Reverse Engineering Using Loop Subdivision

Pornchai Mongkolnam[1], Anshuman Razdan[2] and Gerald E. Farin[3]

[1]King Mongkut's University of Technology Thonburi, Pornchai@it.kmutt.ac.th
[2]Arizona State University, Razdan@asu.edu
[3]Arizona State University, Farin@asu.edu

ABSTRACT

Subdivision surfaces have become popular in Computer Aided Design (CAD) and animation packages. Popular choices include Loop, Catmull-Clark, Doo-Sabin, etc. Subdivision surfaces have many advantages over the traditional use of NURBS, which are problematic where multiple patches meet. Possible applications of subdivision surfaces are surface reconstruction, mesh compression and reverse engineering of dense triangle meshes. We present the Loop subdivision scheme as a tool to approximate dense triangle meshes of arbitrary topology. The paper shows the process as well as some satisfactory results of CAD models.

Keywords: Loop subdivision surfaces; Surface reconstruction; Multiresolution meshes; Triangle mesh; Approximation.

1. INTRODUCTION

Laser scanned 3D objects can consist of millions of points. These point clouds can then be triangulated by various methods such as 3D alpha shapes [5]. The resulting triangular meshes are piecewise linear surfaces which are not optimal for rendering, editing, modeling, etc. Dense triangle meshes that are highly detailed are expensive to represent, store, transmit and manipulate. This problem is typically solved by the process of *reverse engineering* [20].

Typically, tensor product NURBS [6] or B-spline surfaces are used to approximate triangle meshes. Multiple B-spline patches are needed for arbitrary topological surfaces; however, there are some geometric continuity conditions that must be met for adjacent patches, and using them is not the most desirable approach since it requires high-dimensional constraint optimization.

In this paper, we explore how to use Loop surfaces [13, 17] instead of a NURBS representation. Loop surfaces do not suffer from the above problems, allow for compact storage and simple representation (as a base triangle control mesh) and can be evaluated on the fly to any resolution.

The *motivation* for this work is to reconstruct dense triangle meshes of mechanical parts having feature edges such as creases, corners and darts [9] using both the Loop subdivision scheme [13, 9] and the scalar-valued displacement method [12]. The method has been applied successfully for a general smooth data, and it is extended to handle data with feature edges such as those found in CAD models. The method enables us to dismiss most connectivity and parameter information as opposed to a method that uses vector-valued displacement [10], and thus making our method suitable for possible applications such as 3D mesh compression.

In addition, the proposed method is fully automatic, i.e., it does not require user intervention at any step. This is in contrast to most NURBS reverse engineering software available today.

The *main contributions* of this paper are: *first*, a novel approach of remeshing without resorting to parametrization like other techniques [4, 11]. We modified our approach for mechanical parts to preserve sharp edges (as illustrated in Fig. 7). *Second*, a new method to obtain a multiresolution mesh where each level can be written as a normal offset from a coarser level by recursively solving the inverse Loop subdivision matrix without resorting to wavelets such as multiresolution analysis of Lounsbery et al. [14] or continuous readjustment of parametrizations of Guskov et al. [8] normal meshes. Fig. 5 and Fig. 6 illustrate examples of our multiresolution meshes, where the base meshes are recursively subdivided and displacement values are added along vertices' Loop limit normals at each subdivision level.

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. Doo and Sabin [3] and Catmull and Clark [1] first introduced subdivision schemes for arbitrary meshes. Their schemes respectively generalized bi-quadratic and bi-cubic tensor product B-splines. The triangular based subdivision scheme was introduced by Loop [13], which was a generalization of C^2 quartic triangular B-splines.

Some authors have combined both B-splines and subdivision schemes for surface fitting and surface reconstruction. Takeuchi et al. [19] used a Doo-Sabin subdivision scheme to fit a surface to a dense triangular mesh and automatically construct B-spline surfaces from the subdivision surface. Ma et al. [15] proposed a Catmull-Clark subdivision surface fitting as a network of smoothly connected bi-cubic B-spline surfaces.

Hoppe et al. [9] presented a piecewise smooth surface fitting method to scattered range data points using Loop subdivision scheme to fit the data through an optimization process. The method could model surface of arbitrary topology. The subdivision rules were locally modified to model sharp features such as creases, darts and corners. An energy function was employed to fit a piecewise smooth subdivision surface to the piecewise linear surface.

Ma et al. [16] presented a direct approach for subdivision surface fitting of a dense triangle mesh of arbitrary topology. The control mesh is obtained through least squares fitting. Their work can be viewed as an extension of the work on subdivision surface fitting of Suzuki et al. [18] in that it can handle sharp features.

Lee et al. [12] proposed a new surface representation of an arbitrary triangle mesh, namely a *displaced subdivision surface*. It generates a detailed surface by displacing a scalar-valued offset over a smooth parametric domain surface instead of using a vector-valued displacement map. To obtain an initial control mesh, a sequence of edge collapse transformation is used to simplify the original mesh. The smooth domain surface is then obtained by applying a number of levels of Loop subdivision to the initial control mesh, and then at each vertex of this subdivided mesh the surface limit point as well as its normal is computed. Finally, the signed distance is computed from the limit point to the original surface along the normal. To get the reconstructed smooth surface to the data points, the control mesh is subdivided to a given level where the displacement was computed, and then the displacement map is added to the subdivided mesh producing the approximated smooth surface.

Guskov et al. [8] constructed *normal semi-regular meshes* with all wavelet coefficients (displacements) 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 is needed at each subdividing level. The normal mesh can be seen as a generalization of the displaced subdivision surfaces [12].

3. LOOP RECONSTRUCTION

The works in the area of displaced subdivision surfaces [12], normal meshes [8] and multiresolution analysis [14] have been a motivation factor for our interest and research. Our fundamental idea is collectively based on those works; however, we propose a novel approach of performing a mesh multiresolution analysis by solving the inverse Loop subdivision and using a novel approach of remeshing.

A *problem statement* of our method is posed as follows. Given an arbitrary topological mesh with or without feature edges, a new mesh is reconstructed to approximate the input mesh. The method has been applied successfully with smooth data, and now it is extended to data with sharp features. Fig. 1 illustrates a flow chart of our method. The method can be summed up in the following steps.

I. Input: *original triangle mesh* of any topological type. In general our method is most suitable for smooth and dense triangle meshes. However, it has been adapted to handle meshes with sharp feature edges such as those from CAD models.
II. Simplification/Decimation: obtain a *simplified mesh* using a quadric error metric (QEM) and vertex pair contractions based on a surface simplification method of Garland et al. [7]. A good simplified mesh is important to a final mesh reconstruction. The method is able to preserve sharp features.
III. Adjustment: apply the local iterative approximation method of Suzuki et al. [18] to the simplified mesh and obtain an *adjusted mesh*.
IV. Subdivision: apply a modified Loop subdivision scheme [9, 13] to the adjusted mesh to get a *subdivided mesh* having a total number of the triangles close to that of the original mesh.
V. Displacement: apply a displacement method [12] to relocate each vertex of the subdivided mesh to the original surface and obtain a *remesh*.
VI. Inverse Loop: apply an inverse Loop scheme to the remesh, resulting in a *control mesh* and *displacement values*. Repeat this step up to four times as needed depending on an exact number of times used in the subdivision step IV.

VII. Output: *base mesh (a coarsest control mesh)* and a set of *displacement values*. They are used to reconstruct an approximation to the input mesh.

Each step of the method is given in details as follows.

Step I. Input

Input mesh is a triangle mesh of arbitrary topology such as meshes of CAD models. We focus on these high precision meshes rather than other smooth meshes because sharp features of the CAD mechanical parts need to be preserved throughout an approximation process, resulting in a more challenging problem to us.

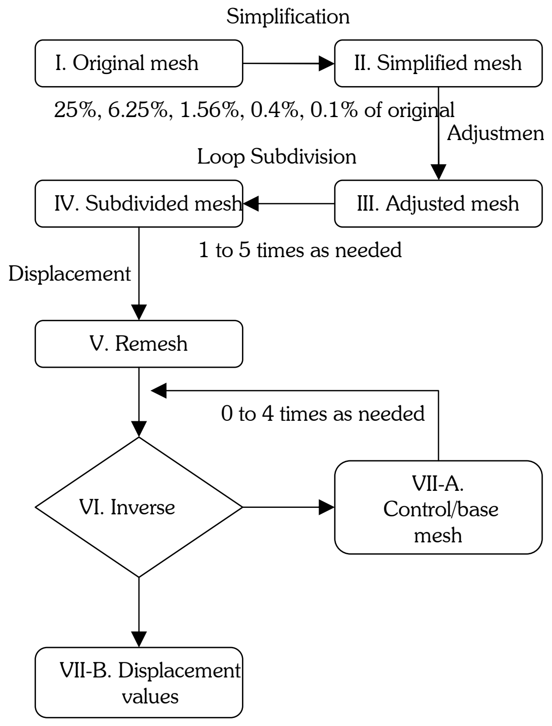

Fig. 1. Flow chart of our method.

Step II. Simplification

The input mesh is simplified (decimated) using Garland et al. [7] mesh simplification method such that the simplified mesh has approximately 1/4, 1/16, 1/64, 1/256 or 1/1024 of a total number of original triangles. These numbers are so chosen because after applying Loop subdivision scheme to the *simplified mesh* one, two, three, four or five times, respectively, the resulting mesh would have a total number of triangles close to that of the input mesh. The number of triangles increases to four times after each subdivision (see Step IV). The simplified mesh remains the original topology (genus) but not the original connectivity information.

The quality of the simplified mesh is important to the final reconstruction; therefore, an optimization process is needed in the adjustment step (Step III) after this step to adjust the simplified mesh.

Sharp features such as *creases*, *corners*, *darts and boundaries* are preserved by tagging the edges to guard against any decimation during the simplification process. A crease is where a tangent line of smooth curve on the surface is C^0, but not C^1. A corner is where three or more creases meet. A dart is where a crease terminates. Boundary edges are tagged to prevent them from deformation or decimation. These tagged edges and vertices are handled differently from others in subsequent steps.

Step III. Adjustment

In order to better capture original surface details for the *subdivided mesh*, each vertex of the *simplified mesh* needs to be adjusted such that in the limit each vertex lies close to the original surface mesh. We use the local iterative approximation method of Suzuki et al. [18] based on Loop *subdivision limit point* (SLP) to adjust each vertex of the simplified mesh. The subdivision limit point is defined as:

$$p_i^{\infty} = (1 - n\alpha)p_i + \alpha \sum_{j \in i^*} p_j \quad ; \text{where}$$

$$\alpha = \left(\frac{8\beta}{3 + 8n\beta}\right);$$

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

$$i^* = \left\{ j \mid p_j \in p_{i^*} \right\}; \quad n = | i^* |$$

Each vertex is iteratively updated as:

$$p_i{}' = p_i + \lambda\left((q_i - p_i) - \alpha_i \sum_{j \in i^*}(p_j - p_i)\right) \quad ; \quad 0 \leq \lambda \leq 1$$

q_i = closest input surface from p_i
λ = acceleration (we use 0.8)

Each tagged vertex is fixed and not affected by the adjustment.

Step IV. Subdivision

We then apply Loop scheme one to five times to the *adjusted mesh* obtaining a *subdivided mesh* with a number of triangles close to that of the original mesh. A Loop subdivision surface [13] is a refinement process applying on an initial control mesh and resulting in a finer mesh. Fig. 2 illustrates a 1-to-4 splitting on each triangle resulting in new *vertex points* (***v***) and *edge points* (***e***) and four new triangles.

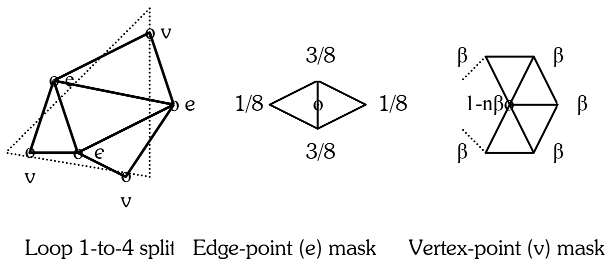

Fig. 2. Loop subdivision and its masks.

Given a dotted triangle (Fig. 2) new *vertex points* and *edge points* are computed using the given masks. Four new sub-triangles are created after applying Loop to the dotted triangle. Now assume we are given those vertex points and edge points, and we need to find vertices of the dotted triangle in an inverse problem. Clearly the *edge points* do not contribute in computation of the vertices of the dotted triangle. Only the *vertex points* affect the computation of the vertices of the dotted triangle. We will use this observation to help reduce redundancy while setting up a Loop subdivision matrix in the inverse Loop step. In general the following equation shows how a Loop subdivision matrix equation can be set up.

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

Here ***v*** is a vertex point, ***e*** is an edge point, ***i*** is a subdivision level, and ***M*** is a *Loop subdivision matrix*.

The Loop scheme is an approximating subdivision scheme, and if it is being blindly applied to the sharp features, those features would be smoothed out and could not be faithfully reconstructed. The Loop scheme needs to be modified to handle sharp features as follows: in each subdivision step the vertex points remain positions of the tagged vertices while the edge points are set to midpoints of the tagged edges. Those vertex points derived from the tagged vertices are then tagged. Likewise, the edge points derived from the tagged edges are tagged. Next, some edges of four newly created triangles are also tagged if they are parts of the tagged edges of the subdivided triangle. All these operations can be done quickly and effectively while doing Loop subdivision. Hoppe et al. [9] proposed a more sophisticated modified Loop subdivision scheme, which is more suitable for smooth surface reconstruction with some features.

Step V. Displacement

Displacement is needed for each vertex of the *subdivided mesh*. The limit normal of each vertex is computed as the cross product of two vectors spanning the tangent plane of the surface at the vertex [9]. These two vectors are defined as:

$$T_u = \sum_{j \in i^*} c_j \cdot p_j \; ; \quad T_v = \sum_{j \in i^*} c_{(j+1) \bmod n} \cdot p_j$$
$$c_j = \cos\left(\frac{2j\pi}{n}\right)$$

The displacement value for each vertex is the *Euclidean* distance from the vertex to the intersecting point on the original surface along the vertex limit normal. Each vertex is then relocated by the displacement value amount along the limit normal direction. The resulting mesh is called *remesh*. Note that no displacement is required for tagged vertices.

Finding the intersection accurately and quickly is important to the overall performance of our method. We use a novel approach to speed up the computation by limiting the search in the original surface area. During the simplification process (Step II) the two collapsed triangles corresponding to each vertex pair contraction are recorded. After a sequence of vertex pair contractions, each vertex of the *simplified mesh* can be traced back to the *original mesh* by keeping track of all the collapsed triangles. However, some vertices may not be part of any vertex pair contraction; their star triangles are recorded in that case. And the original surface search area of each vertex of the *subdivided mesh* is confined to a certain region obtained through the corresponding *simplified mesh*. This means that we do not need a domain to parametrize the input mesh, and thus resulting in a considerably faster computation.

Step VI. Inverse Loop

Let's take a closer look at Equation (1). We notice that the Loop subdivision matrix, ***M***, is not square. Let us assume we are given the left hand side of the equation including vertex points and edge points (i.e., all vertices of the *remesh*), and we want to compute a coarser control mesh (all vertices of the right hand side of the equation). First an over-determined linear system of equations is set up for the inverse problem. A naïve approach to solve for the coarser control vertices is through a *normal equation method*, which results in approximate solutions to most or all variables.

We propose a novel approach of computing the inverse Loop scheme by not considering Loop edge points, ***e***, while solving the linear system of equations. The edge points can be safely ignored because they do not contribute any new information to the linear system. They are redundant to the Loop vertex points because they are simply affine combinations of the corresponding vertices from the coarser level, which can be obtained from the vertex points as shown in the edge-point mask in Fig. 2.

Now the matrix is a square matrix, and exact solutions for the control vertices in the coarser level can then be obtained by using a *Gauss-Seidel method* for iteratively solving linear system of equations on a sparse matrix. Initial good guess solutions are already available from the *remesh* resulting in a faster convergence to the solutions. However, some of the edge points need a correction (displacement value) in a reconstruction phase when a forward Loop scheme is performed on the solved control vertices. Note that all the vertex points do not require any correction because the exact solutions are obtained when the square Loop subdivision matrix is used. Computation for the displacement values for edge points is shown in Fig. 3 and Fig. 4.

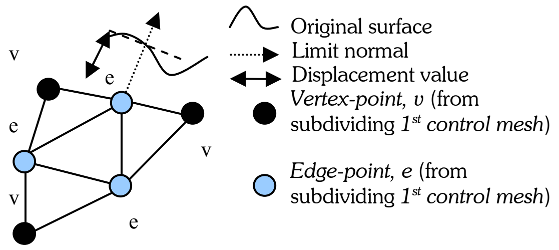
v
e
e
v
v
e
v
Original surface
Limit normal
Displacement value
Vertex-point, v (from subdividing 1st control mesh)
Edge-point, e (from subdividing 1st control mesh)

Fig. 3. Displacement value after a 1st inverse Loop.

We distinguish the first and the subsequent inverse Loop displacement value computation. In the first inverse Loop step the displacement value of each subdivided edge point is the *Euclidean* distance from it to the closest point on the original along the vertex's Loop limit normal. For the subsequent inverse Loop steps, the displacement value is the dot product of a distance vector and the vertex's unit Loop limit normal. The distance vector is computed from the Loop edge point to the corresponding vertex of the previous control mesh (pseudo edge point from a finer/previous control mesh). Note that if one is willing to trade off a slightly higher distortion for speed, the calculation of displacement value in Fig. 4 can be used for all steps of the inverse Loop process.

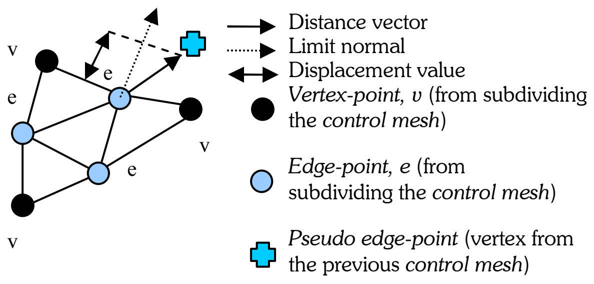
v
e
e
v
e
v
Distance vector
Limit normal
Displacement value
Vertex-point, v (from subdividing the control mesh)
Edge-point, e (from subdividing the control mesh)
Pseudo edge-point (vertex from the previous control mesh)

Fig. 4. Displacement value for subsequent inverse Loop.

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

Step VII. Output

The output is a *base mesh* (the coarsest control 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 is applied to the base mesh for a given number of times (one to five times) depending on the total number of the decomposition levels used in the inverse Loop process. At each subdivision step, each displacement value of each vertex is added along its Loop limit normal, as illustrated in Fig. 5 and Fig. 6.

4. RESULTS

We have obtained satisfactory results after applying our method to CAD parts even if the method generally performs much better on dense smooth triangle meshes. To measure the distortion of our reconstruction to the original mesh we use the Metro error measurement software [2]. The L^2 norm distance error is defined as:

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

where $d(x,Y)$ = closest distance from point x of surface X to surface Y.

The error is scale invariant, but not symmetric; therefore, the maximum of $\{E(X,Y), E(Y,X)\}$ is used. The results are shown in Fig. 8, 9 and 10. Note the change in mesh connectivity resulting from the remeshing process.

5. ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation (grant IIS-9980166) and the Ministry of Science and Technology 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 also wish to thank the reviewers of this paper for their helpful comments and suggestions.

Special thanks also to H. Hoppe for the Fandisk data, M-S Bae for providing other data used in this work and P. Cignoni for the *Metro*, the error measurement software. Igea is courtesy of Cyberware, Inc. Skull is courtesy of Headus, Inc.

6. REFERENCES

[1] *Catmull, E. and Clark, J.* Recursively generated B-spline surfaces on arbitrary topological meshes. Computer-aided design (CAD) 1978. Page(s): 350-355.

[2] *Cignoni, P., Rocchini, C. and Scopigno, R.* Metro: Measuring Error on Simplified Surfaces. Computer Graphics Forum, Volume 17, Issue 2. June 1998. Page(s): 167-174.

[3] *Doo, D., and Sabin, M.* Behavior of recursive division surfaces near extraordinary points. Computer-aided design (CAD) 1978. Page(s): 356-360.

[4] *Eck, M., Derose, T., Duchamp, T., Hoppe, H., Lounsbery, M. and Stuetzle, W.* Multiresolution analysis of arbitrary meshes. Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (SIGGRAPH '95), September 1995. Page(s): 173-182.

[5] *Edelsbrunner, H. and Mücke, E. P.* Three-dimensional alpha shapes. ACM Transactions on Graphics (TOG) January 1994. Volume 13, Issue 1. Page(s): 43-72.

[6] *Farin, G. E.* Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide. 5th Edition, 2001.

[7] *Garland, M. and Heckbert, P. S.* Surface simplification using quadric error metrics. Proceedings of the 24th annual conference on Computer graphics and interactive techniques (SIGGRAPH '97), August 1997. Page(s): 209-216.

[8] *Guskov, I., Vidimče, K., Sweldens, W. and Schröder, P.* Normal Meshes. Proceedings of the 27th annual conference on Computer graphics and interactive techniques (SIGGRAPH 2000), July 2000. Page(s): 95-102.

[9] *Hoppe, H., DeRose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schweitzer, J. and Stuetzle, W.* Piecewise Smooth Surface Reconstruction. Proceedings of the 21st annual conference on Computer graphics (SIGGRAPH '94), 1994. Page(s): 295-302.

[10] *Krishnamurthy, V. and Levoy, M.* Fitting smooth surfaces to dense polygon meshes. Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (SIGGRAPH '96), August 1996. Page(s): 313-324.

[11] *Lee, A., Sweldens, W., Schröder, P., Cowsar, L. and Dobkin, D.* MAPS: a multiresolution adaptive parameterization of surfaces. Proceedings of the 25th annual conference on Computer graphics and interactive techniques (SIGGRAPH '98), July 1998. Page(s): 95-104.

[12] *Lee, A., Moreton, H. and Hoppe, H.* Displaced Subdivision Surfaces. Proceedings of the annual conference on Computer graphics (SIGGRAPH 2000), July 2000. Page(s): 85-94.

[13] *Loop, C.* Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, Department of mathematics, 1987.

[14] *Lounsbery, M., DeRose, T.D. and Warren, J.* Multiresolution analysis for surfaces of arbitrary topological type**.** ACM Transactions on Graphics (TOG**)**. January 1997. Volume 16, Issue 1. Page(s): 34-73.

[15] *Ma, W. and Zhao, N.* Catmull-Clark surface fitting for reverse engineering applications**.** Geometric Modeling and Processing (GMP2000), Theory and Applications. Proceedings, 2000. Page(s): 274 -283.

[16] Ma, W., Ma, X., Tso, S-K. and Pan, Z. A Direct Approach for Subdivision Surface Fitting from a Dense Triangle Mesh. Computer-Aided Design (CAD), September 2003. Article in Press.

[17] *Sabin, M.* Subdivision Surfaces. In *Farin, G. E., Hoschek, J. and Kim, M.-S., editors,* Handbook of Computer Aided Geometric Design. North Holland, 1st Edition, 2002. Page(s): 309-325.

[18] *Suzuki, H., Takeuchi, S. and Kanai, T.* Subdivision Surface Fitting to a Range of Points. The 7th Pacific Conference on Computer Graphics and Applications, 1999. Page(s): 158-167.

[19] *Takeuchi, S., Kanai, T., Suzuki, H., Shimada, K. and Kimura, F.* Subdivision surface fitting with QEM-based mesh simplification and reconstruction of approximated B-spline surfaces. The 8th Pacific Conference on Computer Graphics and Applications, 2000, Page(s): 202 –212.

[20] *Varady, T. and Martin, R.* Reverse Engineering. In *Farin, G. E., Hoschek, J. and Kim, M.-S., editors,* Handbook of Computer Aided Geometric Design. North Holland, 1st Edition, 2002. Page(s): 651-681.

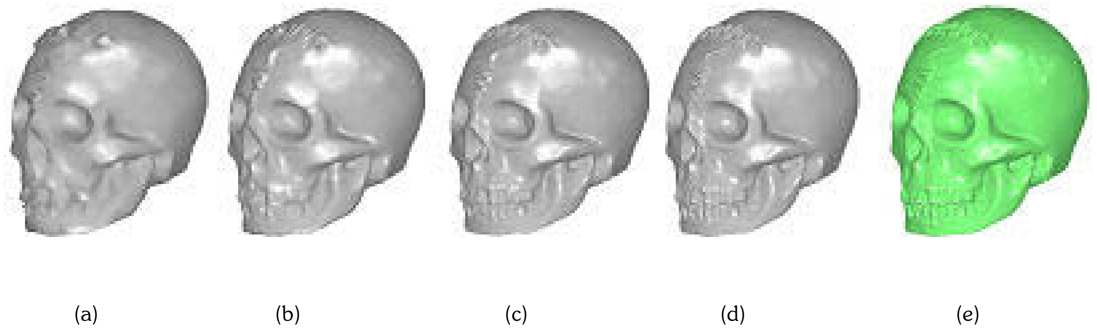

Fig. 5. Skull multiresolution meshes from our method. (a) A base triangle mesh with 2,494 triangles. (b) 1st level. (c) 2nd level. (d) 3rd and final reconstruction with 159,616 triangles (an approximation to the original Skull) with L^2 error of 0.03. (e) Original Skull mesh with 160,000 triangles with a bounding box diagonal of 162.03 units. Note that after each subdivision level the total number of triangles is 4 times that of the previous level, and the displacement value is added along each vertex's Loop limit normal.

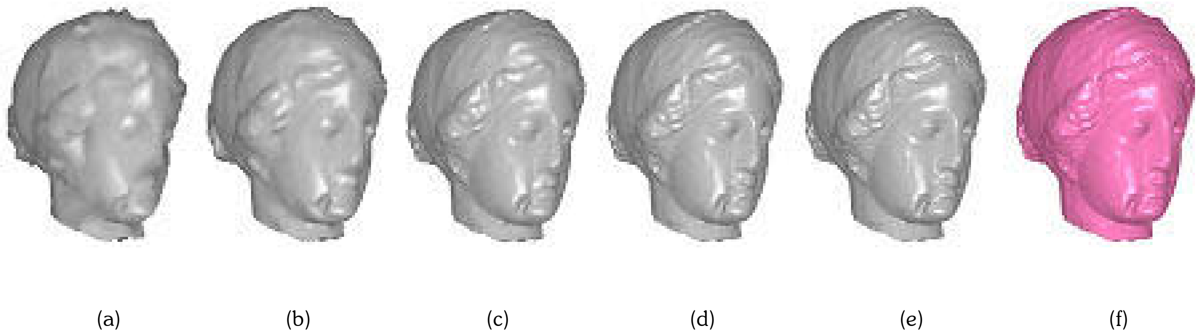

Fig. 6. Igea data multiresolution meshes from our method. (a) A base triangle mesh with 1,548 triangles. (b) 1st level. (c) 2nd level. (d) 3rd level. (e) 4th and final reconstruction with 396,288 triangles (an approximation to the original Igea) with L^2 error of 0.03. (f) Original Igea mesh with 397,312 triangles with a bounding box diagonal of 157.47 units.

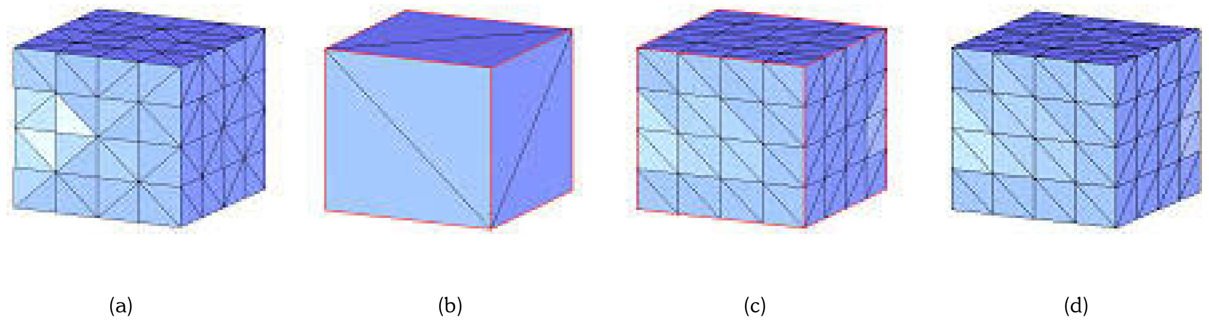

Fig. 7. Illustration of our method being applied to data with sharp features. (a) Unit cube data. (b) Simplified mesh with feature edges being tagged in red (thick lines). (c) Subdivided and displaced mesh. (d) Reconstructed mesh. In this case, (d) and (c) are the same because the displacement values are all zeros due to flat planes. That is not the case for smooth surfaces. Note that the connectivity in (d) is not the same as that in (a) because of remeshing.

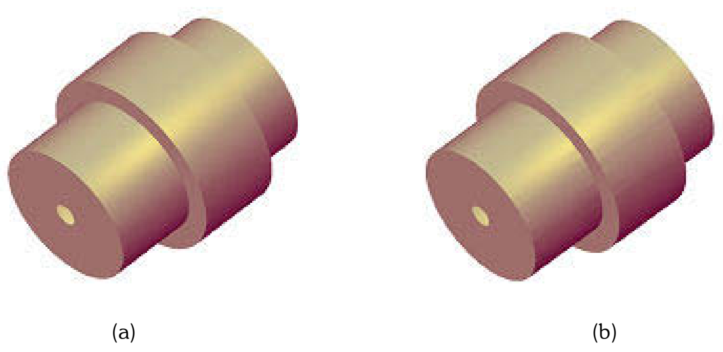

Fig. 8. Round data and its approximation. (a) Original mesh (10,560 triangles) with a bounding box diagonal of 155.72 units. (b) Approximation from our method (10,544 triangles) with L^2 error of 0.013 using a base triangle mesh of 659 triangles.

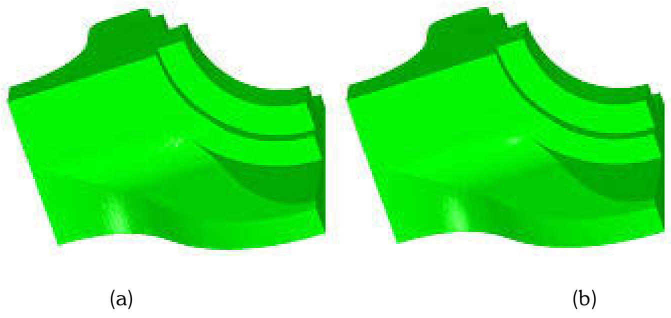

Fig. 9. Fandisk data and its approximation. (a) Original mesh (12,946 triangles) with a bounding box diagonal of 145.21 units. (b) Approximation from our method (12,944 triangles) with L^2 error of 0.0094, using a base triangle mesh of 809 triangles.

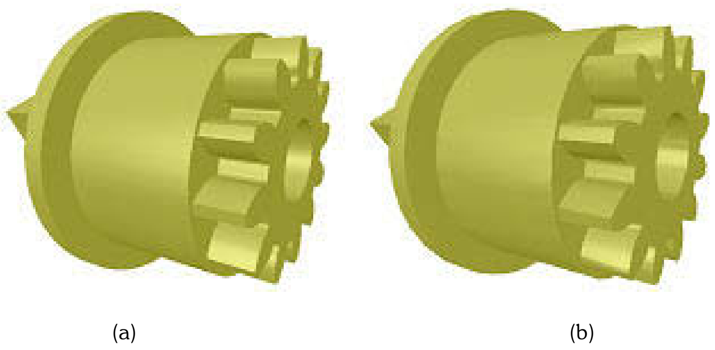

Fig. 10. Gear data and its approximation. (a) Original mesh (18,984 triangles) with a bounding box diagonal of 156.85 units. (b) Approximation from our method (18,976 triangles) with L^2 error of 0.28, using a base triangle mesh of 1,186 triangles.