Constructing Scalable 3D Animated Model by Deformation Sensitive Simplification

Sheng-Yao Cho
National Taiwan University
ven@cmlab.csie.ntu.edu.tw

Bing-Yu Chen
National Taiwan University
robin@ntu.edu.tw

ABSTRACT
To date, more high resolution animated models are required to present important details and fine structures, however, sometimes such high resolution models are unnecessary and undesired. For example, we usually want to preview a low resolution animated model to decide if we want to download it or not. Interactive systems, for another example, sometimes use low resolution models to obtain better performance. Though there are many well-known algorithms dealing well on simplifying 3D models, most of them are limited to static ones. Applying these mesh simplification methods to 3D animated models, a good simplified model in a specified pose can be obtained. However, some features of the original animated model, which can be shown in other poses, may be destroyed. In this paper, we propose an automatic method to simplify a 3D animated model which takes the features shown in every poses into account and preserves the geometry details of it.

Categories and Subject Descriptors
I.3.5 **Computer Graphics**]: Computational Geometry and Object Modeling; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism - *Animation*.

General Terms
Algorithms, Experimentation.

Keywords
Progressive Meshes, Level-Of-Detail, Mesh Simplification, Animated Model, Computer Animation.

1. INTRODUCTION
Currently, animated models are often created at high resolution to present important details and better visual effects, however, sometimes such high resolution models are unnecessary and undesired. For example, when we want to preview an animated model to decide if we want to download it or not, we may not need to download the finest one to preview. Interactive systems, for another example, sometimes use low resolution models to obtain better performance. Hence, many algorithms were proposed to simplify meshes automatically

There is another example which can use simplified models. *Conceptual Farm* [9], a behavior authoring system for artificial characters, simulates characters' behaviors and displays them realistically with virtual reality techniques. Computing time for determining behaviors of characters dependents on the amount of characters. While the amount is large, it is hard to keep the rendering performance in real-time. Using animated models with level-of-details (LOD) seems a solution to this problem which will mush improve the performance while there are enormous artificial characters in the scene.

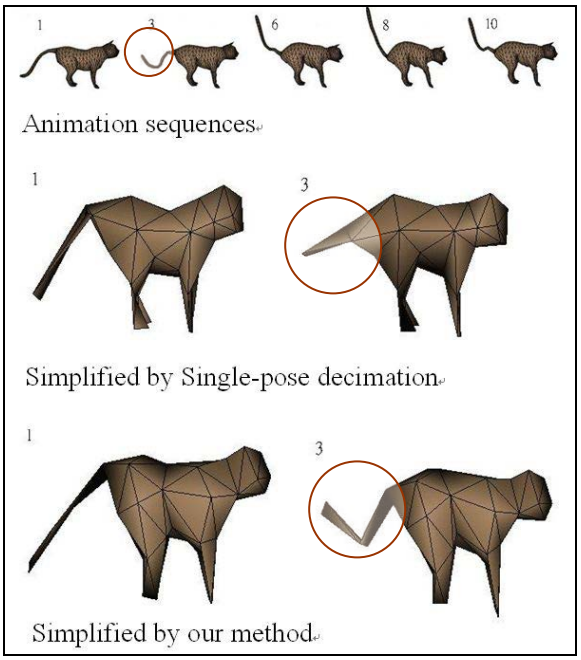

Figure 1. Upper row: the original sequence of a 3D animated model. Middle row: the simplified models of the first and third frames when simplifying the model shown in the first frame. Lower row: the simplified models of the first and third frames when using our method. Obviously, the tail of the cat model in the third frame is preserved in our method.

In general, simplification methods are based on error-approximation of a 3D model in a single pose. This makes good simplified models in a neutral pose; however, it may destroy some features of the models in some particular poses. As shown in Figure 1, the upper row shows some key-frames of a cat animation sequence while the numbers on the picture indicates the index. The middle row is the result obtained by using Quadric Error Metric (QEM) simplification method [7] which only references the first pose of the model. Both of the simplified models have 100 faces and 52 vertices. Take a look at the tail part of the third-frame model; it did not present the curved effects of the object. While the lower row shows the result obtained by our method, the tail part gets good approximation and the vertices of the model remains consistent.

2. RELATED WORK

2.1 Mesh Simplification and LOD

The concept of LOD [5] was proposed by Clark. Over the years many researches and frameworks for LOD have been developed, including Progressive Meshes (PM) for continuous LOD (CLOD) [11], vertex hierarchies for view-dependent LOD [12][16][22]. Although the essential concepts of LOD management can apply to any model representation, the most common and important application remains the simplification of polygonal meshes. In recant years, great achievements have been made on surface simplification and multi-resolution modeling. The following lists the related mesh simplification algorithms which can be applied into to our system.

Edge collapse operator for mesh simplification [14] was first proposed by Hoppe et al. This operator collapses an edge into a single vertex. An error function which approximates the geometry error caused by the edge collapse operation is used to decide the edge collapse sequences. It is the most used method to simplified mesh and many algorithms or concepts are extended from this, such as view-independent simplification [11], view-dependent simplification [12], progressive compression [2], as well as progressive transmission [2][10].

Quite similar to edge collapse operations, vertex-pair collapse operator collapses two vertices into a single vertex without the limitation that there must be an edge between those vertices. We can say that there is a virtual edge between them and so-called as virtual-edge collapse operation [6][7][17][18]. For two unconnected vertices, it has to define a threshold to limit the number of virtual edges to be small.

Vertex cluster operator uses a bounding box to divide a mesh into grids, and a single vertex is chosen among vertices within each grid. Rossignac and Borrel [23] proposed a method capable of processing arbitrary polygonal meshes. However, the size of the grid dominates the approximation error. The quality of meshes from this method is usually quite low.

Vertex removal is first proposed by Schroeder et al. [19]. It removes a vertex and then re-triangulates the hole. There are several ways to choose the vertex to be removed and to re-triangulate the holes [21].

2.2 Time-Dependent Meshes

Alexa and Müller [1] proposed an approach to represent time-varying geometry by principal components. Through their method, a matrix composed of consistent meshes of original key-frames is build at first. The vertices of each frame of this matrix are then decomposed to bases and weights after a Singular Value Decomposition (SVD) operation. By adapting different number of the basis, LOD can be achieved. It also supports progressive animation compression with spatial, as well as temporal. However, the computational time for SVD decomposition is expensive and the view-dependent property can not be fulfilled with this approach.

Houle and Poulin presented a very similar algorithm of ours on skeletal meshes [15]. They combined the PM concept and skeletal models to produce animation with CLOD. Their contribution is major for game industry, especially on-line games; however, it is limited to the models with skeleton, e.g. human or animals. Furthermore, they simplify the articulated mesh by taking only one single static model into account without considering the poses of animation sequences, so they had to adjust the model to a special pose by hand for better simplification effect.

Shamir and Pascucci proposed a scheme for creating LOD models for time-dependent meshes [20]. It supports both temporal and spatial LOD. They divided temporal variation into factors with low and high frequency. In their definition, low-frequency factor stands for global affine transformation while high-frequency factor stands for local vertex deformations. They had different and good LOD effects by applying different updates (e.g. spatial and temporal); however, the size of the encoded data is much larger than the original mesh.

Based on "Geometry Image" [8], Briceno et al. proposed "Geometry Video" [3] as a new representation for 3D animation. For each key-frame model, it uses global cut algorithm and parameterization method to create consistent geometry images. It inherits many advantages from Geometry Images and provides meshes in each frame with regular connectivity, LOD, and allows for applying numerous processing and compression methods targeted at videos to animated models. Though it has several advantages over previous techniques; however, for models with high-genus, it will cause high distortion due to the defeats of geometry image.

3. Our Method

In this section, we will introduce the major algorithm that constructs animated models whose deformed features are well-preserved. We propose a simplification scheme reordering sequences of the edges to be collapsed. The fundamental simplification method is based on edge-collapse operation but is not limited to any edge-collapse method. Currently, we use QEM as our metric of simplification.

3.1 Background: Review of QEM

Currently, we adopt QEM which was proposed by Garland and Heckbert [7] as our simplification metric because it is widely used and it gives good tradeoffs between the quality and simplification speed. The major contribution of it is to decide the order of pair contraction and calculate the optimal position of the new vertex after the pair-contract operation.

We can separate this algorithm into two main parts. The first one is the initialization part while the other is decimation part. The set of candidate pairs that may be contracted throughout simplification is first determined in the initialization part. This set typically includes all the edges and some non-edge pairs whose distance from end to end are short than a threshold. The collapse cost is also computed for each candidate pair at the same time. Each candidate pair is placed in a priority queue according to its cost. During the simplification stage, the algorithm selects the candidate pair with lowest cost and removes it from the priority queue. This pair is then collapsed, and the mesh data structures are updated to remove any degenerate geometry. All affected pairs' costs in the local neighborhood are reevaluated, and this simplification stage is iterated until the desired approximation is reached.

The quadric $Q^f(\mathbf{v})$ is defined first to represent the squared distance of a point $\mathbf{v}$ to the plane containing the face f. For each vertex v of the mesh, it defines $Q^v(\mathbf{v})$ from summing the quadric $Q^f(\mathbf{v})$ weighted by the face area where f is the adjacent face of

v. After doing a pair construction operation $(v_1, v_2) \to v$, the new position of vertex v is got by minimizing

$$Q^v(\mathbf{v}) = Q^{v_1}(\mathbf{v}) + Q^{v_2}(\mathbf{v}) .$$

The squared distance between a vertex v and a plain P with normal $\mathbf{n}$ is give by: $Q^f(\mathbf{v}) = D^2 = (\mathbf{n}^T\mathbf{v} + d)^2 = \mathbf{v}^T(\mathbf{n}^T\mathbf{n})\mathbf{v} + 2d\mathbf{n}^T\mathbf{v} + d^2$, which can be represented as $Q^f(\mathbf{v}) = \mathbf{v}^T(\mathbf{A})\mathbf{v} + 2\mathbf{b}^T\mathbf{v} + \mathbf{c}$, where $\mathbf{A}$ is a symmetric 3x3 matrix, $\mathbf{b}$ is a vector of size 3, and $\mathbf{b}$ is a scalar. $(\mathbf{A}, \mathbf{b}, \mathbf{c})$ is stored using 10 coefficients. The position of $\mathbf{v}$ which makes $Q^v(\mathbf{v})$ minimized can be found by solving the linear system: $\nabla Q^v(\mathbf{v}) = 2\mathbf{A}\mathbf{v} + 2\mathbf{b} = 0$.

3.2 Simplification Scheme

As mentioned above, QEM is used to approximate the cost of contraction which is caused by pair collapse operation. Following the measurements, we determine the order of the pair collapses. The cost of contraction is simply computed as $\mathbf{v}^T(Q_1 + Q_2)\,\mathbf{v}$, where $\mathbf{v}$ is the new vertex position and Q_1 and Q_2 are error quadrics of the pair vertices to be collapsed. Although this metric works well for a static model, it may not work for an animated model. If we only apply the QEM contraction cost measured by using a model of arbitrary single frame for simplification, the model shown in other frames may not be shown in its good approximation. Therefore, instead of considering only a single model, we would like the cost measure of pair contraction to take the deformations of the model over all poses into account.

A common sense of the solution is to take the average cost approximation of vertex pair (v_1, v_2) of every frame as the QEM values of that pair, which is shown as the following formula:

$$Q_{(v_1,v_2)} = \frac{1}{n}\sum_{i=0}^{n} \mathbf{v}_i^T(Q_{v_1}^i + Q_{v_2}^i)\mathbf{v}_i ,$$

where i indicates the index number of frames and v_1 and v_2 are two endpoints of the pair to be collapsed. Simplifying models by this formula will indeed get better simplified models while considering the error caused by decimation. It is because that, mathematically speaking, what averaging method considers is the major motion tendency of the animation, but it would not detect the features of motion among the whole animation sequence.

Take a look at the following chart (Figure 2), for example, it shows the decimation cost as a function of time (frame number). Average method concerns about the area covering by the curve. In this chart, *x*-axis means the frame number and *y*-axis means the QEM value after we decimate this vertex pair. And, the blue curve stands for one vertex pair and the red one stands for another. The areas under both curves are almost equivalent, so we would not be conscious of the feature part of the vertex pair 1. It is because that the sudden peak will be smoothed by the average method.

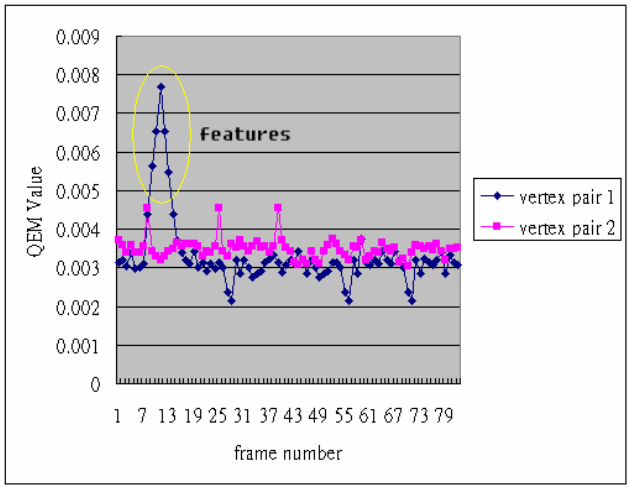

Figure 2. It would not detect the peak of the curve by the average scheme.

Thus, to solve this problem, the simplification scheme we proposed is defined as the following formula:

$$Q_{(v_1,v_2)} = \max_i \mathbf{v}_i^T(Q_{v_1}^i + Q_{v_2}^i)\mathbf{v}_i .$$

It means that every time we would like to do a pair-collapse operation, we will choose the pair with the minimal QEM value to decimate, but the value of it is determined by the identical pair with maximal value of the animated model among all of the poses (shown as Figure 3).

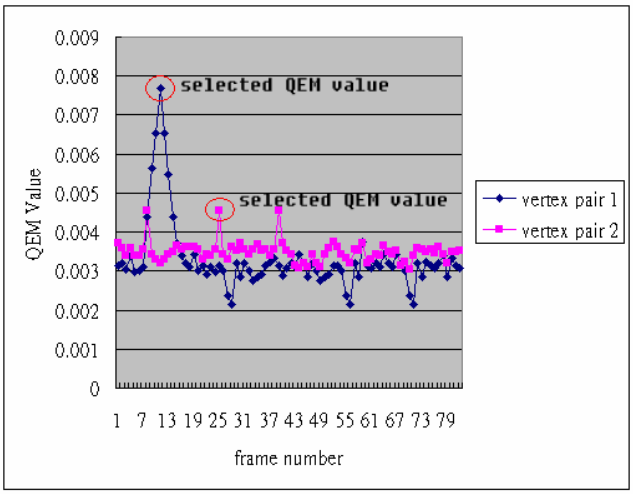

Figure 3. We preserve the feature of the motion by take the maximal QEM value as our decimation criterion.

The reasons that we use this simplification scheme are based on the following heuristics:

1. The decimation cost of the original features of the model are certainly high within all frames.
2. Features caused by the deformation will introduce peaks of vertex pairs in the chart of frame-number and QEM-Value.

From the first heuristic, we can still preserve the original features of the animated character because those features are motion independent, and the second one takes the features caused by some specific motions into account whether those motions are frequent or not. If the motion is frequent in this animation sequence, it will introduce many peaks in the chart of QEM value. There is no doubt that both average scheme and our scheme can detect it, however, if this motion is not frequent in those animation sequence, e.g. it only happens once or twice, the average scheme will neglect it, because the heavy weight of the pair in this motion will be averaged to those of all other frames.

3.3 Representation Scheme

We describe an animation comprised of a sequence of animated models which are denoted as $\widehat{M}_i = \widehat{M}_0 + \triangle M_i$ for the i-th key-frame, where $\widehat{M}_0$ is a triangle mesh; each animated model is deformed from it, and it can also be the model at the first key-frame, i.e., $\widehat{M}_1 = \widehat{M}_0$. $\triangle M_i$ stands for the geometric offset of the i-th key-frame between $\widehat{M}_i$ and $\widehat{M}_0$. Then, the state of the object can be calculated by interpolating between two animated models on two consecutive key-frames. Therefore, if the number of polygons of the object is large, to generate the model sequence is a time-consuming task.

To decrease the number of polygons of the object is a good idea to solve the problem. However, if we simplify the animated models separately, the interpolation of two simplified animated models may be an artifact. To make the simplified animated models consistent as the original animated models, we only simplify the initial mesh $\widehat{M}_0 = M_0^n$ into a coarser mesh by applying a sequence of n successive edge collapse operations. Since edge collapse operations are invertible, $\widehat{M}_0$ can therefore be denoted as its simplified mesh M_0^0 with a sequence of n vsplit records as described in [11], where vsplit is a vertex split operation which is the inverse operation of edge collapse. Hence, $\widehat{M}_0 = M_0^n$ can be denoted as

$$\widehat{M}_0 = M_0^n = M_0^0 + \sum_{j=1}^{n} vsplit_j .$$

Then, the animated model of the i-th key-frame can be defined as

$$\widehat{M}_i = M_i^n = M_0^0 + \sum_{j=1}^{n} vsplit_j + \triangle M_i .$$

Therefore, by applying some vsplit records to M_i^0, we can get multi-resolution animated models for each key-frame.

4. RESULTS

In order to compare to results of other simplification schemes, we have implemented three different schemes (including ours) which are all illustrated in Table 1. In Section 4.1, we will show some visualized results of the animated models simplified by different schemes. Mathematic analysis of error approximations will be revealed in Section 4.2. We use Metro [4] as a tool to measure the root mean square (RMS) distances from the surface of the simplified meshes to the original ones as error approximation metric. Three animations were taken to be experimented. Table 2 will introduce those animations. In Section 4.3, we will make a comparison of the LOD of our representation with it of other 3D model representation.

4.1 Visualized Results (Animation 1)

In this animation, the dog will first sit down, scratch itself, and swing its tail. After a stretch, it will lower its head to drink water. Features of this animated model in this animation lie on the four limbs and the tail.

In Figure 4, there shows the original model (8,135faces) and simplified models (201 faces) of this animation. Each simplified models are generated by using different simplification scheme. From Figure 4, it is easy to find out that while the average scheme and minimal scheme over simplify the details of four limbs and the tail of the animated model in frame 0, our method preserve them well. The quality of the third model generated by single frame scheme is also good, because the reference model is the same as the simplified one. Though the effect looks good in this frame, it is not as good as in the whole animation sequences.

In Figure 5, it shows some continuous key-frames of this animation. Notice the tail and limbs of the dog. Our method preserves both of the parts well than others. The tail part of the model generated by single frame and minimal scheme, and the back limbs of the model generated by average method are distorted.

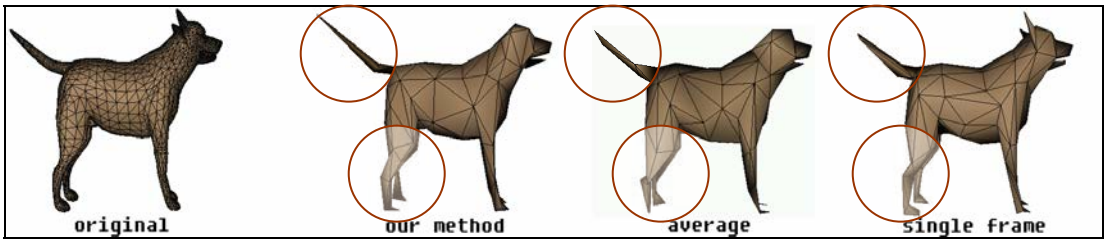

Figure 4. Original model and simplified models. (Frame 0)

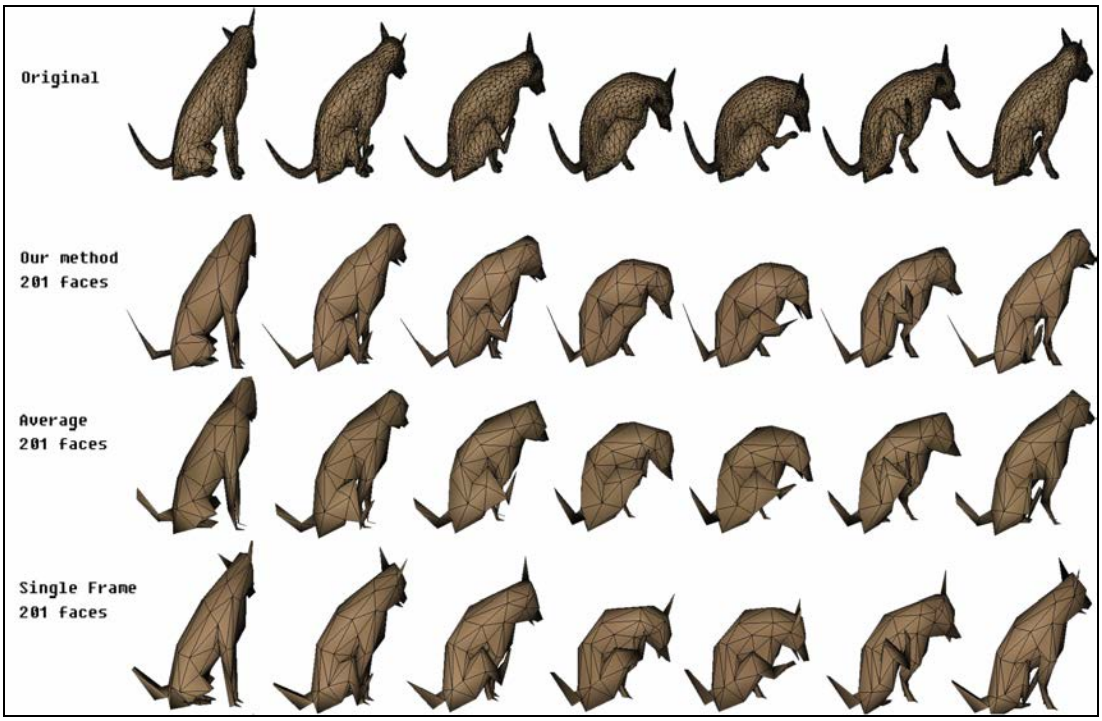

Figure 5. This is a dog-scratching animation (7 key-frames of 111 frames)

	Scheme Descriptions	Representative QEM for a pair (v_1, v_2)
1	Select the maximal QEM of a pair (v_1, v_2) among all models as the QEM of this pair.	$Q_{(v_1,v_2)} = \max_i \mathbf{v}_i^{\mathrm{T}} (Q_{v_1}^i + Q_{v_2}^i)\mathbf{v}_i$
2	Select QEM of a pair (v_1, v_2) from an arbitrary model as the QEM of this pair	$Q_{(v_1,v_2)} = \mathbf{v}_0^{\mathrm{T}} (Q_{v_1}^0 + Q_{v_2}^0)\mathbf{v}_0$
3	Average QEM of a pair (v_1, v_2) of all models as the QEM of this pair	$Q_{(v_1,v_2)} = \frac{1}{n}\sum_{i=0}^{n} \mathbf{v}_i^{\mathrm{T}} (Q_{v_1}^i + Q_{v_2}^i)\mathbf{v}_i$

Table 1. Selected QEM Value of a pair (v_1, v_2) in different simplification schemes

	Description	#Frame	#Vertex	#Face
1	Happy Dog	111	4070	8135
2	Angry Cat	31	2700	5400
3	Walking Dog	20	4070	8135

Table 2. Description of experimental animations

4.2 Statistic Results (Animation 1)

As mentioned above, for each animation, we use four simplification schemes respectively to make it 10 levels of details. In this section, we will illustrate the statistic results.

We use Metro [4] as a measuring tool to compare the simplified models to original ones. It is designed to compensate for a deficiency in many simplification methods proposed in literature. It allows one to compare the difference between a pair of surfaces (e.g. a triangulated mesh and its simplified representation) by adopting a surface sampling approach. It has been designed as a highly general tool, and it does no assumption on the particular approach used to build the simplified representation. And It can return both numerical results (meshes areas and volumes, maximum and mean error, etc.) and visual results, by coloring the input surface according to the approximation error.

For each simplified model (in different level and frame), we measure a mean distance and a root mean square distance from its surface to the correspond model (in the same frame but full resolution). Then Matlab[1] is used to make charts for observation: *x*-axis stands for the level (from 10 to 1) of the simplified model, where level-10 is the full resolution and level-1 contains 10% faces of the original model; *y*-axis stands for the frame number, and *z*-axis stands for the measured error approximations.

Figure 6 shows the RMS distance approximation of all simplified model generated by four different simplification schemes. RMS distance is a better measurement or metric than mean distance,

[1] http://www.mathworks.com/

because it is two-norm distances and it considers the space-relationship of distances between sampled points on original model and simplified model. From Figure 7, we find out that the RMS between original model and simplified model are smaller if we use our method or average scheme. Another observation is that the surface of the chart created by our method is much smoother than those by other simplification scheme.

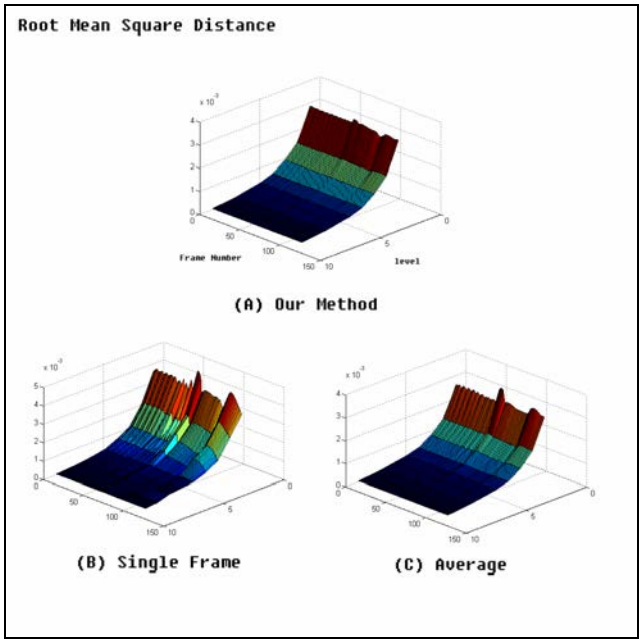

Figure 6. RMS distance between simplified animated model and original animated model.

Figure 7 shows the difference of RMS distance between our method and others. It is easy to find out that our method produces better simplified animated models in every segment than single frame and minimal schemes. Average scheme gets better error rate than ours due to its mathematical model, but our method has better visually results which have been shown in Section 4.1. Notice that between frames 53 to 62, results of our method get better approximations than others. It is because that among all the animation sequences, the body of the dog keeps straight, but only in those frames, the body bends. This is only a current assumption, and the more details about the analysis are left on the future works.

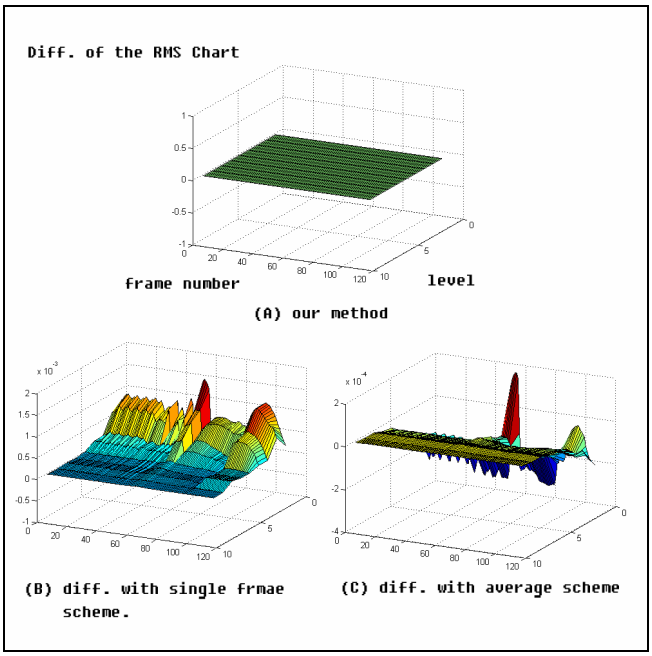

Figure 7. Difference of Root Mean Square distance between other schemes and ours.

4.3 Other Results (Animation 2 and 3)

In this section, we will only show the result models generated by our method. One is angry cat animation and the other is walking dog animation. Notice the tail and limbs parts of the characters. We preserve those features well. The cat model is simplified from 5, 400 faces to 100 faces while the dog model is from 8, 135 faces to 150 faces too.

Figure 8. An angry cat animation. The lower row shows 5 of the original models, each model consists of 5,400 triangles. The middle and upper rows show the corresponding simplified models, each one consists of 1,000 and 100 triangles, respectively.

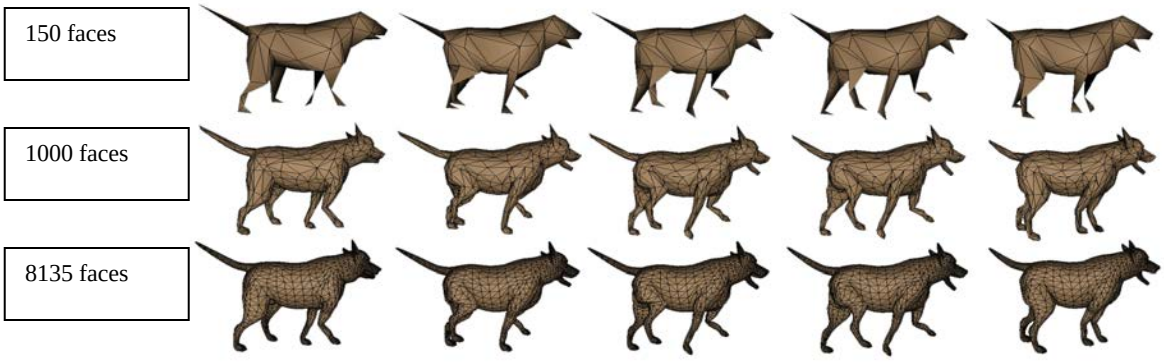

Figure 9. A walking dog animation. The lower row shows 5 of the original models, each model consists of 8,136 triangles. The middle and upper rows show the corresponding simplified models, each one consists of 1,000 and 150 triangles, respectively.

5. CONCLUSION AND FUTURE WORK

In this paper, a fully automatic simplification scheme, which preserves the deformation features of objects, for animated models is proposed. Three dimensional animated models, either key-framing or skeletal models, can be inputs of our system, and a scalable animated model is generated.

The scheme we proposed takes every pose of the animated model into account, and determines a better decimation sequence to simplify it. We can find out that our method really produce better simplified models than other schemes from the visualized results. Moreover, we can conclude that our method can generate a smooth animated model from the statistic results, though the evaluation of error approximation of our method is slightly higher than the average scheme.

Currently our simplification does not consider the attributes of the vertices, so the first work is to make the system complete. The attributes of vertex include color, textures, and even the binding weights of the skeletal models..

Though the animated model is scalable, the storage space is huge. We would like to propose a well-organized data structure to store those data. The TDAG [20] which Shamir and Pascucci proposed may be a good reference for us to start.

Finally, this technique can be used widely in several kinds of application area, e.g. Massive Multiplayer Online Role-Playing Game, systems which need real-time performance, *Conceptual Farm* [9]. So we would like to apply it into more kinds of applications.

6. REFERENCES

This work was partially supported by the National Science Council of Taiwan under the contract No. NSC92-2218-E-002-056.

7. REFERENCES

[1] M. Alexa and W. Müller. Representing animations by principal components. *Computer Graphics Forum (Eurographics 2000 Conference Proceedings)*, 19(3):411–418, 2000.

[2] C. L. Bajaj, V. Pascucci, and G. Zhuang. Progressive compressive and transmission of arbitrary triangular meshes. In *IEEE Visualization 1999 Conference Proceedings*, pages 307–316, 1999.

[3] H. M. Briceno, P. V. Sander, L. McMillan, S. Gortler, and H. Hoppe. Geometry videos: a new representation for 3d animations. In *Proceedings of ACM Symposium on Computer Animation 2003*, pages 136–146, 2003.

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

[5] J. H. Clark. Hierarchical geometric models for visible surface algorithms. *Communication of the ACM*, 19(10):547–554, 1976.

[6] J. El-Sana and A. Varshney. Generalized view-dependent simplification. *Computer Graphics Forum (Eurographics 1999 Conference Proceedings)*, 18(3):83–94, 1999.

[7] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. *ACM SIGGRAPH 1997 Conference Proceedings*, pages 209–216, 1997.

[8] X. Gu, S. J. Gortler, and H. Hoppe. Geometry images. *ACM Transactions on Graphics (SIGGRAPH 2002 Conference Proceedings)*, 21(3):355–361, 2002.

[9] S.-H. Guan, S.-Y. Cho, Y.-T. Shen, R.-H. Liang, B.-Y. Chen, and M. Ouhyoung. Conceptual farm. In *Proceedings of IEEE International Conference on Multimedia & Expo 2004*, 2004.

[10] A. Gueziec, G. Taubin, B. Horn, and F. Lazarus. A framework for streaming geometry in vrml. *IEEE Computer Graphics and Applications*, 19(2):68–78, 1999.

[11] H. Hoppe. Progressive meshes. In *ACM SIGGRAPH 1996 Conference Proceedings*, pages 99–108, 1996.

[12] H. Hoppe. View-dependent refinement of progressive meshes. In *ACM SIGGRAPH 1997 Conference Proceedings*, pages 189–198, 1997.

[13] H. Hoppe. Smooth view-dependent level-of-detail control and its application to terrain rendering. In *IEEE Visualization 1998 Conference Proceedings*, pages 35–42, 1998.

[14] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *ACM SIGGRAPH 1993 Conference Proceedings*, pages 19–26, 1993.

[15] J. Houle and P. Poulin. Simplification and real-time smooth transitions of articulated meshes. In *Graphics Interface 2001 Conference Proceedings*, pages 55–60, 2001.

[16] D. Luebke and C. Erikson. View-dependent simplification of arbitrary polygonal environments. In *ACM SIGGRAPH 1997 Conference Proceedings*, pages 199–208, 1997.

[17] J. Popović and H. Hoppe. Progressive simplicial complexes. In *ACM SIGGRAPH 1997 Conference Proceedings*, pages 217–224, 1997.

[18] W. J. Schroeder. A topology modifying progressive decimation algorithm. In *IEEE Visualization 1997 Conference Proceedings*, pages 205–212, 1997.

[19] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of Triangle Meshes. *ACM Computer Graphics (SIGGRAPH 1992 Conference Proceedings)*, 26(2):65–70, 1992.

[20] A. Shamir and V. Pascucci. Temporal and spatial level of details for dynamic meshes. In *Proceedings of ACM Symposium on Virtual Reality Software and Technology 2001*, pages 77–84, 2001.

[21] M. Soucy and D. Laurendeau. Multiresolution surface modeling based on hierarchical triangulation. *Computer Vision and Image Understanding*, 63(1):1–14, 1996.

[22] J. C. Xia and A. Varshney. Dynamic view-dependent simplification for polygonal models. In *IEEE Visualization 1996 Conference Proceedings*, pages 327–334, 1996.

[23] J. Rossignac and P. Borrel. Multi-resolution 3D approximations for rendering complex scenes. In *Proceedings of Geometric Modeling in Computer Graphics*, pages 455–465, 1993.