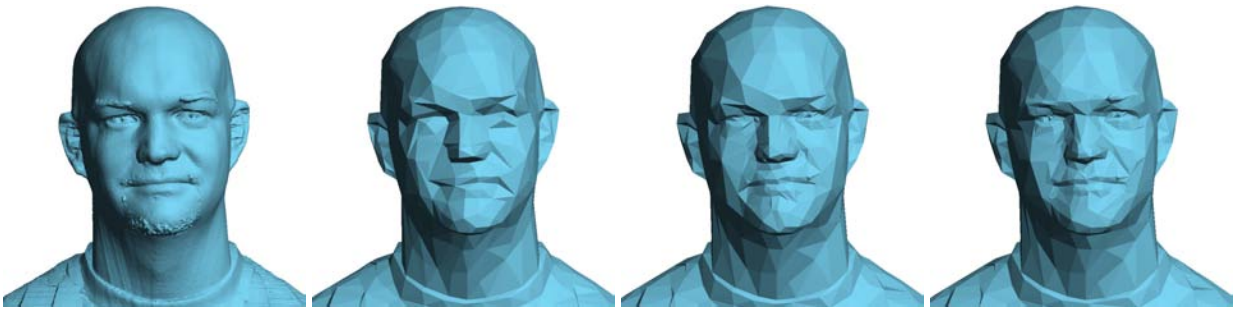
User-Assisted Mesh Simplification

Tan-Chi Ho[1,*] Yi-Chun Lin[1,*] Jung-Hong Chuang[1,*] Chi-Han Peng[1,*] Yu-Jung Cheng[2,†]

[1]National Chiao Tung University [2]Institute for Information Industry



(a) Original mesh (b) Simplified mesh (c) After applying weighting scheme (d) After local refinement

Figure 1: Male model in two-stage user-assisted simplification.

Abstract

During the last decade, many simplification methods have been proposed to generate multi-resolution meshes for real-time applications. Practitioners have found that these methods alone usually fail to produce satisfactory result when models of very low polygon count are desired. This is due to the fact that the existing methods take no semantic or functional metric into account, and moreover, each error metric has its own strength and weakness. In this paper, we propose a user-assisted mesh simplification framework that allows users to improve the quality of simplified meshes derived by any error metric. The framework consists of two stages. The first stage employs a weighting scheme that allows users to refine a unsatisfactory region to achieve a user-specified resolution. The second stage is a local refinement scheme aiming to provide a user-guided fine-tune to recover local sharp features. The proposed weighting scheme differs from the previous approaches in that the weights are used to directly reorder the edge collapsing sequence rather than weighting the collapsing cost. Such a direct reordering mechanism ensures a predictable increase of resolution in the selected region, and is both error-metric and resolution independent.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations;

Keywords: level of detail, mesh simplification, user-assisted simplification

*e-mail: {danki,yglin,jhchuang,chpeng}@csie.nctu.edu.tw

†e-mail: appleman@iii.org.tw

VRCIA 2006, Hong Kong, 14–17 June 2006.
© 2006 ACM 1-59593-324-7/06/0006 $5.00

1 Introduction

Polygonal mesh is one of the most common model presentation in computer graphic applications. With the development of 3D scanning technologies and modeling tools, raw meshes can be composed of thousands to millions of polygons. Real-time rendering of such a large amount of data is always a challenge. Many mesh simplification algorithms have been proposed to decrease the complexity of models while maintaining similarity with the original models.

Among the previously proposed methods, progressive mesh derived by using a series of primitive collapsing, such as edge collapsing, in the increasing order of simplification cost [Hoppe 1996; Garland and Heckbert 1997; Cohen et al. 1998; Lindstrom and Turk 2000] has been proposed. These algorithms usually differ in how the simplification cost is measured. Each of these metrics has its own strength and weakness in preserving geometric and texture features. However, all of these metrics do not take semantic or functional features into account. As a result, practitioners have found that these metrics alone are not able to produce satisfactory result when the simplified mesh of very low-polygon count are expected.

To overcome such limitations, the concept of *user-assisted or user-guided simplification* becomes attractive. One way to this end is to perform refinement or simplification on the simplification hierarchy [Cignoni et al. 1998; Li and Watson 2001; Hussain et al. 2004]. Such setup is usually constrained by the vertex-split dependence problems. Another approach reorders the primitive collapsings by weighting the collapsing cost [Kho and Garland 2003; Pojar and Schmalstieg 2003]. Since the collapsing cost cannot be described by a simple function, the weights applied have no direct relation to the result of refinement. In consequence, the weights are usually chosen in a trial and error basis. Moreover, the weights that are appropriate to a simplified mesh derived by an error metric may not be appropriate to the one derived by another error metric.

In this paper, we propose a user-assisted simplification framework that allows users to improve the quality of simplified meshes derived by any existing error metric, such as QEM [Garland and Heckbert 1997] or APS [Cohen et al. 1998]. The framework consists of two stages. The first stage employs a weighting scheme that

allows users to refine a unsatisfactory region to a user-specified resolution. The second stage is a local refinement scheme based on the vertex hierarchy [Hoppe 1997], aiming to provide a user-guided fine-tune to recovering sharp features. The proposed weighting scheme differs from the previous approaches [Kho and Garland 2003; Pojar and Schmalstieg 2003] in that the weights are used to reorder the edge collapsing sequence rather than weighting the collapsing cost. The reordering mechanism is designed to achieve the following goals:

- The resolution improvement for a given weighting value is predictable.
- The weighting scheme is completely independent of the error metric used, that is, same resolution improvement for a weighting value is expected no matter which error metric is used.
- A weighting value will imply the same resolution improvement when it is applied to meshes in different resolution.

2 Related Work

Level-of-detail (LOD) modeling aims to represent a complex mesh with several levels of detail, and from which an appropriate level is selected at run time to represent the original mesh. A number of methods have been proposed in the literature. Most methods simplify the given mesh by using a sequence of primitive collapsing operations, such as edge collapse [Hoppe et al. 1993], triangle collapse [Hamann 1994], vertex clustering [Rossignac and Borrel 1993], vertex removal [Schroeder et al.], and multi-triangulations [Floriani et al. 1998].

The primitive collapsing operations can be organized in various orders. The simplest way is to perform the operations in arbitrary order. A more sophisticate approach is to perform the operations in the increasing order of collapsing cost, which is analogous to the greedy algorithm. Several error metrics have been proposed to determine the cost of an edge collapsing operation, such as quadric error metrics (QEM) [Garland and Heckbert 1997], appearance-preserving simplification (APS) [Cohen et al. 1998], image-driven simplification (IDS) [Lindstrom and Turk 2000], and perceptually guided simplification of lit, textured meshes [Williams et al. 2003]. Each error metric has its own strength and weakness in preserving certain properties of the original mesh. For example, quadric error metrics [Garland and Heckbert 1997] tends to preserve only the geometric accuracy during the simplification process, appearance-preserving simplification (APS) [Cohen et al. 1998] takes the texture deviation into account, and image-driven simplification [Lindstrom and Turk 2000] aims to preserve the visual fidelity between the simplified mesh and the original mesh. Moreover, These metrics fail to consider semantic or functional features on the models. As a result, it is found in practice that these metrics alone are not able to produce satisfactory results when very low polygon count is the goal.

The first system that allows users to guide the simplification is *Zeta* proposed by Cignoni et al. [Cignoni et al. 1998]. *Zeta* takes a precomputed sequence of primitive simplifications as an input, and utilizes *hyper-triangulation model*, which employs vertex decimation as the local mesh reduction operator. Users can selectively refine a model by locally changing error thresholds to extract different approximations that did not appear during the original simplification process. *Semisimp* proposed by Li and Watson [Li and Watson 2001] provides three approaches for users to manipulate the simplification results using the simplification hierarchy. It allows users to improve mesh quality by manipulating the simplification orders, vertex positions, and the hierarchical partitioning of mesh during the simplification.

Kho and Garland [Kho and Garland 2003] proposed a user-guided mesh simplification system particularly for meshes derived using QEM [Garland and Heckbert 1997]. To increase resolution in a selected region, the system multiply quadric errors associated with vertices in the region by the weighting multiplier, and hence postpone the edge collapse operations in the region. The *constraint quadrics* can be augmented into optimal placement computation to bias the optimal position towards the constrained planes. Pojar et al. presented an approach that is very similar to the work of Kho and Garland [Pojar and Schmalstieg 2003]. A sophisticated Maya plug-in is provided to offer rich interface and great compatibility with other modeling applications. Since the distribution of QEM during simplification can not be described by a simple function, the weighting approach proposed in [Kho and Garland 2003; Pojar and Schmalstieg 2003] suffers from the problem that the value of multiplier has no direct relation to the increase in resolution. In consequence, the value of multiplier is chosen in a trial and error basis.

Hussain et al. [Hussain et al. 2004] proposed a unified framework for constructing multiresolution mesh based on the simplification hierarchy and hypertriangulation model [Cignoni et al. 1998], called *adaptive simplification model (ADSIMP)*. It provides the ability of real time navigation across continuous LODs of meshes. Two operations, *selective refinement* and *selective simplification*, are provided to fine tune the simplified mesh at any level of detail.

3 User-Assisted Mesh Simplification

3.1 Overview

The user-assisted simplification framework we propose will allow users to improve the quality of simplified meshes derived by any existing error metric, such as QEM [Garland and Heckbert 1997] or APS [Cohen et al. 1998]. The framework consists of two stages. The first stage employs a weighting scheme that allows users to refine a unsatisfactory region to a user-specified resolution. The second stage is a local refinement based on the vertex hierarchy [Hoppe 1997], aiming to provide a user-guided fine-tune to recover sharp features. As stated in the Introduction, the weighting scheme proposed here will reorder the collapsing sequence directly, rather than indirectly via modification of the collapsing cost [Kho and Garland 2003; Pojar and Schmalstieg 2003]. Our reordering scheme will delay the edge collapses in the selected region and hence increase the resolution in that region. Such a direct reordering mechanism will ensure a predictable increase of resolution in the region, which is often hard to be achieved by previous methods. Moreover, the proposed scheme is quite unique in its ability to be both error-metric and resolution independent. Same resolution improvement will be obtained for a particular weighting value no matter which error metric is used. Moreover, weighting value will result in the same resolution improvement when it is applied to meshes of different resolutions.

Each stage has its own strength and weakness. The weighting scheme reorders the edge collapsing sequence and may greatly change the simplification result. As a result, the weighting scheme is more effective in overall refinement over a larger region, but is hardly used to fine tune the local features. On the other hand, the local refinement is restricted by the existent vertex hierarchy; but is effective in performing refinement over local areas and recovering

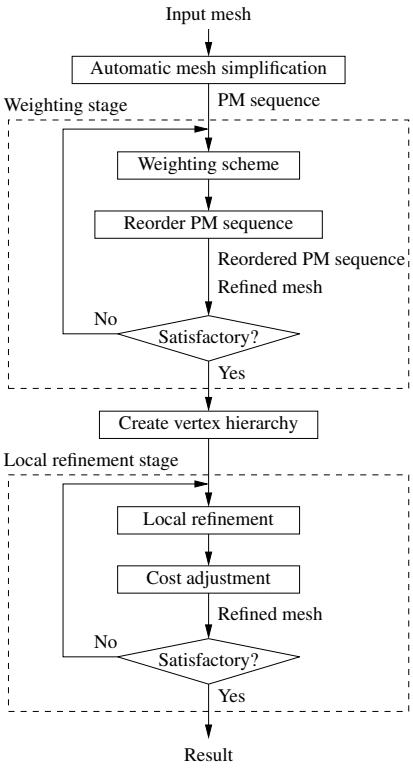


Figure 2: System overview.

sharp features. In the mean time, the local refinement has relatively more control on where to get polygon budget, and hence can be applied to models with low polygon count.

Fig. 2 depicts an overview of the framework. At startup, we construct a progressive mesh (PM) sequence from the input mesh using an automatic mesh simplification algorithm, such as QEM [Garland and Heckbert 1997] or APS [Cohen et al. 1998]. If the user is not satisfied with the quality of the simplified mesh, he or she can use weighting scheme to refine some selected regions. Before the local refinement is performed, the vertex hierarchy is built using the reordered PM sequence. Then user can refines the local features using the local refinement scheme. The refinement process can be repeated until the user is satisfied with the simplified mesh.

3.2 Weighting Scheme

We consider the weighting value as the *multiple of resolution* user expect to have in a selected region. Given a user-specified weighting value, all the edge collapses in that region will be delayed in proportion such that the required resolution improvement in the region can be achieved while maintaining the polygon count. Before getting into the detailed reordering scheme, we first define the *order* of an edge collapse. Consider a complete progressive mesh sequence for simplifying a given original mesh to a vertex, the order of an edge collapse is its order in the PM sequence. For all edge collapses in the selected region, we enumerate them from back to front in the complete PM sequence and in the meantime define the enumeration as the *rank* of the edge collapse. That is, the rank of the last edge collapse in the selected region is 1, the last second is 2, and so on. To make the reordering computation clean, we let the last edge collapse in the complete PM sequence has rank 0.

For a user-selected region R and a user-specified weighting value w, let r_i and o_i be the rank and order of edge collapse i in R, respectively. The new rank $\tilde{r}_i$ of edge collapse i is computed by

$$\tilde{r}_i = \frac{r_i}{w}. \tag{1}$$

The new order $\tilde{o}_i$ of edge collapse i is obtained by the linearly interpolation between o_j and o_{j-1}, where $r_j \geq \tilde{r}_i > r_{j-1}$. That is, for the edge collapse i having new rank $\tilde{r}_i$, we first find o_j and o_{j-1} such that $r_j \geq \tilde{r}_i > r_{j-1}$, and then perform the following linear interpolation:

$$\tilde{o}_i \quad = \quad \left(\tilde{r}_i - r_{j-1}\right) \times o_j + \left(r_j - \tilde{r}_i\right) \times o_{j-1}. \tag{2}$$

Let's illustrate the reordering process using the example shown in Fig. 3, where the triangle dots on the top horizontal line indicate edge collapses in the selected region and their ranks and orders before the weighting value 2 is applied, while the triangle dots on the bottom horizontal line represent reordered edge collapses and their new ranks and orders. The edge collapse with rank 5 is assigned a new rank $2.5\,(=5/2)$, and its new order 675 is the result of a linear interpolation between the orders of edge collapses whose ranks are 3 and 2 before weighting. As a result of the reordering, the first of these six edge collapses is reordered to a place where the fourth edge collapse most likely lies. Since a weighting scheme doesn't take the collapsing cost into account, it is apparent that its effectiveness is independent of the error metric employed.

Since the proposed weighting scheme determines the new order for an edge collapse according to where its new rank lies in the original PM sequence, its effectiveness is applied to whole PM sequence. Hence the effectiveness of a particular weighting value works for simplified meshes of different resolution. Take the example shown in Fig. 4, where weighting value is 3 and M_1, M_2, and M_3 represent the termination points of simplified meshes of three different resolutions. We can see that there are one edge collapse remains before the collapsing terminates for M_1. After applying weighting value 3, the number of edge collapses remain becomes 3. Similar results are observed for M_2 and M_3.

3.3 Local Refinement

The proposed weighting scheme reorders the collapsing sequence for edges inside the selected region, resulting in an increase of the resolution. For certain refinements, such as recovering a sharp corner, it is, however, not as effective as expected. The second stage of our user-assisted simplification framework is a local refinement scheme aiming to provide an effective tool for recovering local features. The proposed refinement operation is similar to the selective refinement and simplification in view-dependent level-of-detail modeling [Hoppe 1997]. The selective refinement (simplification) refines (simplifies) a mesh by moving down (up) the active cut of the vertex hierarchy.

Given a simplified mesh with its progressive mesh sequence, normally the result of the first stage, the system constructs the corresponding vertex hierarchy with collapsing cost recorded on each vertex and the active cut associated with the given simplified mesh. To do the local refinement, user selects a set of vertices and the system will perform vertex split on these vertices, and in the meantime do the vertex collapsing on some vertices to maintain the polygon count. Those vertices that have the lowest collapsing cost are the candidate vertices for edge collapsing. Note that the vertex split or collapsing are just the moving down or up of the active cut.

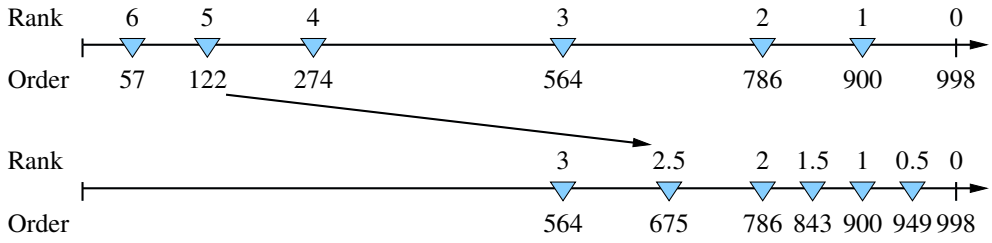


Figure 3: Reordering edge collapses in the selected region after weighting.

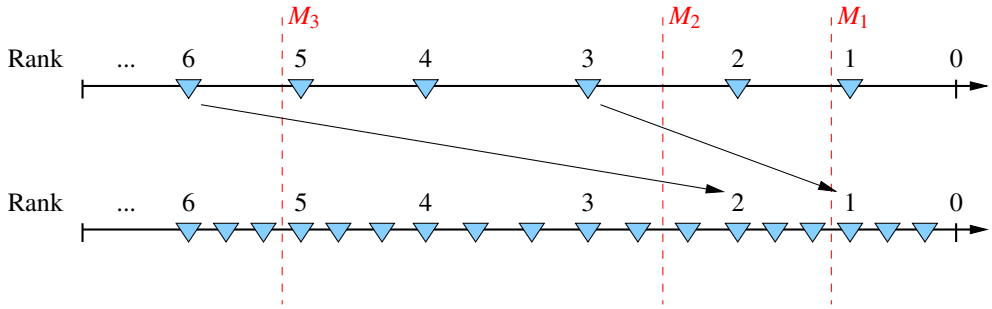
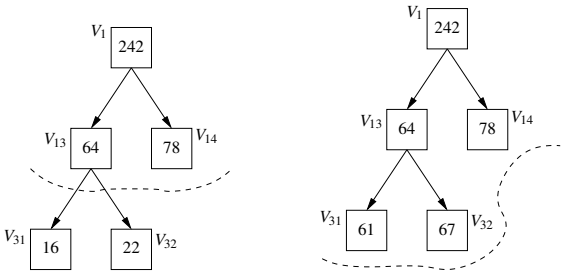


Figure 4: Effect of applying the same weighting value to meshes of different resolution.

One thing worth mentioning is that the vertex split dependency problem may limit the ability of local refinement since a vertex can be split only if all its neighboring vertices after split are reachable. In our implementation, such problems are overcome by applying the approach proposed in [Kim and Lee 2001]. Another thing needs to be addressed is that, after local refinement, vertices resulting from a vertex split normally have costs lower than their parent. After a sequence of vertex splits applied to a vertex v, the subtree originates from v may have leaf vertices whose costs are relatively lower than that of vertices in the active cut. This implies that the split vertices may soon be collapsed when vertices in other region are split. To prevent this problem, we need to adjust the costs of split vertices such that they have about the same magnitude as the cost of v. Further, the cost difference for vertices in the subtree should be maintained to preserve the local features.



(a) Before vertex split (b) After vertex split

Figure 5: Cost adjustment.

The cost adjustment is done along with the vertex split operations in the local refinement process. Let c_v be the cost of a vertex v to be split, and c_1 and c_2 be the costs of children of v. Suppose $c_1 \geq c_2$, c_1 and c_2 are adjusted to c_1^* and c_2^* as follows:

$$\begin{aligned} c_1^* &= c_v + \tfrac{c_1 - c_2}{2} \\ c_2^* &= c_v - \tfrac{c_1 - c_2}{2}. \end{aligned} \tag{3}$$

Note that Eq. 3 ensures that the average cost of the split vertices is the same as their parent and the cost difference between the split vertices is maintained.

As shown in Fig. 5, the costs of V_{31} and V_{32} are adjusted after V_{13} is split, and the average cost of V_{31} and V_{32} are the same as their parent V_{13}. Moreover, the difference between V_{31} and V_{32} remains the same after local refinement.

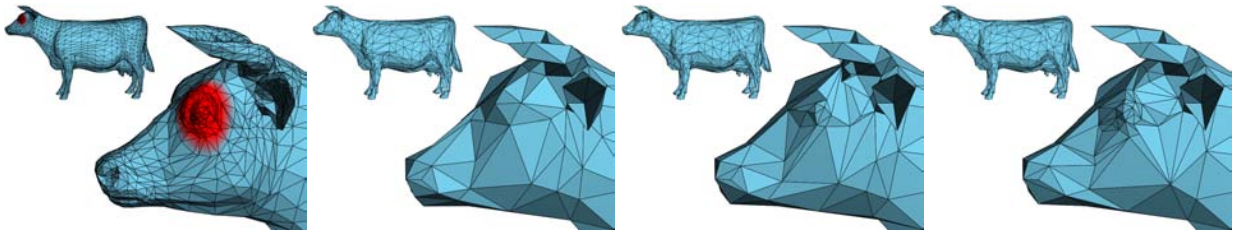
4 Implementation and Results

The proposed user-assisted mesh simplification framework is implemented using C++ and OpenGL. Both QEM (QSlim version) [Garland and Heckbert 1997; Garland] and APS [Cohen et al. 1998] are implemented for measuring collapsing error.

Several experimental tests are performed to demonstrate the effectiveness of the proposed weighting scheme. First example is a cow model of 5804 polygons, which is simplified to a mesh of 1160 polygons by using QSlim. Different weighting values are applied to the region of left eye as shown in Fig. 6(a). Fig. 6(c)-(d) depict the refined meshes after applying weighting values 2 and 3, respectively.

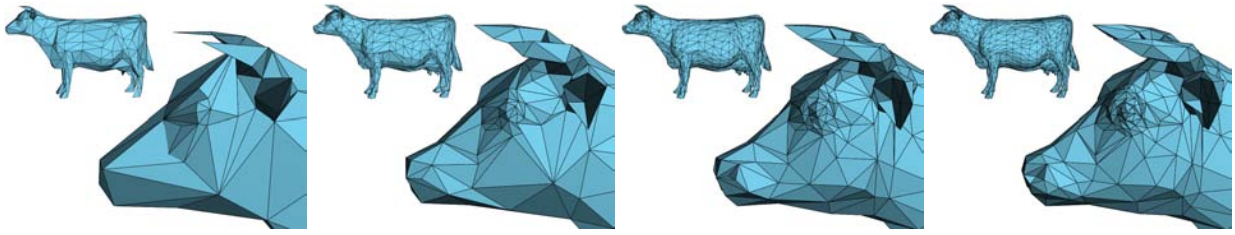
Second test is to apply weighting scheme to the region of left eye on simplified cow models of different resolutions, namely 500, 1160, 1739, 2321, and 2902 polygons. Fig. 7 shows the resultant meshes after applying weighting value 3. As shown in Table 1, the resolution improvement for meshes of different resolutions are quite close to the amount expected. Note that the small inaccuracy in resolution improvement is due to the dependency problem happens at the boundary of the selected region.

Third test focuses on the effectiveness of two-stage user-assisted simplification framework. A dragon model of 50000 polygons is simplified to a mesh of 1500 polygons, as shown in Fig. 8(a) and (b). The weighting value of 3 is applied to the regions of eyes, with the resultant mesh shown in Fig. 8(c). Local refinement is then applied to areas of teeth and nose, producing refined mesh



(a) Original mesh (b) Simplified mesh (c) Weighting value = 2 (d) Weighting value = 3

Figure 6: Refined cow model with different weighting values.



(a) 500 polygons (b) 1160 (20%) polygons (c) 1739 (30%) polygons (d) 2321 (40%) polygons

Figure 7: Applying weighting value 3 to the simplified cow model of different resolution.

Polygon count	Vertex count in the selected region		
	W/O weighting	Weighting value = 2	Weighting value = 3
500	6	13	20
1160 (20%)	11	24	37
1739 (30%)	15	29	47
2321 (40%)	19	40	60
2902 (50%)	26	52	76

Table 1: Resolution improvement after applying weighting scheme.

	Simplified mesh	Simplified mesh after refinement
Minimum	2.758e−8	3.187e−8
Maximum	5.045e−3	5.008e−3
Mean	5.131e−4	5.460e−4
Variance	1.880e−7	1.963e−7

Table 2: Geometry deviation of the simplified meshes.

	Simplified mesh	Simplified mesh after refinement
Minimum	7.984e−4	8.007e−4
Maximum	1.925	1.948
Mean	0.222	0.227
Variance	0.04469	0.04461

Table 3: Normal deviation of the simplified meshes.

shown in Fig. 8(d). Another example is a male model of 151K polygons, which is simplified to a mesh of 1500 polygons using QSlim. Weighting scheme is applied to regions of eyes, lips, and nose, and local refinement is applied to recover sharp features such as eyeballs, eyebrows, and nose. The result is shown in Fig. 1.

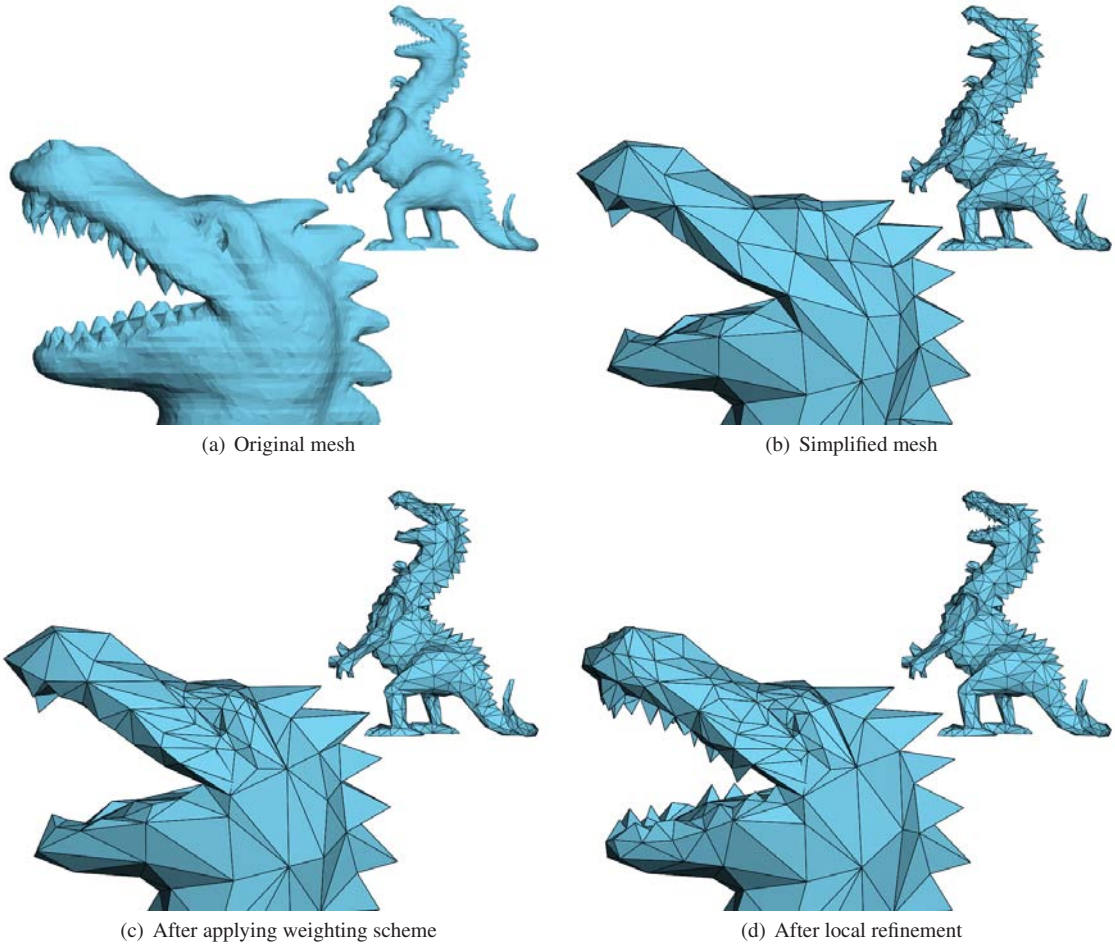
Fourth test tries to improve the simplified Parasaur model with texture mapped. The parasaur model of 7685 polygons is simplified to a mesh of 750 polygons using APS. Texture distortion resulting from the simplification is apparent, as shown in Fig. 9(b). Fig. 9(c) and (d) depict the great reduction of texture distortion after applying weighting scheme and local refinement, respectively.

Table 2 and Table 3 list the geometry and normal deviations measured using MeshDev, a mesh comparison tool using attribute deviation metric [Roy et al. 2004]. Although the mean errors after applying user-assisted simplification are slightly increased, the errors are diffused over the regions where are considered perceptually unimportant. Fig. 10 visualizes the distributions of geometry and normal deviations. Noticable improvements can be found in the selected regions, and the errors introduced by the proposed scheme are almost invisible and diffused over the other regions.

5 Conclusion and Future Work

We have proposed a two-stage user-assisted mesh simplification framework that allows users to improve the quality of simplified meshes derived by any error metric. The first stage is a weighting scheme that allows users to refine a selected region to a user-specified resolution. The second stage is a local refinement scheme aiming to provide a user-guided fine-tune to recover local sharp features. The proposed weighting scheme differs from the previous approaches in that the weights are used to directly reorder the edge collapsing sequence rather than weighting the collapsing cost. Such a direct reordering mechanism ensures a predictable increase in resolution of the selected region, and is both error-metric and resolution independent. The effectiveness of applying a weighting value is the same for meshes derived by any error metric, and for meshes in different resolutions.

Current weighting scheme deals with one selected region at a time. The capability of handling user-assisted refinement simultaneously in several selected regions can greatly improve the computation efficiency of the simplification system. We will extend current implementation to this multiple weighting scheme. The optimal position for edge collapsing can be obtained by using QEM. In the proposed framework, the position of vertices on the boundary of the selected region may not be well preserved after edge collapsed since the



(a) Original mesh (b) Simplified mesh

(c) After applying weighting scheme (d) After local refinement

Figure 8: Dragon model in two-stage user-assisted simplification.

weighting scheme doesn't modify the value of QEM. An optimal placement derivation that takes the weighting value into account is a possible way to improve the quality at the boundary of the selected region.

6 Acknowledgements

The authors would like to thank Pedro V. Sander for providing the model and texture map of parasaur head used in this paper. This project was partially supported by Institute of Information Industry under III 93-0259.

References

CIGNONI, P., MONTANI, C., ROCCHINI, C., AND SCOPIGNO, R. 1998. Zeta: A resolution modeling system. *Graphical Models and Image Processing 60*, 5 (Sept.), 305–329.

COHEN, J., OLANO, M., AND MANOCHA, D. 1998. Appearance-preserving simplification. In *Proceedings of ACM SIGGRAPH 98*, 115–122.

FLORIANI, L. D., MAGILLO, P., AND PUPPO, E. 1998. Efficient implementation of multi-triangulations. In *Proceedings of IEEE Visualization 98*, 43–50.

GARLAND, M. Qslim simplification software. http://graphics.cs.uiuc.edu/ garland/software/qslim.html.

GARLAND, M., AND HECKBERT, P. S. 1997. Surface simplification using quadric error metrics. In *Proceedings of ACM SIGGRAPH 97*, 209–216.

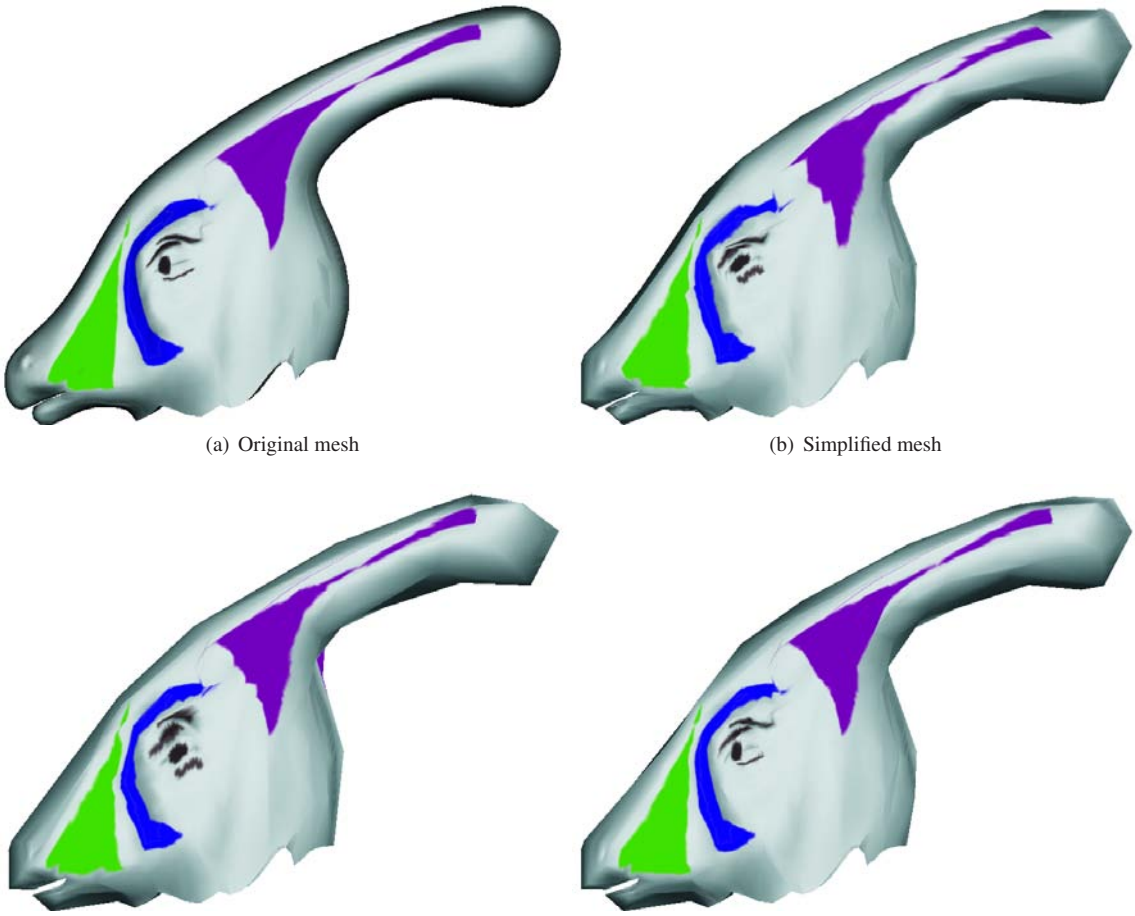
HAMANN, B. 1994. A data reduction scheme for triangulated surfaces. *Computer Aided Geometric Design 11*, 2, 197–214.

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. 1993. Mesh optimization. In *Proceedings of ACM SIGGRAPH 93*, 19–26.

HOPPE, H. 1996. Progressive meshes. In *Proceedings of ACM SIGGRAPH 96*, 99–108.

HOPPE, H. 1997. View-dependent refinement of progressive meshes. In *Proceedings of ACM SIGGRAPH 97*, 189–198.

HUSSAIN, M., OKADA, Y., AND NIIJIMA, K. 2004. User-controlled simplification of polygonal models. In *International Symposium on 3D Data Processing, Visualization and Transmission*, 918–925.



(a) Original mesh (b) Simplified mesh

(c) After applying weighting scheme (d) After local refinement

Figure 9: Parasaur model in two-stage user-assisted simplification.

KHO, Y., AND GARLAND, M. 2003. User-guided simplification. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics*, 123–126.

KIM, J., AND LEE, S. 2001. Truly selective refinement of progressive meshes. In *Graphics interface 2001*, 101–110.

LI, G., AND WATSON, B. 2001. Semiautomatic simplification. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics*, 43–48.

LINDSTROM, P., AND TURK, G. 2000. Image-driven simplification. *ACM Transactions on Graphics 19*, 3 (July), 204–241.

POJAR, E., AND SCHMALSTIEG, D. 2003. User-controlled creation of multiresolution meshes. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics*, 127–130.

ROSSIGNAC, J., AND BORREL, P. 1993. Multi-resolution 3d approximations for rendering complex scenes. 455–465.

ROY, M., FOUFOU, S., AND TRUCHETET, F. 2004. Mesh comparison using attribute deviation metric. *International Journal of Image and Graphics 4*, 1 (january), 127–140.

SCHROEDER, W. J., ZARGE, J. A., AND LORENSEN, W. E.

WILLIAMS, N., LUEBKE, D., COHEN, J. D., KELLEY, M., AND SCHUBERT, B. 2003. Perceptually guided simplification of lit, textured meshes. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics*, 113–121.

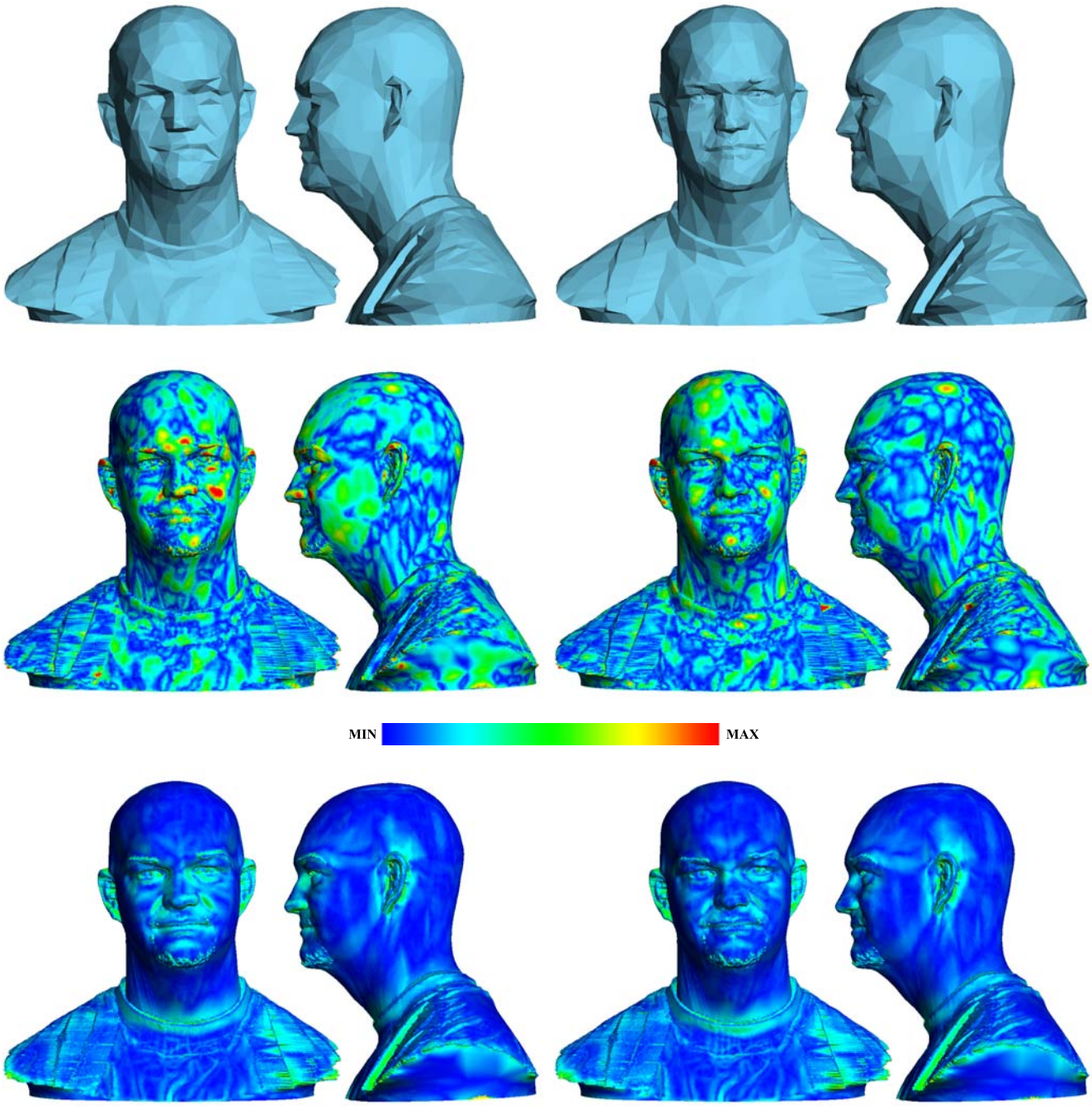


Figure 10: Visualization of the error distributions for simplified male model before (left) and after apply user-assisted simplification (right). The top row are the shaded meshes, the middle row shows the distributions of geometry deviation, and the bottom row visualizes the normal deviation. Both deviations are measured using MeshDev [Roy et al. 2004].