

Loop Progressive Geometry Compression for Triangle Meshes

P. Mongkolnam¹ A. Razdan² G. Farin³

Abstract

Subdivision surfaces are finding their way into many Computer Aided Design (CAD) and animation packages. They have been used in lossy 3D mesh compression with possible applications such as level of details (LOD), progressive transmission and 3D streaming. Also, in recent years 3D mesh compression has become necessary for transferring and browsing 3D objects over the internet. This paper explores developing one such method for Loop subdivision surfaces, which can be used as a lossy compression scheme.

The method presented is simple, yet efficient and effective compared to other existing 3D mesh coders, such as Loop subdivision wavelets. A new approach is used to remesh a mesh of arbitrary topology without resorting to parametrization. The method then applies a new technique to solve the inverse Loop subdivision surface problem without resorting to subdivision wavelets resulting in multiresolution meshes which can be used as progressive geometry compression.

Keywords: Mesh compression, subdivision surfaces, remeshing, multiresolution meshes, subdivision wavelets

1. Introduction

Dense triangle meshes are highly detailed and are expensive to represent, store, transmit and manipulate. *Loop subdivision surfaces* [17] allow for compact storage and simple representation (as a base triangle control mesh) and can be evaluated on the fly to any resolution. Its simplicity makes it a preferred choice for animation, texture mapping, editing and manipulation as illustrated by Zorin et al. [28] and DeRose et al. [3].

The *goal* of this work is to approximate a smooth and highly detailed surface of an original mesh while still achieving a high compression ratio and low distortion (L^2 distance error, defined later). Our method is simple, yet efficient and effective compared to other existing 3D mesh coders, such as 3DCT of 3D Compression Technologies, Inc. [1] and Progressive Geometry Compression by Khodakovskiy et al. [11]. Our geometry compression is based on smooth *semi-regular* meshes using a Loop scheme with *scalar-valued displacement* enabling us to dismiss most connectivity and parameter information as opposed to vector-valued displacement as used by Krishnamurthy et al. [14]. In order to have a very high compression ratio and low distortion while preserving surface details, we obtain a base mesh with sequences of negligible or very small magnitude of displacement values (details).

The main *contributions* of this work are:

- A novel approach for speeding up the remeshing process without resorting to parametrization and optimization as done in Hoppe et al. [9], Eck et al. [5], Lee et al. [15], Kobbelt et al. [13] and Khodakovskiy et al. [12].

- A novel approach of solving the inverse Loop subdivision problem by recognizing that the *Loop edge points* are redundant and not contributing any new information to solving the inverse Loop subdivision surface problem, without resorting to subdivision wavelets as done in Eck et al. [5], Lounsbery et al. [19] and Khodakovskiy et al. [11].

2. Related work

A subdivision surface is defined by a refinement of an initial control mesh. In the limit of the refinement process, a smooth surface is obtained. A triangular based subdivision scheme was introduced by Loop [17], which was a generalization of C^2 quartic triangular B-splines.

Hoppe et al. [9] presented a method to fit a piecewise smooth surface to scattered range data points using Loop subdivision scheme [17]. The method can model arbitrary topological types of a surface. The subdivision rules were locally modified to model sharp features such as creases, darts and corners.

Suzuki et al. [24] used a *Loop subdivision limit position* to repeatedly adjust a control mesh and subdivide it to fit the data points of arbitrary topological objects in their method. It quickly captures the geometrical shape of the object because the fitting is done locally at each vertex instead of having to solve a large linear system of equations as done in the third phase of Hoppe et al. [9]. However, their method fails to preserve the sharp features. This method is different from our method in that it is not a multiresolution mesh method, and it does not solve any inverse Loop subdivision surface problem.

Lee et al. [16] proposed a new surface representation to an arbitrary triangle mesh, namely a *displaced subdivision surface*. Unlike the method proposed by Krishnamurthy et al. [14], it generates a detailed surface by displacing a scalar-valued offset along Loop limit normals over a smooth domain surface. This method is different from our method in that it uses parametrization to do remeshing and to compute displacement values, and it does not solve any inverse Loop subdivision surface problem.

Guskov et al. [8] constructed the *normal semi-regular meshes* with all displacement values (wavelet coefficients) lying exactly in a normal direction to approximate any surface. The normal mesh is a multiresolution mesh whose hierarchical displacement is successively applied to the mesh as it is subdivided starting from some coarse level. To make an arbitrary mesh the normal mesh, a continuous readjustment of parametrization using MAPS scheme [15] is needed at each subdividing level. The normal mesh can be seen somewhat as a generalization of the displaced subdivision surfaces [16]. This method is different from our method in that it uses the continuous parametrization readjustment and does not require solving inverse surface problem to find displacement values at each subdivision level.

Wavelet representations have been known to be very effective in de-correlating the original data for image coding. Likewise, in 3D piecewise smooth models, neighboring vertices are highly correlated, and the wavelet transform can remove a large amount of the correlation which is the key to compression. Lounsbery et al. [18], [19] have worked on wavelet decompositions of arbitrary topological surfaces. They are typically based on *interpolating subdivision schemes* such as Butterfly scheme [4] and a modified Butterfly scheme [27]. These constructions are subsequently applied to a progressive approximation of data on surfaces as done by Schröder et al. [23], Nielson et al. [20]. Khodakovskiy et al. [11] extended the subdivision wavelets of Lounsbery et al. [19] to include *Loop subdivision wavelets* for their progressive mesh compression. A target application for the wavelet decompositions

¹ School of Information Technology, King Mongkut's University of Technology Thonburi, Thailand (pornchai@it.kmutt.ac.th).

² Partnership for Research In Spatial Modeling (PRISM), Arizona State University (Razdan@asu.edu)

³ Department of Computer Science and Engineering, Arizona State University (farin@asu.edu)

of surfaces is the compression of densely sampled and highly detailed surfaces. Our method has the same goal as that of the subdivision wavelet based methods in that both methods solve the inverse subdivision surface problem to obtain small magnitude displacement values, but our method is based on the inverse Loop subdivision matrix rather than the Loop subdivision wavelets. Both methods also differ on how to do remeshing.

In summary, our work can be viewed as a combination of ideas on *displaced subdivision surfaces* of Lee et al. [16], *normal meshes* of Guskov et al. [8], *mesh multiresolution analysis* of Lounsbery et al. [19] and *progressive geometry compression* of Khodakovsky et al. [11]. However, our method uses a novel approach to do remeshing and a novel approach to solve the inverse Loop subdivision surface problem.

3. Methodology

A *problem statement* of our work is as follows:

Given: Input to our method is a triangle mesh TM of arbitrary topology. The mesh TM is topological two-manifold with or without boundary. A *manifold* is a surface in which the infinitesimal neighborhood of every point is *homeomorphic* (or topologically equivalent) to a disk, or a half-disk for a manifold with boundary. Surfaces with boundary are not closed surfaces. M is defined as a pair of (K, V) where K is a *simplicial complex*. The simplicial complex K specifies a connectivity of vertices, edges and faces, and therefore it determines the mesh topology. V is a set of vertex positions of the mesh, and it specifies a shape or geometry of the mesh.

Goal: Output a *base mesh* and a set of *displacement values*. The base mesh is a coarse control mesh that is obtained after some iteration of inverse Loop steps. After each inverse Loop step, a set of displacement values is computed. Using both the base mesh and sets of displacement values, we can reconstruct an approximation to the original mesh TM . To reconstruct the approximation of the original mesh, the displacement values are hierarchically added to the *subdivided base mesh* after each subdivision level starting from the coarsest level (the base mesh) to some finer levels, as exemplified by Fig. 9.

Method: TM is decimated to a *simplified mesh*. The simplified mesh is adjusted and then subdivided. The subdivided mesh is then displaced, thus having all vertices lying on the input mesh. The displaced mesh is called a *remesh*. We then apply the inverse Loop scheme to the remesh, and output the *base mesh* and a set of *displacement values*. A flowchart of the method is given in Fig. 1.

The method can be summed up in the following steps:

- I. **Input:** Original triangle mesh TM .
- II. **Simplification:** A *Simplified mesh* is obtained by applying a mesh simplification algorithm based on the quadric error metric and iterative edge contractions of Garland et al. [7].

The original mesh is decimated to $\frac{1}{4^{th}}$, $\frac{1}{16^{th}}$, $\frac{1}{64^{th}}$,

$\frac{1}{256^{th}}$ or $\frac{1}{1024^{th}}$ of a total number of original triangles.

The numbers are so chosen because when a Loop subdivision scheme (in Step IV) is applied to the simplified mesh *one, two, three, four* or *five* times, respectively, the total number of the mesh triangles is close to that of the original mesh.

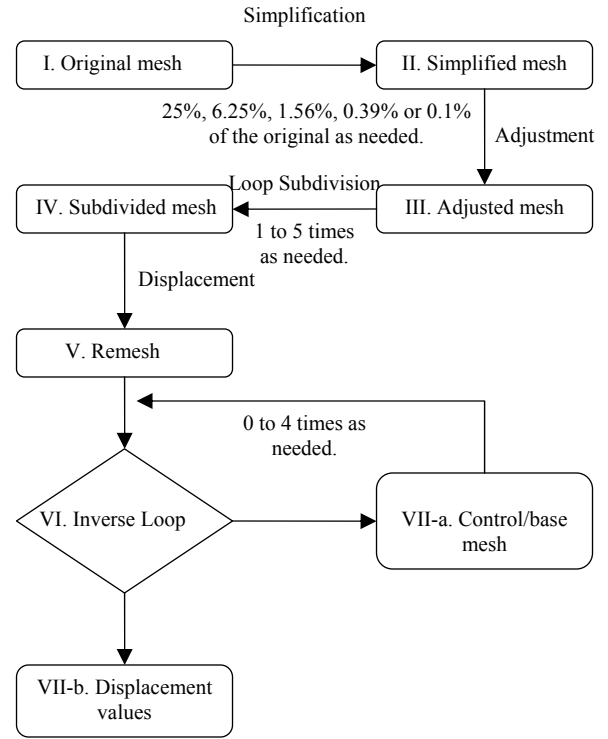


Fig. 1. Flow chart of our method.

- III. **Adjustment:** The *Adjusted mesh* is obtained after applying the local iterative approximation method of Suzuki et al. [24] to the simplified mesh. The method recursively relocates each vertex of the simplified mesh such that its *Loop limit position* (obtained by applying Loop subdivision scheme to infinity) would lie close to the original surface. This step is critical because of the smoothing and shrinking nature of the Loop subdivision scheme. If the adjustment step was omitted, the reconstructed mesh would require large magnitude of displacement values, which are not efficient for compression.
- IV. **Subdivision:** The *Subdivided mesh* is obtained after applying the Loop subdivision scheme [17] to the adjusted mesh *one, two, three, four* or *five* times according to how much the original mesh was decimated in the simplification step (Step II).
- V. **Displacement:** The *Remesh* is obtained after performing a displacement operation based on Lee et al. [16] to the subdivided mesh. Each vertex of the subdivided mesh is moved to the original surface. The remesh is so called because its connectivity has been changed from the original mesh in that the remesh is a *semi-regular* mesh. The semi-regular mesh has predominantly valence six except for vertices corresponding to vertices of the initial coarse mesh with valence not equal to six. A novel technique is used for finding *ray-triangle intersections* within the so-called *contributing areas* (explained in section 3.1.4) during the displacement operation to speed up the process without resorting to parametrization to do remeshing. Therefore we do not have to resort to finding a domain surface or a parametrization.
- VI. **Inverse:** Next we have to solve the inverse Loop subdivision surface problem. The inverse Loop operation

takes the *remesh* as input, and it outputs a *control mesh* and a set of *scalar displacement values*. The inverse process is repeated with the control mesh being used as the input instead of the remesh. The number of repetitions is *zero, one, two, three* or *four* times according to the number used in the subdivision step (Step IV). A novel approach is used for solving the inverse Loop subdivision surface problem without resorting to subdivision wavelets or a least squares approach.

VII. **Output:** The *Base mesh* and a set of *displacement values* are the output. The base mesh has the same connectivity and topology as that of the *simplified mesh*. However, its geometry (vertex positions) is different from the simplified mesh because of the displacement operation and the inverse Loop operation. Displacement values are scalar values which are hierarchically added back to the *subdivided base mesh* during a reconstruction process.

We describe step IV-VI of the method in detail below:

3.1 Remeshing

3.1.1 Subdivision

In the refinement step, the *Loop subdivision* scheme is applied one, two, three, four or five times to the *adjusted mesh* depending on how much the mesh has been decimated in the simplification step. After refinement, the *subdivided mesh* will have a number of triangles close to that of the original mesh. Even though the adjustment process best attempts to force all vertices of the simplified mesh to lie on the original surface in the limit, most of them would not lie exactly on the original surface as desired because of the finite number of subdivision levels (not really in the limit) used. In addition, newly created vertex points and edge points that are not of the simplified mesh would not be forced by the adjustment process to lie on the original surface. Therefore, all vertices of the subdivided mesh need to be displaced so that they would lie on the original surface.

The Loop subdivision scheme [17] is composed of two repeating steps: a *splitting* step and a *positioning* step. The splitting step splits each triangle face into four sub triangle faces (called a 1-to-4 split) as shown in Fig. 2. The positioning step repositions each vertex of the split triangles according to weighted averages or affine combination [6] of certain vertices before splitting. There are two types of the affine combinations which can be visualized by diagrams called masks: a *Loop vertex point* mask and a *Loop edge point* mask as shown in Fig. 2.

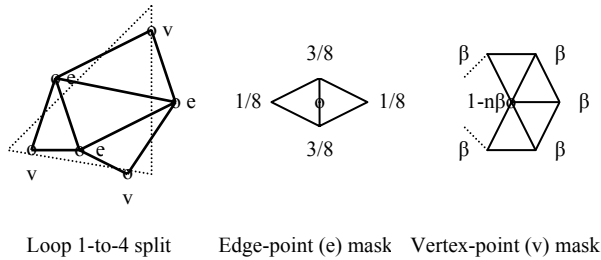


Fig. 2. Loop subdivision and its masks.

In Fig. 2, β is a weight used in the Loop vertex point mask and is defined as:

$$\beta(n) = \frac{1}{n} \left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4} \cos \frac{2\pi}{n} \right)^2 \right) \quad \text{for } n \geq 3,$$

where n is a valence of a vertex. A *valence* of a vertex is the number of edges incident to the vertex.

In general a refinement step of the *Loop subdivision* scheme can be expressed in a matrix equation form is:

$$\begin{bmatrix} v_1^j \\ v_2^j \\ \vdots \\ v_{n-1}^j \\ v_n^j \\ e_1^j \\ e_2^j \\ \vdots \\ e_{m-1}^j \\ e_m^j \end{bmatrix} = M_{(n+m) \times n} \begin{bmatrix} v_1^{(j-1)} \\ v_2^{(j-1)} \\ \vdots \\ v_{n-1}^{(j-1)} \\ v_n^{(j-1)} \end{bmatrix} \quad (1)$$

where v is a vertex point, e is an edge point, j is the subdivision level, and M is the *Loop subdivision matrix*. The entries of the matrix M are obtained from the Loop subdivision masks. Because of the overdetermined linear system of equations of Equation (1), one typically would resort to the least squares technique (via the *normal equations*) to solve for the vertices of a coarser mesh, resulting in an approximate solution. However, the exact solutions can be obtained if the Loop subdivision matrix is modified to be the *Loop vertex-point subdivision matrix*, which is *square* and *nonsingular*, as shown in section 3.2.

3.1.2 Displacement

Displacement is needed for each vertex of the *subdivided mesh* in order to capture original surface details of the *remesh*. Our method is based on the *displaced subdivision surfaces* of Lee et al. [16]. However, we use a novel approach to efficiently find the displacement values based on the *ray-triangle intersections* computation in the so-called *contributing areas*, which limit the search and thus speed up the remeshing process as described below.

A *Loop limit normal* of each vertex p_i is computed as a cross product of two vectors, \bar{u} and \bar{v} , spanning the tangent plane of the surface at the vertex, defined as [9]:

$$\bar{u} = \sum_{j \in i^*} c_j p_j$$

$$\bar{v} = \sum_{j \in i^*} c_{(j+1) \bmod n} p_j$$

where $c_j = \cos\left(\frac{2j\pi}{n}\right)$, $i^* = \{j \mid p_j \in p_{i^*}\}$ and a valence $n = |i^*|$.

The displacement value for each vertex is the *Euclidean* distance from the vertex to the intersecting point on the original surface along the vertex Loop limit normal. All vertices of the subdivided mesh are displaced by their displacement values along their limit normal directions, resulting in the *remesh*.

Finding a *ray-triangle intersection* accurately and quickly is important to our remeshing method. We speed up the computation by limiting the original surface search area (called *contributing area*) for the ray-triangle intersection computation in order to obtain the displacement value for each vertex of the *subdivided mesh*.

or $n_i \beta_i < \frac{1}{2}$

From Equation (3) $\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4} \cos \frac{2\pi}{n_i}\right)^2 < \frac{1}{2}$ (4)

To maximize the left-hand side (LHS) of Equation (4), we want the negative term to be as small as possible. One can do that by ensuring the term $\cos \frac{2\pi}{n_i}$ is zero (i.e., n_i is 4). Thus, the LHS of

Equation (4) becomes $\frac{5}{8} - \left(\frac{3}{8}\right)^2 = \frac{5}{8} - \frac{9}{64} = \frac{31}{64}$

Since $\frac{31}{64} < \frac{1}{2}$, diagonal dominance is proved.

3.2.2 Displacement Value Computation

Our scalar *displacement values* are what the wavelet coefficients are to the Loop subdivision wavelet or the wavelet transform in general. For each inverse Loop subdivision step, the displacement value of each *Loop edge point* is computed and stored for the reconstruction or the decompression phase, as illustrated in Fig. 4. We differentiate the displacement value computations of the *first inverse step* (shown in Fig. 5) from the *subsequent inverse steps* (shown in Fig. 6). However, if one is willing to trade off a slightly increased error (shown in Tab. 1) for a much faster computation time (from an order of minute down to a few seconds), the method used in the subsequent inverse steps can replace the method used in the first inverse step.

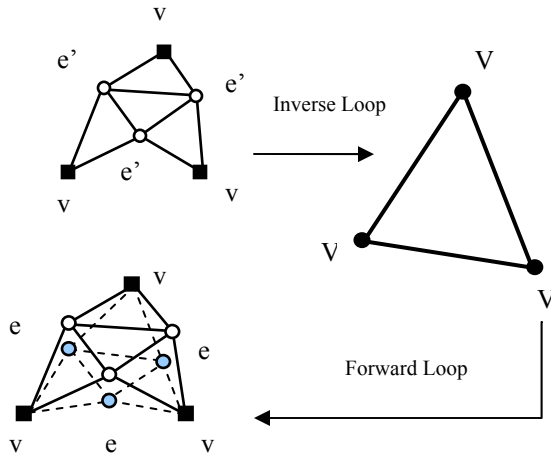


Fig. 4. Displacement between pseudo edge points, e' , (hollow circle) and the newly created edge points, e , (light cyan).

Data	#Triangles	L^2 Error	
		Intersection Approach	Dot Product Approach
Foot	19,974	0.015241	0.015341
Igea (low res.)	67,170	0.019939	0.019964
Bunny	70,556	0.017304	0.017262
Skull	160,000	0.008320	0.008344

Tab. 1. Reconstruction mesh error comparisons between the use of the intersection approach and the dot product approach in the first inverse step.

The first *control mesh* is obtained after the first inverse Loop. To obtain the *displacement values* at this level, the control mesh is subdivided corresponding to the Loop vertex point mask and the Loop edge point mask (see also the forward Loop in Fig. 4). The *vertex points* do not require any displacement values because the *Loop vertex point subdivision matrix* used in Equation (2) is square and nonsingular, and that results in exact solution as previously described. However, every *edge point* requires a displacement value. To compute the displacement values for the edge points, the *Euclidean distance* from each edge point to the intersecting point on the original surface is computed along the direction of the *Loop limit normal* as shown in Fig. 5.

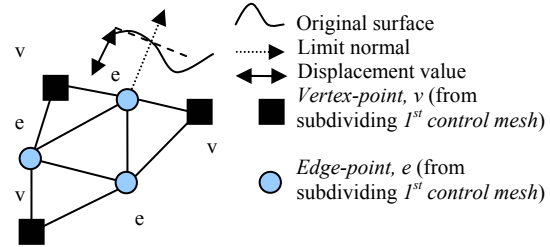


Fig. 5. Displacement value after a 1st inverse Loop.

The inverse Loop process is repeated a few more times (from zero to four times). This depends on the decimation desired (in the simplification step). The *base mesh* (the coarsest control mesh) is obtained after completing multiple inverse Loop processes. Like in the first inverse Loop step, none of the *vertex points* requires the displacement value at each level, but all the *edge points* do.

The displacement values in these steps are computed differently from the first inverse step because the original surface is not involved in the computation as illustrated in Fig. 6. Here the displacement values are the distances from the *edge points* to the positions of the corresponding vertices (so-called *pseudo edge points*) of the previous control mesh, and not the distances to the original surface as used in the first inverse step.

The displacement value is thus the length of the *distance vector*. Nevertheless, if that value is to be used, a vector value indicating direction for each displacement value needs to be computed and stored. However, one can resort to using the *Loop limit normal*, which can be computed on the fly, for a direction of the displacement. Therefore, the displacement value for each edge point is computed as a *dot product* of its unit Loop limit normal and the distance vector.

The displacement values so obtained are very small in magnitude because of the locally smooth nature of the dense triangle meshes where the surface does not significantly change, making our method promising for mesh compression and reconstruction.

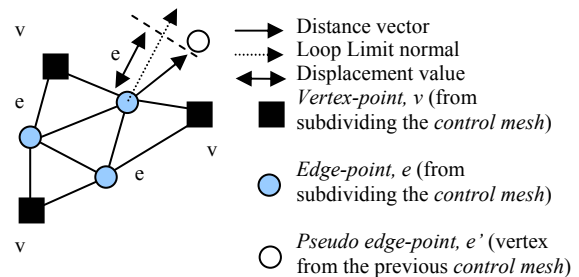


Fig. 6. Displacement value for subsequent inverse Loop.

3.2.3 Output

The output of the inverse process is the *base mesh* and a sequence of sets of displacement values. These outputs are used in the mesh reconstruction process.

To reconstruct the approximation to the input mesh, the Loop subdivision scheme [17] is applied to the base mesh for a given number of times. At each subdivision step, each displacement value of each *Loop edge point* is added along the vertex's Loop limit normal resulting in the so-called *multiresolution meshes* as illustrated in Fig. 9. Note that *Loop vertex points* do not require displacement at any subdivision level.

4. Results and Comparisons

4.1 Error Measurement

To measure the distortion between the *decompressed* or *reconstructed mesh* and the *original mesh*, we use the L^2 distance error as used in [11]. Let $d(X, Y)$ be the distance between two surfaces, X and Y . Let $d(x, Y)$ be a *Euclidean* distance from a point x on surface X to the closest point on the surface Y . Then the L^2 distance $d(X, Y)$ is given by:

$$d(X, Y) = \left(\frac{1}{\text{Area}(X)} \int_{x \in X} d(x, Y)^2 dx \right)^{1/2}$$

The $d(X, Y)$ is *scale invariant* and has no unit. This distance is not symmetric; therefore, the maximum of $d(X, Y)$ and $d(Y, X)$ is used and defined as:

$$L^2 = \max \{ d(X, Y), d(Y, X) \}$$

We use the *Metro* error measurement software by Cignoni et al. [2] to compute the L^2 distance error. In addition to the error, the software returns the length of the bounding box diagonal of the input data, which is used for the relative comparison to the obtained error.

4.2 Encoding Scheme and Quantization

Our method is conceptually similar to the subdivision wavelet in that it outputs the *base mesh* and the set of *displacement values* instead of the wavelet coefficients. The displacement values are needed to hierarchically displace each vertex of the subdivided meshes starting from the base mesh. Similarly, the wavelet coefficients in each scale are added back to the wavelet basis functions, which in turn are added back to the scaling functions to get a reconstructed signal.

Once the *base mesh* is obtained after multiple inverse Loop processes, it can be further losslessly encoded using Edgebreaker [22], the triangle mesh compression of Touma and Gotsman [25] or other compression software packages such as the 3DCT [1]. We use 3DCT in this work. The *displacement values* so obtained are small in magnitude. The magnitude of each vertex Loop limit normal vector is used as a scaling factor to further reduce the size. To take advantage of the small magnitude, the displacement value needs to be quantized to fewer-bit number instead of using the standard IEEE 32-bit floating-point number. Typically, 8-bit to 16-bit quantization levels are used in geometry coding.

To select an appropriate quantization level for our method, we experimented on various numbers of bits (from 1 to 11 bits) used to quantize each magnitude of displacement values. Example of a plot of errors versus numbers of quantization bits is given in Fig. 7. From the analysis of the plots of many data sets, the use of

more than 7 bits in quantization does not lower the error by significant amount, and thus we choose 7 bits and use it in a *variable bit coding* scheme in which 1 bit is for a *zero* value and 9 bits (1-bit *flag*, 1-bit *sign* and 7-bit *quantization*) for a *non-zero* value.

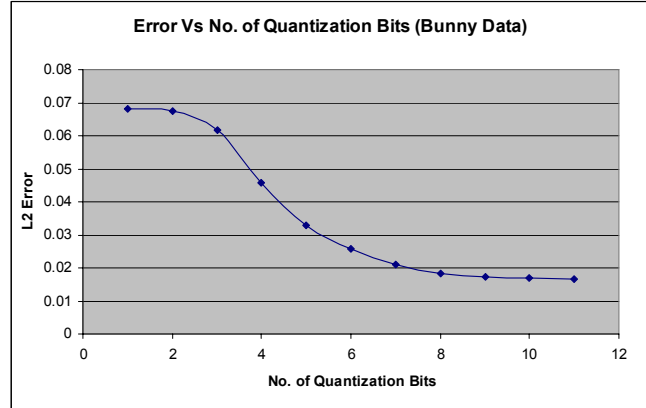


Fig. 7. Plot of L^2 error versus a different number of bits per quantization of the displacement value magnitude.

Tab. 3 shows that a large number of vertices do not require displacement values after the 7-bit quantization. Note that the number of *vertex points* is about 25% of the total number of vertices in each level, and the rest are *edge points* (because after each subdivision level, the total number of triangles increases to four times of that from the previous level, and so does the number of vertices). None of the vertex points require displacement values, but some of the edge points do.

4.3 A Note about Comparisons to Loop Subdivision Wavelets

For a lossy compression comparison, two large data sets, Igea and Feline, are used to compare our method to the Loop subdivision wavelet method [11]. Because of the limitation of the obtained subdivision wavelet software from the author of the paper (no remeshing tool available at the moment) and its oversampling for remeshing (remeshes have more vertices than the original meshes), the ideal L^2 distance errors cannot be computed accurately. The original paper does not give compression ratios either.

We would like to point out how the distortions are computed; the *original data sets* with fewer total numbers of triangles (100,000 triangles for Igea and 99,732 triangles for Feline, provided by [11]) are compared to the *reconstructed meshes* of both methods as given in Tab. 4 and Fig. 10 and Fig. 11. Because the original data sets are not of high enough resolution as compared to the reconstructed meshes, the distortions are the same for all decomposition levels and thus are not good comparisons.

We would have preferred for true comparisons the original data sets with the total number of triangles close to that of the reconstructed meshes. The *remeshes* provided from [11] (already remeshed), having the total number of triangles close to that of the reconstructed meshes, are not compared to the reconstructed meshes because the reconstructed meshes obtained from the Loop subdivision wavelet method have the *same connectivity* as that of the provided remeshes, whereas the reconstructed meshes obtained from our method do not have the same connectivity information due to our remeshing algorithm. To make the total number of triangles of our reconstructed meshes close to that of

the provided remeshes, our remeshing algorithm takes the provided remeshes as inputs and output new remeshes, resulting in different connectivity. Therefore, the distortion comparisons would be flawed. The original paper gives distortion curves for various bit rates but again the *remeshed* meshes are used for comparison.

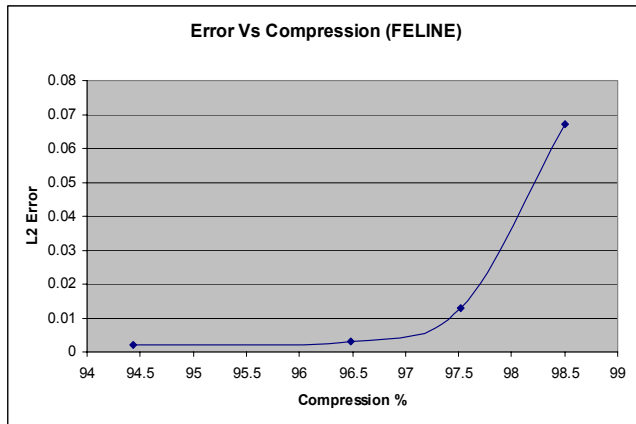


Fig. 8. Plot of L^2 errors versus compression ratios of our method at various decomposition levels (from 1 to 4 levels). The errors are computed between our reconstructed meshes and the provided remesh from [11], having close total number of triangles but with different connectivity information.

The software provided to us by the authors did not allow intermediate level meshes to be saved during decompression. Therefore we are unable to compare the error at each level. However, visually inspection revealed that the quality of our decompressed meshes was much higher to that of the *wavelets* method.

In summary, the comparisons remain *inconclusive* until we can apply the *wavelet method* on high resolution test cases with raw triangulation rather than semi regular *remeshed* meshes. If that is the case, we expect the distortions of our method to be lower at the lower decomposition levels as illustrated in Fig. 8. In Fig. 8, the distortions are computed between our *reconstructed meshes* and the *provided remeshes* from [11]; both meshes have the total number of triangles close to each other and have *different connectivity* information.

4.4 Runtime

Data	#T	Runtimes		
		Remesh	Compress	Decompress
Foot	19,974	11.80 sec.	1.61 sec.	0.86 sec.
Bunny	70,556	41.25 sec.	6.16 sec.	3.06 sec.
Skull	160,000	1:28 min.	15.17 sec.	6.92 sec.
Igea	397,312	3:30 min.	46.06 sec.	16.94 sec.
Feline	516,096	4:23 min.	1:08 min.	22.03 sec.

Tab. 2. Runtimes of various steps of our method

The compression time is the time for solving the inverse Loop vertex-point subdivision matrix and for computing displacement

values. The decompression time is the time for mesh reconstruction by hierarchically adding corresponding displacement values along each vertex's Loop limit normal starting from the base mesh. Our runtimes were recorded on the Intel Pentium 4 based computer with a clock speed of 2.266 GHz.

5. Conclusions and Future Work

We have fulfilled the goal of this work by successfully developing a method to approximate 3D triangle meshes with a high compression ratio, while keeping high surface details but a low L^2 distance error. The main *contributions* of this work are:

- A novel technique of speeding up the remeshing process, without resorting to parametrization and optimization as done in [5], [10], [12], [13], [15].
- A novel approach of solving the inverse Loop subdivision matrix by recognizing that the *Loop edge points* are redundant and not contributing any new information in solving the inverse Loop subdivision surface problem, without resorting to the least squares or subdivision wavelets as done in [5], [11], [18], [19].

Our method is simple, yet efficient and effective compared to other existing state-of-the-art 3D mesh coders such as the Loop subdivision wavelet [11]. The runtimes of our method on the remeshing, the compression (for solving the inverse surface problem) and the decompression (for reconstruction) are fast.

We have shown here that a mesh of arbitrary topology of any genus can be approximated with a high compression ratio and high surface details. According to Peng et al. [21] and to the best of our knowledge, the best approximating 3D mesh compression method in terms of the compression ratio is based on the Loop subdivision wavelet [11]. Over the years both the wavelet and the subdivision surfaces have become mature. On the other hand, our work is still in an early stage, but the method has consistently shown to be successful even in the comparison to the subdivision wavelet method.

Improvement on the data structure used in keeping the records of the *contributing areas* would result in a faster computation time for remeshing. Other mesh simplification methods in addition to that of Garland et al. [7] may be further explored, modified and used in order to have more control on the obtained *simplified mesh* because it is important to the final output surface.

5. Acknowledgments

This work was supported in part by the National Science Foundation (grant IIS-9980166) and the Ministry of Science, Technology and Environment of Thailand. For more information on the 3D Knowledge project, please visit <http://3dk.asu.edu>. The authors would like to thank various members of the Partnership for Research in Spatial Modeling (PRISM) team at Arizona State University for their support.

We would like to thank A. Khodakovsky, P. Cignoni and 3D Compression Technologies, Inc. for the Progressive Geometry Compression software, the error measurement software and the 3D mesh compression software, respectively. Horse, Igea and Feline are courtesy of Cyberware, Inc. Bunny is courtesy of the Stanford Computer Graphics laboratory. Foot is courtesy of the Raindrop Geomagic, Inc. Skull is courtesy of Headus, Inc.

6. References

- [1] 3DCT of 3D Compression Technologies, Inc., at <http://www.3dcompress.com>.

- [2] P. Cignoni, C. Rocchini and R. Scopigno, "Metro: Measuring Error on Simplified Surfaces," *Computer Graphics Forum*, vol. 17, no. 2, pp. 167-174, June 1998.
- [3] T. DeRose, M. Kass and T. Truong, "Subdivision Surfaces in Character Animation," *Proceedings of the 25th annual conference on Computer graphics and interactive techniques (SIGGRAPH 1998)*, pp. 85-94, July 1998.
- [4] N. Dyn, D. Levin and J.A. Gregory, "A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control," *ACM Transactions on Graphics (TOG)*, vol. 9, no. 2, pp. 160-169, April 1990.
- [5] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery and W. Stuetzle, "Multiresolution Analysis of Arbitrary Meshes," *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (SIGGRAPH 1995)*, pp. 173-182, September 1995.
- [6] G.E. Farin. *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide*. 5th Edition. Morgan Kaufmann, 2001.
- [7] M. Garland and P.S. Heckbert, "Surface Simplification using Quadric Error Metrics," *Proceedings of the 24th annual conference on Computer graphics and interactive techniques (SIGGRAPH 1997)*, pp. 209-216, August 1997.
- [8] I. Guskov, K. Vidimce, W. Sweldens and P. Schröder, "Normal Meshes," *Proceedings of the 27th annual conference on Computer graphics and interactive techniques (SIGGRAPH 2000)*, pp. 95-102, July 2000.
- [9] H. Hoppe, T. DeRose, T. Duchamp and M. Halstead, "Piecewise Smooth Surface Reconstruction," *Proceedings of the 21st annual conference on Computer graphics (SIGGRAPH 1994)*, pp. 295-302, 1994.
- [10] K. Hormann, U. Labsik and G. Greiner, "Remeshing Triangulated Surfaces with Optimal Parametrizations," *Computer-Aided Design (CAD)*, vol. 33, no. 11, pp. 779-788, September 2001.
- [11] A. Khodakovsky, P. Schröder and W. Sweldens, "Progressive Geometry Compression," *Proceedings of the 27th annual conference on Computer graphics and interactive techniques (SIGGRAPH 2000)*, pp. 271-278, July 2000.
- [12] A. Khodakovsky, N. Litke and P. Schröder, "Globally Smooth Parametrizations with Low Distortion," *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3, pp. 350-357, July 2003.
- [13] L. Kobbelt, J. Vorsatz, U. Labsik and H.P. Seidel, "A Shrink Wrapping Approach to Remeshing Polygonal Surfaces," *Computer Graphics Forum 18, Eurographics '99 issue*, pp. 119-130, 1999.
- [14] V. Krishnamurthy and M. Levoy, "Fitting Smooth Surfaces to Dense Polygon Meshes," *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (SIGGRAPH 1996)*, pp. 313-324, August 1996.
- [15] A. Lee, W. Sweldens, P. Schröder, L. Cowsar and D. Dobkin, "MAPS: Multiresolution Adaptive Parametrization of Surfaces," *Proceedings of the 25th annual conference on Computer graphics and interactive techniques (SIGGRAPH 1998)*, pp. 95-104, July 1998.
- [16] A. Lee, H. Moreton and H. Hoppe, "Displaced Subdivision Surfaces," *Proceedings of the annual conference on Computer graphics (SIGGRAPH 2000)*, pp. 85-94, July 2000.
- [17] C. Loop. *Smooth Subdivision Surfaces Based on Triangles*. Master's Thesis, University of Utah, Department of mathematics, 1987.
- [18] M. Lounsbery. *Multiresolution Analysis for Surface of Arbitrary Topological Type*. PhD Thesis, University of Washington, 1994.
- [19] M. Lounsbery, T. DeRose and J. Warren, "Multiresolution Analysis for Surfaces of Arbitrary Topological Type," *ACM Transactions on Graphics (TOG)*, vol. 16, no. 1, pp. 34-73, January 1997.
- [20] G.M. Nielson, I.H. Jung and J. Sung, "Haar Wavelets over Triangular Domains with Applications to Multiresolution Models for Flow over a Sphere," *Visualization 1997. IEEE Proceedings*, pp. 143-149, October 1997.
- [21] J. Peng, C.S. Kim and C.C.J. Kuo. *Technologies for 3D Triangular Mesh Compression: A survey*. U. of Southern California, February 2003.
- [22] J. Rossignac, "Edgebreaker: Connectivity Compression for Triangle Meshes," *IEEE Transactions on Visualization and Computer Graphics*, 5(1), pp. 47-61, January-March 1999.
- [23] P. Schröder and W. Sweldens, "Spherical Wavelets: Efficiently Representing Functions on the Sphere," *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (SIGGRAPH 1995)*, pp. 161-172, September 1995.
- [24] H. Suzuki, S. Takeuchi and T. Kanai, "Subdivision Surface Fitting to a Range of Points," *The 7th Pacific Conference on Computer Graphics and Applications, IEEE Proceedings*, pp. 158-167, 1999.
- [25] G. Touma and C. Gotsman, "Triangle Mesh Compression," *Graphics Interface*, pp. 26-34, 1998.
- [26] R.S. Varga. *Matrix Iterative Analysis*. Prentice-Hall, Inc., NJ USA, 1962.
- [27] D. Zorin, P. Schröder and W. Sweldens, "Interpolating Subdivision for Meshes with Arbitrary Topology," *Proceedings of the 23rd annual conference on Computer graphics (SIGGRAPH 1996)*, pp. 189-192, 1996.
- [28] D. Zorin, P. Schröder and W. Sweldens, "Interactive Multiresolution Mesh Editing," *Proceedings of the 24th annual conference on Computer graphics (SIGGRAPH 1997)*, pp. 259-268, August 1997.

Input		Output Displacement Values				
Data	#Vertices	Level	#Vertices	%Vertex points	%Edge points with zero displacement values	Total % of vertices with zero displacement values
Foot	10,010	Base mesh	636			
		1	2,518	25.26	11.00	36.26
		2	10,023	25.12	29.72	54.84
Teapot	26,235	Base mesh	1,639			
		1	6,556	25.00	66.24	91.24
		2	26,224	25.00	68.88	93.88
Igea (low res.)	33,587	Base mesh	2,102			
		1	8,398	25.02	8.86	33.88
		2	33,586	25.00	20.47	45.47
Bunny	35,280	Base mesh	2,206			
		1	8,818	25.02	13.60	38.62
		2	35,266	25.00	24.34	49.34
Horse	48,485	Base mesh	3,032			
		1	12,122	25.01	55.02	80.03
		2	48,482	25.00	72.15	97.15

Tab. 3. Statistics of the 7-bit quantization of the displacement values

Input	Comparisons				Loop Subdivision Wavelet
	Our Method at Various No. of Decomposition Levels				
Remesh Data #Triangles (BBox-Diag)	1	2	3	4	
	#Triangles	#Triangles	#Triangles	#Triangles	#Triangles
Igea (high res.) 397,312 (157.47 units)	397,312	397,312	396,672	396,288	397,312
Feline 516,096 (145.76 units)	516,096	516,096	515,200	515,072	516,096

Input	Our Method		Loop Subdivision Wavelet	
	CR	L^2	CR	L^2
Igea (high res.)	98.01	1.22	99.26	1.22
Feline	98.50	0.80	99.18	0.80

Tab. 4. Our compression and L^2 error comparisons to Loop subdivision wavelet

Note:

BBox-Diag is the bounding box diagonal of the data measured in units. CR is compression ratio (%). Decompositions indicate the number of decomposition levels applying to the data. L^2 is L^2 error computed between the reconstructed mesh and the original mesh. Original data sets are provided, but they do not have the same resolution as the provided remesh. The provided original Feline has just 99,732 triangles, and the Igea has 100,000 triangles. Ideally, the error should be computed from the *higher resolution original mesh*, having a number of total triangles close to the reconstructed meshes and used as inputs to the remeshing tool of the Loop subdivision wavelet software [11] and our method, as explained in section 4.3.

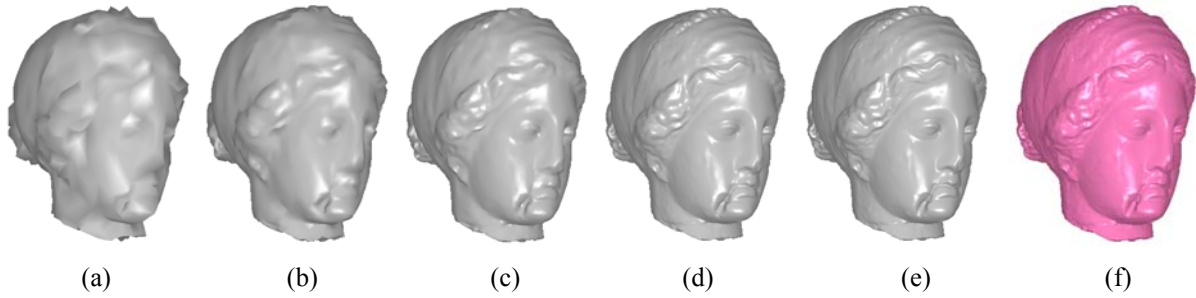


Fig. 9. Multiresolution meshes from our method. (a) Base mesh with 1,548 triangles. (b) 1st level. (c) 2nd level. (d) 3rd level. (e) 4th and final reconstruction with 396,288 triangles (an approximation to the original Igea) with L^2 error of 0.030794. (f) Original Igea with 397,312 triangles with bounding box diagonal of 157.47 units.

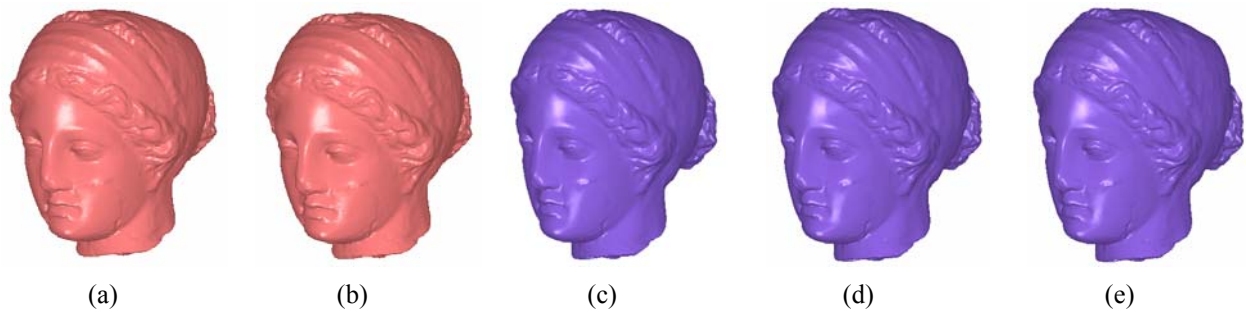


Fig. 10. Igea data compression and L^2 error comparisons of reconstructed meshes of Loop subdivision wavelets and our method at various decomposition levels. See also Tab. 4. (a) Original. (b) Loop subdivision wavelets. (c) Our method with 4 decomposition levels. (d) Our method with 3 decomposition levels. (e) Our method with 2 decomposition levels.

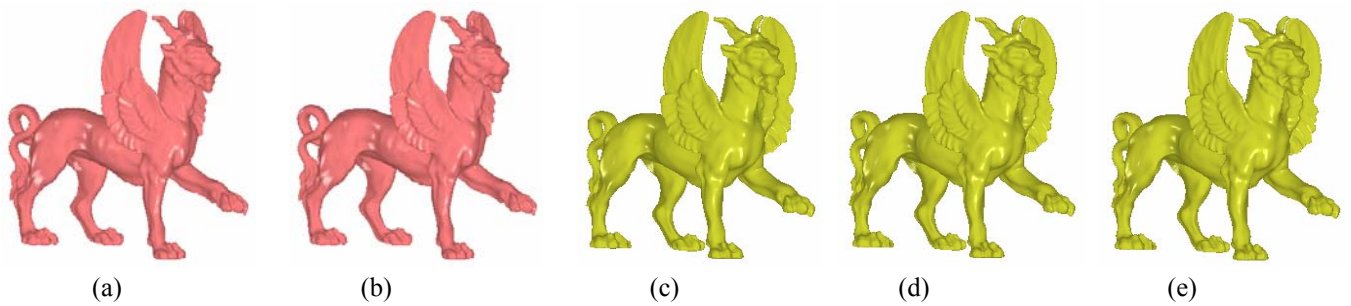


Fig. 11. Feline data compression and L^2 error comparisons of reconstructed meshes of Loop subdivision wavelets and our method at various decomposition levels. See also Tab. 4. (a) Original. (b) Loop subdivision wavelets. (c) Our method with 4 decomposition levels. (d) Our method with 3 decomposition levels. (e) Our method with 2 decomposition levels.