresponding optimal half-edge. The first phase fixes the vertex to be eliminated. For this purpose, vertices are ordered according to their priorities which are computed using the Eq. (3). The second phase finds the optimal half-edge \( h_o \) corresponding to a selected vertex using the metric specified by the Eq. (5). This process of removing vertices continues until the desired count of faces is reached. The precise description of the algorithm is given below.

**Algorithm VolSIMP**

**Input:** \( M = (V, F) \), the original triangular mesh consisting of the set of vertices \( V \), and the set of faces \( F \), and \( num_f \), the target number of faces,

**Output:** An LOD with \( num_f \) faces

**Processing**

Phase-1: Order vertices according to their importance

For each vertex \( v \in V \), compute its priority using Eq. (3) and insert it into a vertex heap \( VH \).

Phase-2: Select vertices for removal, find respective optimal half-edges and collapse

(a) Remove the lowest priority vertex \( v \) from \( VH \)

(b) For each \( h_i \in NHE_v \), compute its cost using Eq. (5) and select \( h_o \in NHE_v \) such that \( HC(h_o) = \min \{ HC(h_i) \mid h_i \in NHE_v \} \).

(c) Collapse \( h_o: (v, v_o) \rightarrow v_o \), if it does not create fold-overs, by eliminating faces incident on the edge \( e = \{v, v_o\} \) and substituting \( v_o \) for every occurrence of \( v \) in left-over faces in \( NF_v \).

(d) For each \( v \in NV_v \), compute its cost using Eq. (3) and update \( VH \).

(e) Repeat steps (a) through (d) until the number of faces in the decimated mesh is \( num_f \).

For identifying fold-overs, we use the usual test that checks whether the normal vector of any face in \( FN \), turns through an angle greater than 90° after collapse.

5. PERFORMANCE ANALYSIS

In this section, we discuss the results and evaluate the performance of VolSIMP by comparing it with similar state-of-the-art algorithms QSlim, MS and FMLOD. The four simplification algorithms (VolSIMP, FMLOD, QSlim, and MS) are scaled using four parameters: running time, memory consumption, quality of the generated LODs, and the preservation of salient features at low levels of detail. Models of varying complexities: mechpart, cow, horse, teeth, raptor and David are used as benchmark models; the statistics of these models are given in Table 1. The experimental environment used for this study consists of a system equipped with Intel Centrino Duo 2.1GHz CPU and 2GB of main memory, and C++ has been used as a programming development tool.
Table 1. Simplification times (in seconds to simplify to 100 faces).

<table>
<thead>
<tr>
<th>Model</th>
<th>M. Size</th>
<th>VolSIMP</th>
<th>FMLOD</th>
<th>QSlim</th>
<th>MS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#Vertices</td>
<td>#Faces</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mechp</td>
<td>1,900</td>
<td>4,998</td>
<td>0.020</td>
<td>0.094</td>
<td>0.046</td>
</tr>
<tr>
<td>Cow</td>
<td>2,903</td>
<td>5,804</td>
<td>0.021</td>
<td>0.093</td>
<td>0.047</td>
</tr>
<tr>
<td>Horse</td>
<td>48,485</td>
<td>96,966</td>
<td>0.523</td>
<td>1.875</td>
<td>1.20</td>
</tr>
<tr>
<td>Teeth</td>
<td>116,604</td>
<td>233,204</td>
<td>1.535</td>
<td>4.813</td>
<td>3.704</td>
</tr>
<tr>
<td>Raptor</td>
<td>1,000,000</td>
<td>2,000,000</td>
<td>14.457</td>
<td>41.860</td>
<td>31.131</td>
</tr>
<tr>
<td>David</td>
<td>4,999,996</td>
<td>7,227,031</td>
<td>66.672</td>
<td>167.953</td>
<td>153.297</td>
</tr>
</tbody>
</table>

Table 1 lists the execution times of the four algorithms. It indicates that VolSIMP outperforms FMLOD, QSLIM and MS in terms of running time; it is, on average, 1.7, 2.28 and 3.21 times faster than MS, QSLIM and FMLOD respectively. In fact, VolSIMP makes one pass through vertices to compute their costs and to insert them into the heap, and then performs maximum $n$ pop operations where $n$ is the total number of vertices, so its time complexity is $O(n \log n)$, whereas the time complexity of each of FMLOD, QSLIM, and MS is $O(e \log e)$, where $e$ is the total number of edges in the mesh, because in each of these algorithms global greedy decision involves choosing an edge.

For objective evaluation of VolSIMP, we employ Symmetric Hausdorff Distance (SHD) that is widely used for thorough comparison of polygonal models in graphics community; it provides tight error bounds and does not discount local deviations. To avoid bias, SHD is calculated using well-known Metro tool [27, 28]. Plots of SHD for 5 LODs of three benchmark models created by the four algorithms are shown in Fig. 3. These plots show that VolSIMP has improvement over FMLOD, QSLIM and MS in terms of SHD. Also, the error maps (Fig. 4) of four LODs, produced by the four methods, further validate our assertion; red areas identify the regions of maximum geometric error.

Fig. 3. Plots of Symmetric Hausdorff distance for (top-left) cow model, (top-right) teeth model, (bottom-left) horse model and (bottom-right) noisy horse model.
VolSIMP preserves visually important information in a better way and keeps the semantic meaning of the surface model even after drastic reduction. For example, Figs. 5 and 8 show that detail features like horns and ears of cow model, and eye, teeth and tongue of raptor model are better preserved by VolSIMP. Fig. 1 reveals that VolSIMP keeps the high resolution detail like eyes of a huge David model (consisting of 7.2 million faces) even after drastic simplification (40,000 faces, 0.05% of original) whereas other algorithms fade out this information. This fact is further validated by correlation coefficients (CC) for David model (see Fig. 9); in case of VolSIMP the value of CC is highest which indicates that in visual space the LOD generated by VolSIMP closely resembles the original model; for computing CC, images of original model, and its approximations are rendered from the same viewpoint and under the same lighting conditions. Fig. 6 shows that VolSIMP competes well with other algorithms in the presence of noise. VolSIMP also simplifies huge models efficiently and faithfully, e.g., see raptor and David models in Figs. 8 and 9. VolSIMP consumes almost as much memory as MS but takes about 44% and 23% less memory than QSlim and FMLOD, respectively, because it needs not to keep error quadrics like QSlim and original normal vectors like FMLOD or any other form of geometric history.
Fig. 6. (a) Original horse model – #Faces: 96966, and its four LODs (consisting of 2000 faces each) created by the four algorithms; (b) Original horse model with random noise (the noise level is 0.125% of the bounding box diagonal) and its three LODs (consisting of 2000 faces each) created by the four algorithms.

Fig. 7. Original teeth model (#Faces: 233,204), and its four LODs (consisting of 2000 faces each) created by the four simplification methods.

Fig. 8. A close-up view of the head of original raptor model (#Faces: 2,000,000), and its LODs (#Faces: 10,000) generated by the four methods.

Fig. 9. Original David model (#Faces: 2,000,000), and its an LODs (#Faces: 40,000) generated by the four simplification algorithms. CC is 0.9662 (VolSIMP), 0.9653 (FMLOD), 0.9553 (QSlim) and 0.9575 (MS).
6. CONCLUSION

Though simplification is well-searched area, still there is space for improvement. The idea of using normal field deviation and volume measure in measuring simplification error has been around for a while, however in this paper these measures have been used in a novel way for developing a simple and reliable automatic simplification algorithm for applications where good time-accuracy trade-off is important and the presence of vertex attributes does not allow the creation of new vertices. Thorough comparison with similar state-of-the-art algorithms such as MS, QSLim and FMLOD, which are known for their good time-accuracy trade-off, demonstrates that it is more efficient than MS, FMLOD, and QSLim in terms of execution time. It has less memory overhead than QSLim and FMLOD. It creates LODs which are better than those produced by other algorithms in terms of symmetric Hausdorff distance and preserves salient shape features in a better way as compared to FMLOD, MS and QSLim.

The proposed algorithm involves two degrees of freedom i.e. to select a vertex and to fix the corresponding optimal half-edge. Selecting a vertex dominates the simplification process. We employ normal field variation as a measure for selecting vertices. Even better measures can be further investigated for better simplification results.

REFERENCES

Muhammad Hussain is an Associate Professor in the Department of Computer Science, King Saud University, Saudi Arabia. He received an M.Sc. and an M.Phil., both from University of the Punjab, Lahore, Pakistan, in 1990 and 1993 respectively. In 2003, he received a Ph.D. in Computer Science from Kyushu University, Fukuoka, Japan. He worked as a researcher at Japan Science and Technology Agency from April 2003 to September 2005. In September 2005, he joined King Saud University as an Assistant Professor. He worked on a number of funded projects in Kyushu University, Japan and King Saud University, Saudi Arabia. His current research interests include multiresolution techniques in computer graphics and image processing.