International Journal of CAD/CAM Vol. 3, No. 2, pp. 97–109 (2003)

INTERNATIONAL JOURNAL OF CAD/CAM
www.ijcc.org

S-CODE: A Subdivision Based Coding System for CAD Models

Hiromasa Suzuki*, Yosuke Takarada, Shingo Takeuchi, Isao Kawano[1] and Jun Hotta[1]
Department of Precision Engineering, School of Engineering, The University of Tokyo, Tokyo, Japan
[1]Elysium Co. LTD, Shizuoka, Japan

Abstract – A large scale polygon models are often used to approximately represent 3D CAD models in Digital Engineering environment such as DMU (Digital Mockups) and network based collaborative design. However, they are not suitable for distribution on the network and for interactive rendering. We introduce a new coding system based on subdivision schemes called S-CODE system. In this system, it is possible to highly compress the model with sufficient accuracy and to view the model efficiently in a level of detail (LOD) fashion. The method is based on subdivision surface fitting by which a subdivision surface and curves which approximate a face of a CAD model are generated. We also apply a subdivision method to analytic surfaces such as conical and cylindrical surfaces. A prototype system is developed and used for evaluation with reasonably complicated data. The results show that the method is useful as a CAD data coding system.

***Keywords*: Subdivision surfaces, Surface fitting, Approximation, Data compression, Level-of-detail, Analytic surface**

1. Introduction

In advanced digital engineering environment, 3D CAD models are widely distributed to various engineering activities. One of the most typical applications of the CAD models is DMU (Digital Mockup) system in which a complete full assembly model of a product is used to evaluate various aspects of functions and manufacturability of the product. And the models are also transmitted to other companies for collaborative engineering over the network.

In those cases, the CAD models are not used in their own data format but rather they are usually converted to triangular mesh models. The reason is, first of all, that those applications do not require complete information of the CAD models, and thus CAD systems are just too expensive. Second, CAD models are classified property information and kept inside the company.

There are several *CODEC* systems to encode a CAD model to be in a specific format by which the shape information is compactly represented so as to be transmitted on the network. The encoded model is then decoded by a *viewer* which displays the CAD model. This is the same *CODEC* process with other digital media such as audio and video. Today, there are many commercial systems, XVL [8], HOOPS [15] etc.

In this paper we introduce our approach named S-CODE, which is based on approximation of CAD surface patches using *subdivision meshes*. We try to deal with CAD faces uniformly in a scheme of subdivision. Fig. 1 shows the system overview. The system consists of two parts, S-CODER and S-VIEWER. The S-CODER reads a CAD file and generates an encoded file. It first coarsely tessellates face elements of the CAD model and generates triangular meshes. These triangular meshes are used to define subdivision patches by fitting it to the CAD faces. We use two types of the subdivision meshes, *a closed patch* and *a boundary patch*. The former is mainly used to approximate parametric surfaces and spherical surfaces. The latter is used to represent cylindrical and conical surfaces. We use different subdivision methods for those two types.

These meshes are compressed using a compression algorithm to produce an S-CODE file. Then S-VIEWER reads this file and first decompresses it. Then subdivision patches are recovered and subdivided for LOD (level of detail) rendering. In the following, after introducing some of the major related works, our system is presented along the process of S-CODE. Finally, experimental results are shown.

1.1. Related works

Loop subdivision surfaces [11] are generated by repeatedly subdividing triangle meshes. The surfaces are almost of C^2 continuity except in the neighborhood of extraordinary vertices and converge to the box-spline surface. In each subdivision, every triangle of the mesh is divided into four (4-to-1 subdivision). The new positions of the vertices can be calculated by linear combination with the positions of the 1-ring vertices, and its coefficients called *subdivision mask* are shown

*Corresponding author
Tel: +81(0)-3-5841-6490
Fax: +81(0)-3-5841-8552
Homepage: http://www.cim.pe.u-tokyo.ac.jp/~suzuki/
E-mail: suzuki@cim.pe.u-tokyo.ac.jp

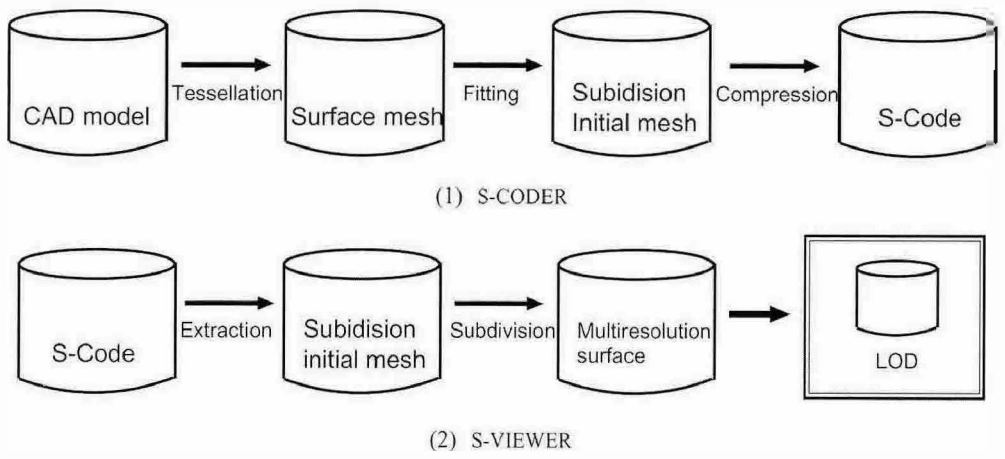


Fig. 1. Flowcharts of S-CODE system

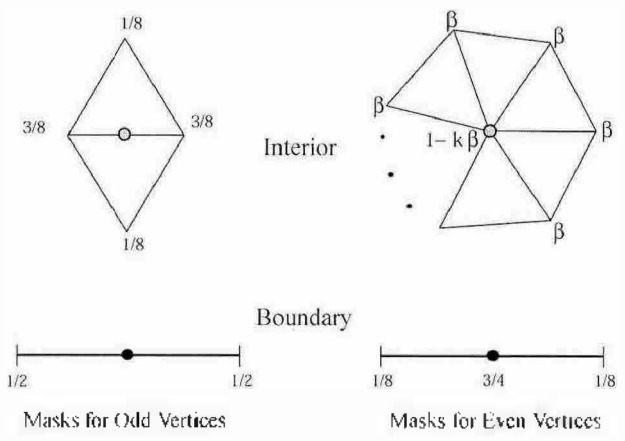


Fig. 2. Masks of Loop subdivision. Odd vertices are newly inserted vertices and even vertices are original vertices of the meshes. k means valence of a vertex and β is $\frac{3}{16}(k=3)$ and $\frac{1}{k}\left(\frac{5}{8}-\left(\frac{3}{8}+\frac{1}{4}\cos\frac{2\pi}{k}\right)^2\right)(k\neq 3)$.

in Fig. 2.

Hoppe *et al.* [5] extended Loop subdivision surfaces so as to incorporate sharp features. They introduced the type of *crease* for sharp and boundary edges, and *dart*, *corner*, *regular* and *non-regular* for crease and boundary vertices. Subdivision masks and subdivision limit position masks are defined according to the types of a vertex and its adjacent vertices. Then the piecewise smooth surfaces can be formed as C^0 continuity across the sharp features in addition to Loop subdivision surfaces. Biermann *et al.* [1] improved that scheme to deal with asymmetric boundaries and concave corners.

Subdivision surfaces can be fitted to other surfaces or a set of measured points. Halstead *et al.* [4] proposed an approximation method to adapt a subdivision surface to interpolate vertices of an input control mesh. Hoppe *et al.* [5] have generated works that deal with the subdivision surface fitting problem. They fit the piecewise smooth subdivision surface to input data points. Both Halstead and Hoppe used linear least squares in surface fitting. Suzuki *et al.* [14] proposed a simple, fast fitting method to data points with adaptive subdivision. The method is based on subdivision limit positions. Litke et al. [10] proposed *Quasi-Interpolation* fitting. Their method is also a fast fitting operation.

2. Subdivision Mesh Generation

2.1. Tessellation and Patch Classification

We assume CAD models are represented in B-Reps solid models. The first step of encoding is tessellation of a CAD model. A tessellator in a software package is used. Face elements of a CAD model are tessellated one by one while the boundaries between two patches are consistently divided. Tolerance parameter δ to control the tessellated mesh density which is shown in Fig. 3 is set to generate rather rough meshes.

Let us define some terminology used in this paper. *CAD face* denotes one face element of a B-Reps model. A CAD face can be planar, quadric or parametric and is usually trimmed. *Triangular mesh* is a tessellated CAD face which is used to generate a subdivision patch. Finally, *subdivision patch* is a mesh to be used to generate subdivided meshes (surfaces).

Each triangular mesh generated by the tessellation

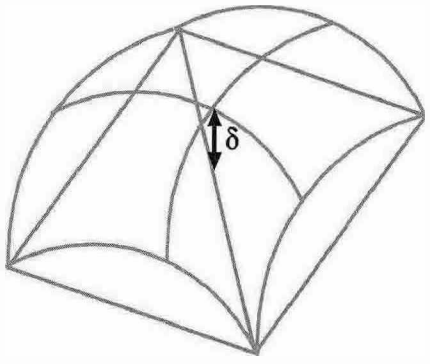


Fig. 3. Tolerance of tessellation δ.

has many triangle faces and many vertices. Some of these vertices lie on the boundary of the original CAD face and the others in its interior. It is our specific assumption that the tessellator generates only boundary vertices for cylindrical, conical and planar CAD faces, and generates both interior and boundary ones for spherical, toroidal and parametric surfaces.

Based on this assumption we classify the subdivision patches into two types:

- boundary patch: subdivision patch only with boundary vertices. This patch is generated from a CAD face of trimmed planar, cylindrical, and conical surfaces.
- closed patch: subdivision patch with both boundary and interior vertices. This patch is generated from parametric, spherical and toroidal CAD faces.

We use different types of subdivision methods for those two classes.

2.2. Closed patch subdivision

For subdividing the closed patch we use Loop subdivision surface with boundary [5]. What we need to do here is to define a Loop subdivision surface which approximates the original CAD face, or more precisely to define an initial control mesh of such a Loop subdivision surface.

For defining the initial mesh we adapt the triangular mesh generated by the tessellation. This is a process of *subdivision surface fitting*. There are several methods to this problem as introduced in Section 1.1. We use a method based on *subdivision limit positions* of a subdivision surface. By infinitely subdividing the control mesh of a subdivision surface, it finally converges to a smooth limit surface. Consider a vertex i of the control mesh whose position is denoted by v_i. Through the subdivision process this vertex converges to a position on the limit surface. This limit position of i is referred to as the subdivision limit position and denoted by v_i^∞.

As mentioned in the previous section, we first tessellate the CAD face and get a closed patch (a triangular mesh). As shown in Fig. 4, if we use this closed patch as an initial control mesh, we get a limit surface. However it will be far away from the CAD face, because the vertices of the closed mesh themselves lie on the CAD face. So we adjust the vertex positions of the closed patch so that the limit surface fits to the CAD face. Then we can use this closed patch as a control mesh for the subdivision surface approximating to the CAD face.

The subdivision limit position for a vertex is explicitly given as an affine combination of positions of its neighbor vertices;

$$v_i^\infty = \alpha v_i + \sum_{j \in i^*} \beta_j v_j \tag{1}$$

where α and β are *position mask* such that $\alpha + \Sigma\beta_j = 1$. And i^* denotes the set of vertices in the 1-ring neighborhood vertices of i. These affine coefficients are defined dependent on the vertex types. We will discuss this point later.

We adjust those vertices v_i and v_j to their new positions v_i' and v_j' respectively so that v_i^∞ becomes coincident with a position on the CAD face. By assuming this point be p_i, we get ;

$$p_i = v_i'^\infty = \alpha v_i' + \sum_{j \in i} \beta_j v_j' \tag{2}$$

Subtracting Eq. (2) from (1), we get

$$p_i - v_i^\infty = \alpha(v_i' - v_i) + \sum_{j \in i} \beta_j (v_j' - v_j) \tag{3}$$

$$\Delta v_i^\infty = \alpha \Delta v_i + \sum_{j \in i} \beta_j \Delta v_j \tag{4}$$

where $\Delta v_i^\infty = p_i - v_i$ and $\Delta v_i = v_i' - v_i$. Since this equation holds at every vertex i, we set up a simultaneous set of equations;

$$\Delta V^\infty = \mathbf{A} \Delta V \tag{5}$$

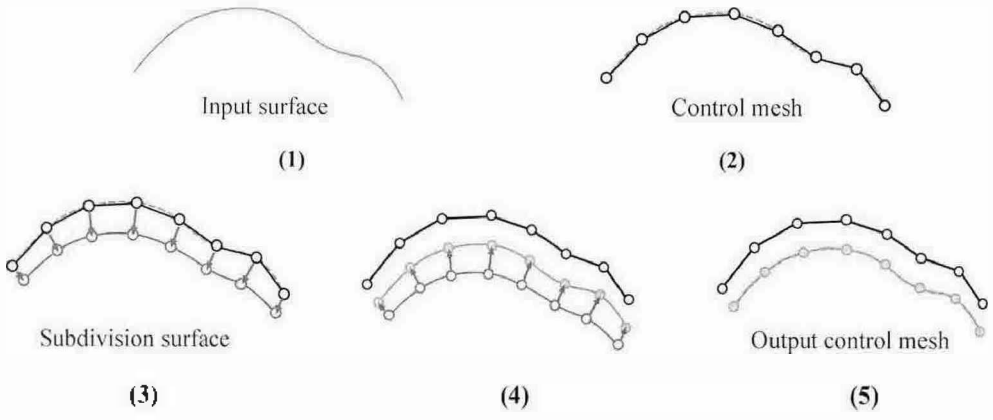


Fig. 4. Subdivision surface fitting using subdivision limit positions.

where ΔV^{∞} is a vector whose i-th element is v_i^{∞} and ΔV is also a vector whose i-th element is $\Delta v_i = v_i^{\infty} - v_i$.

The matrix **A** is defined by the coefficients of the limit position in Eq. (4). The size of matrix **A** is $n \times n$ where n is the number of vertices of the closed patch. Though the size of **A** can be large, the matrix **A** is sparse enough to be efficiently solved. New position V is obtained by;

$$V' = V + \Delta V \tag{6}$$

where V and V' are vectors of v_i and v_i' respectively.

Since the closed patch has a boundary, the fitting process consists of two steps. First we fit the boundary vertices using their subdivision limit positions, then fit interior vertices using their subdivision limit positions. As for the target position p_i, it can be v_i itself, since the v_i is of the triangular mesh and therefore it is on the CAD face.

After fitting we evaluate the geometrical fitting error. And if it is greater than a given threshold, the patch is uniformly subdivided according to the Loop subdivision scheme and fitted again. In this case the target positions of the newly inserted vertices are the closest position on the surface.

2.2. Position masks

In order to apply the above fitting method, we need to know the coefficients for computing the subdivision limit positions, namely, *position masks*. As we use the Loop subdivision surface with boundary introduced by Hoppe *et al.* [5], we use their position masks. In their paper, they classified vertices depending on the number of crease edges incident to the vertex. A crease edge represents a sharp feature where the surface is not smooth. When a vertex is incident to 0 (no crease), 1, 2 and more than 2 crease edges, it is called *smooth*, *dart*, *crease* and *corner*, respectively. Hoppe derived position masks for each of those types as well as subdivision masks.

In our case, we only have smooth vertices in the interior of the closed patch since we assume a CAD face is smooth. So the position mask for the smooth vertex is used, which is the same with the Loop subdivision surface.

As for the vertices on the boundary (i.e. vertices on an edge of a CAD face), Hoppe suggested to treat the boundary as crease edges and presented subdivision masks and position masks. However, we found the set of their position masks are not complete and that there remain some technical discussions. Though it is not problematic for their applications, we derived position masks for our purpose.

Let us quote their subdivision masks first. Let us focus on a boundary vertex c whose neighboring vertices on the boundary are r and l (Fig. 5). By one step of subdivision, a new vertex is inserted to an edge. For instance, r^1 is inserted to edge cr and its positions v_{r^1} is defined by an affine combination of v_c and v_r:

$$v_{r^1} = (1-\eta)v_c + \eta v_r \tag{7}$$

The coefficients $(1-\eta, \eta)$ is called *edge mask* determined by the vertex types of the end vertices c and r. In the same manner, a vertex l^1 is inserted to edge cl and its position is defined by

$$v_{l^1} = (1-\eta')v_c + \eta' v_l \tag{8}$$

where η and η' are dependent on the types of end vertices of the edge.

By the subdivision, the position of c is also changed to its new position v_{c^1}.

$$v_{c^1} = (1-\alpha-\beta)v_c + \alpha v_r + \beta v_l \tag{9}$$

where coefficients α and β are again defined according to the vertex types of c, r and l. The coefficients $(\alpha, 1-\alpha-\beta, \beta)$ is called *vertex mask*.

Now consider the position mask. The limit position of v_c is given by the following equation;

$$v_c^{\infty} = (1-\delta-\gamma)v_c + \delta v_r + \gamma v_l \tag{10}$$

The coefficient $(\delta, 1-\delta-\gamma, \gamma)$ is called *position mask*.

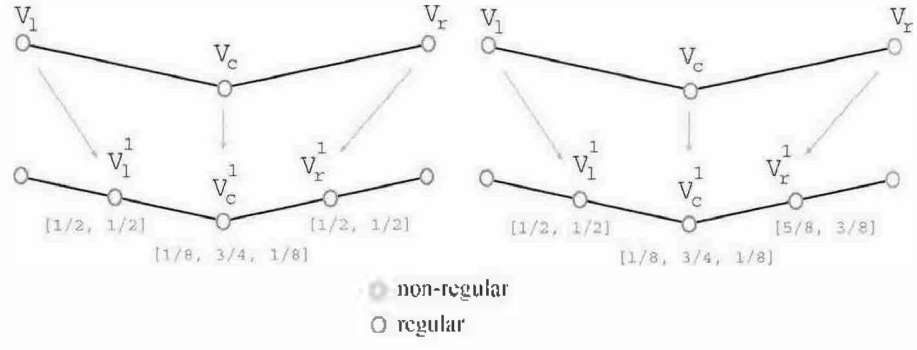


Fig. 5. Calculation of position masks for regular and non-regular vertices.

In order to derive these coefficients from the subdivision masks introduced above, we apply one time of subdivision. The subdivision limit position for c^1 is denoted by $v_{c^1}^{\infty}$. Assuming the position is dependent on v_{c^1} and its neighboring boundary vertices v_{r^1} and v_{l^1}, we get the following equation:

$$v_{c^1}^{\infty} = (1-\delta-\gamma)v_{c^1} + \delta v_{r^1} + \gamma v_{l^1} \quad (11)$$

The theory of subdivision surface tells that the limit position of $v_{c^1}^{\infty}$ agrees with v_c^{∞} in Eq. (10). And v_{c^1}, v_{r^1} and v_{l^1} are given by Eqs. (7), (8) and (9). By solving all of those equations, we can find a position mask of $(\delta, 1-\delta-\gamma, \gamma)$.

The above derivation of the position masks for a boundary vertex suggests that they are dependent on the vertex types of c, r and l. Each vertex can be of either one of three types, so there are 3^3 cases including redundant symmetrical ones. Fortunately it is not necessary to analyze all these cases.

Vertices on the boundary can be classified into three classes. A corner vertex is, in our case, corresponding to a vertex element of B-Reps model, and whose position is fixed through the subdivision (as is the case with Hoppe's corner vertex). Otherwise, the boundary vertices are either regular or non-regular. A regular vertex is one that has exactly four incident vertices and a non-regular vertex does not.

Let us first look at the corner vertex. Since it does not move, the position mask is independent of the incident vertices and has the form of (0, 1, 0). As for the regular and non-regular vertices, referring to the subdivision masks derived in [5], it is known that in the edge subdivision masks, corner and non-regular vertices are identical. So we do not have to distinguish those two types and incident corner vertex can be regarded as non-regular vertex.

For the boundary, note that the numbers of the vertices incident to a boundary vertex can be different on the two sides of the boundary. As indicated by Biermann *et al.* [1], if the vertex is regular on one side and non-regular on the other side, the boundary curves can not be consistent. We regard such a vertex to be non-regular if its either side is non-regular.

In summary, we need to compute position masks for those 6 cases as shown in Table 1.

Table 1. Position masks for boundary vertices

Adjacent Vertex	Vertex	Adjacent Vertex
regular	regular	regular
$\frac{1}{6}$	$\frac{4}{6}$	$\frac{1}{6}$
regular	regular	non-regular
$\frac{1}{6}$	$\frac{11}{16}$	$\frac{7}{48}$
non-regular	regular	non-regular
$\frac{7}{48}$	$\frac{17}{24}$	$\frac{7}{48}$
regular	non-regular	regular
$\frac{1}{5}$	$\frac{3}{5}$	$\frac{1}{5}$
regular	non-regular	non-regular
$\frac{1}{5}$	$\frac{5}{8}$	$\frac{7}{40}$
non-regular	non-regular	non-regular
$\frac{7}{40}$	$\frac{13}{20}$	$\frac{7}{40}$

2.5. Boundary patch subdivision

A boundary patch is a triangular mesh with no interior vertices and defined by perimeter curves. It is generated by tessellating trimmed planar, cylindrical and conical patches. As is the case with boundary curves for the closed patch, we use curve subdivision based on crease edge subdivision [5]. We also use the fitting method to obtain the subdivision curve which approximates the edge curve of the CAD patch. It will be discussed in Section 4 how the subdivision mesh is generated.

3. Compression and Decompression

By the above approximation processes, we obtain subdivision patches in the form of triangular meshes. We apply a compression algorithm for triangular meshes. We currently use Edgebreaker algorithm [13], since its software is available. It compresses topological data of a mesh, but does not compress geometrical data. Nevertheless we find that a triangular mesh, in general, compression is not so useful in our case, since our control meshes are not so dense. We also use zlib compression [3] in addition to Edgebreaker.

Edgebreaker basically works for a manifold mesh of genus zero. Therefore, when applying Edgebreaker, we insert dummy vertices into the mesh for filling holes and making the genus zero. After decompression, we need to remove dummy vertices.

The output file of S-CODE contains the following information, each of which is compressed by zlib.

patch attribute: a patch type of closed patch or boundary patch and the dummy vertex information.

axis vector or position of zenith: If the patch surface is cylinder or cone, our subdivision scheme needs the axis vector and position of the zenith, respectively. It will be discussed later.

corner vertex: a flag to indicate a vertex corresponding to a vertex element of a CAD face.

compressed mesh

4. LOD Rendering

The other half of the system is S-VIEWER which loads the S-CODE file. After decompression, a subdivision meshes are recovered. Using these subdivision meshes S-VIEWER displays the object in a LOD fashion. We apply a simple LOD method as follows;

1. fix the max detail level n
2. clipping each triangle of initial subdivision mesh (level 0).
3. project the triangle and determine its screen size.
4. determine subdivision level m based on the size.
5. display triangles at the level m.

We use different subdivision methods for the closed patch and the boundary patch.

4.1. LOD rendering of closed patch

A closed patch is subdivided according to the 4-to-1 Loop subdivision scheme. In order to achieve good performance, a data structure is important. The efficiency can be improved by utilizing the regularity of 4-to-1 subdivision in the data structure. There are several related works such as Vertex Quad Tree [6], NCA (Nearest Common Ancestor), Linear Quad-Tree [9] and Pulli-Segal method [12].

In the execution of subdivision, it is a bottleneck to acquire the neighbors, such as the neighbor vertices incident to a vertex and the neighbor triangles incident to a triangle. In case of the quad tree, heavy pointer addressing to traverse the tree is needed. For using the linear quad tree a data structure like a hash table is needed, though the search can be done in constant time by using tricky bit operations. Pulli-Segal method is very efficient but requires to match two triangles.

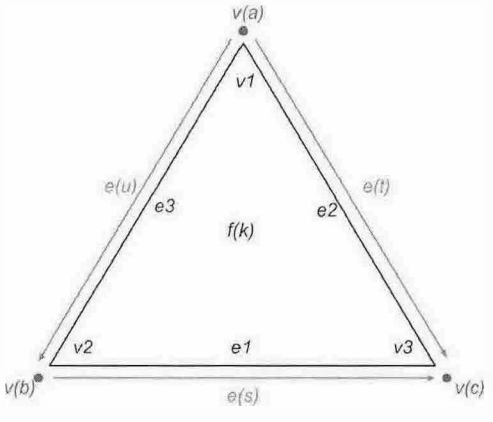
We developed an index numbering rules for vertex/edge/face id's based on the relationship among the numbers of vertex/edge/face elements, #V, #E and #F, before and after one step of subdivision: #V′=#V+#E, #E′=2#E+3#F and #F′4#E where #X′ denotes the number of these elements after subdivision. These rules are similar to but not equivalent to one proposed by Eberly [2].

Fig. 6 shows our index numbering rules. Let's take the triangle face for instance. The face which has index k is subdivided into four triangles. The indices for these four triangles are given by 4k, 4k+1, 4k+2 and 4k+3. The vertex indices and edge indices are also defined in a similar manner. Using this numbering scheme, we can represent a mesh simply in arrays which grow monotonically as the subdivision proceeds.

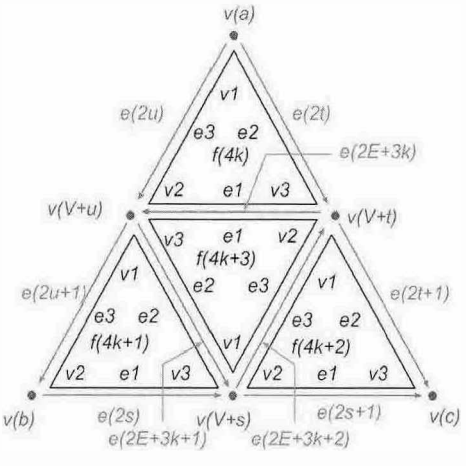
In our LOD method, we first subdivide the mesh at the given maximum times n to generate the densest mesh. Then we retrieve meshes at an intermediate level m using the index numbering scheme. Indices for triangles at the m-th level mesh for an initial triangle face k, D[k] [$\cdots$] can be generated from the highest level mesh S[$\cdots$] by a simple procedure of nested loops given below.

```
foreach (i1; 0≤i1≤4) {
  ⋮
  foreach (im; 0≤im<4) {
    foreach ( j; 0≤j<3) {
      a=((4i1+i2)×4+⋯)×4+im; \
      b=((((4k+i1)×4+i2)×4+⋯+im)
                    ×4+j)×4+⋯)×4+j;
      D[k] [a].v(j)=S[b].v(j);
    }
  }
  ⋮
}
```

Although memory footprint of our method is not less than other approaches, the mesh of the middle level is



(a) i^{th} subdivision level



(b) $(i+1)^{th}$ subdivision level

Fig. 6. Index numbering rules.

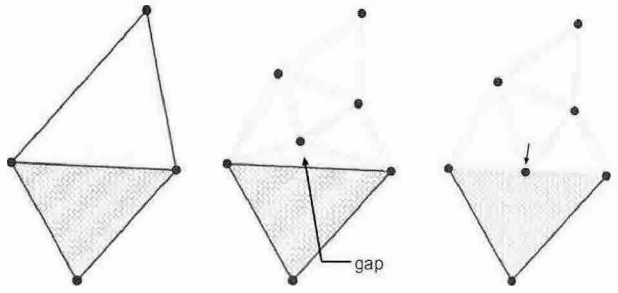


Fig. 7. Filling gap by moving vertices at m-th level to the edge of the n-th level mesh ($m > n$).

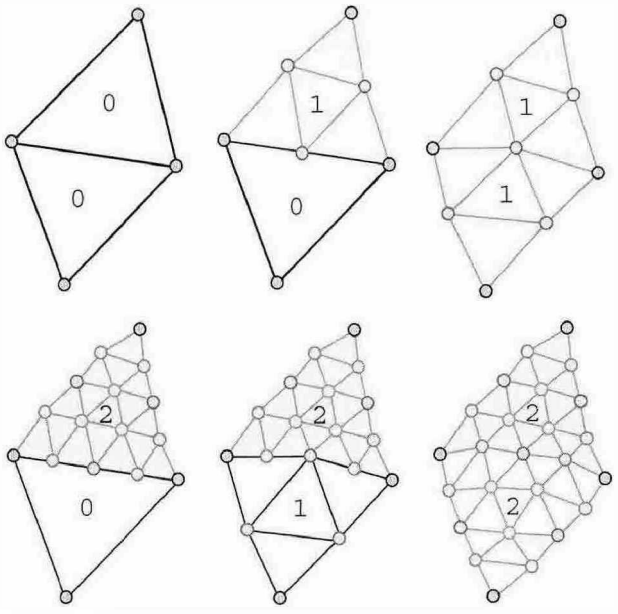


Fig. 8. Filling gap between meshes at different levels.

easily generated by thinning out the vertices in the maximum level mesh in LOD rendering as shown by the above algorithm.

Since the subdivision is made for each triangle of the initial subdivision mesh, gap appears between the meshes. We should fill the gap by adjusting the vertices of higher level as shown in Fig. 7. We pre-compute the adjusted positions at different levels by moving the vertices at n-th level onto the edge of n'-th level ($n > n'$). In the data structure, boundary vertices which are first generated by m-th subdivision have $m+1$ possible positions (Fig. 8). This method can support interactive LOD rendering with semi-adaptive refinement of the initial control mesh.

There are other methods for filling the gap generated by non-uniform subdivision schemes. They fill the gap by temporally replacing the triangle of lower level with a triangle fan [7, 16]. But they assume the difference in the subdivision level between patches is usually one or statistically fixed thus they do not allow the level difference to dynamically change.

Fig. 9 shows the result of our subdivision scheme and LOD rendering for a closed patch.

4.2. LOD rendering of boundary patch

We first tried to apply the same Loop subdivision scheme also to the boundary patches. Its problem is that the generated interior vertices are not necessarily on the original CAD face, and particularly silhouette lines are distorted. We prefer subdivision which subdivides only the boundary and does not introduce interior vertices as shown in Fig. 10.

We classify the triangles according to the number of edges on the boundary. We name 0-face, 1-face, 2-face and 3-face according to the number. We assume the patch has no 3-face. Then 1-face and 2-face are subdivided as shown in Fig. 11, and 0-face is not subdivided. The positions are determined by using Hoppe's crease edge rules as described above.

This works well for trimmed planar patch and a conical patch containing zenith (the apex of a cone). However, the cylindrical and conical patches without the zenith are still distorted because the inserted edges are not being along the axis and the numbers of edges incident to a vertex are not uniform (Fig. 12). And the distortion becomes worse for trimmed faces as shown in Fig. 13.

Since those surfaces are important in mechanical

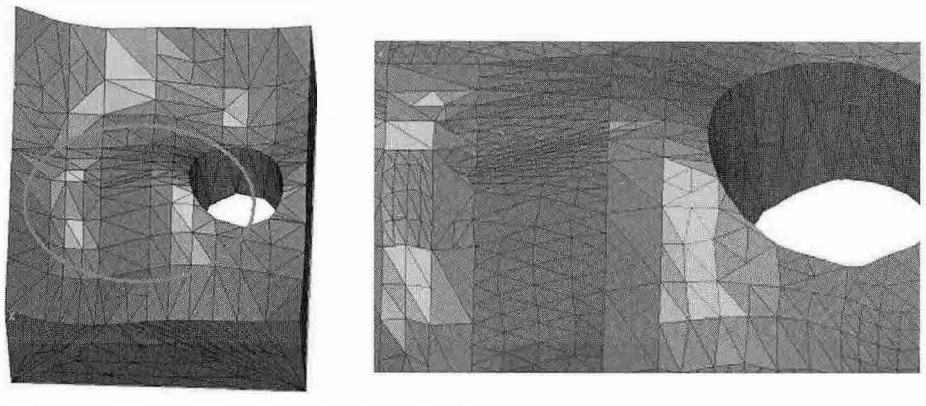


Fig. 9. Subdivision and LOD rendering of closed patch (right: enlarged).

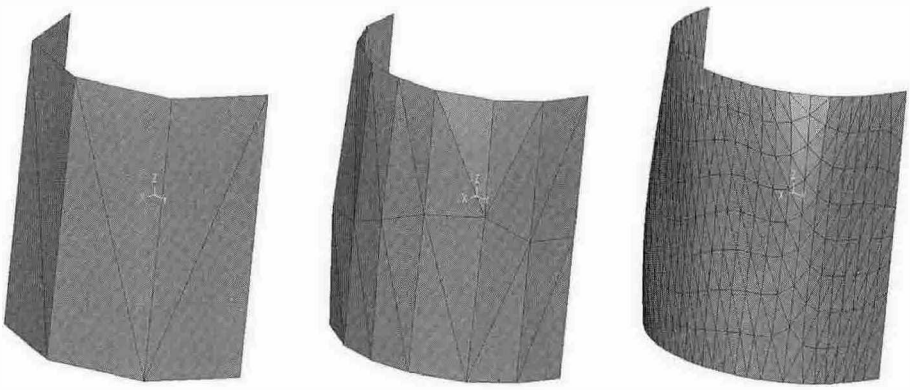


Fig. 10. Distortion in the interior of cylindrical patch generated by 4-to-1 subdivision.

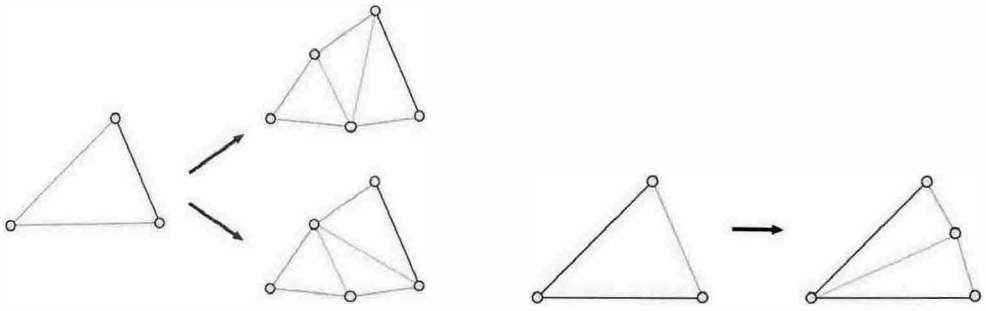


Fig. 11. Subdivision of 1-face (left) and 2-face (right).

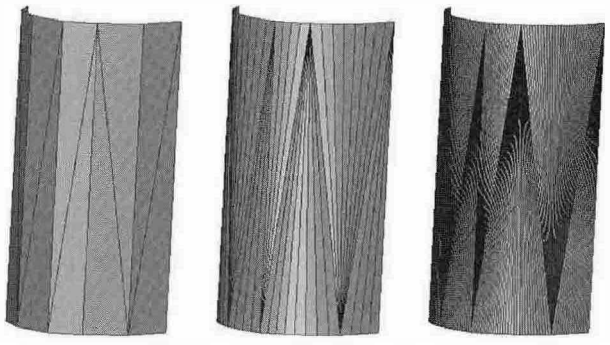


Fig. 12. Distortion in the interior of cylindrical patch generated by 4-to-1 subdivision.

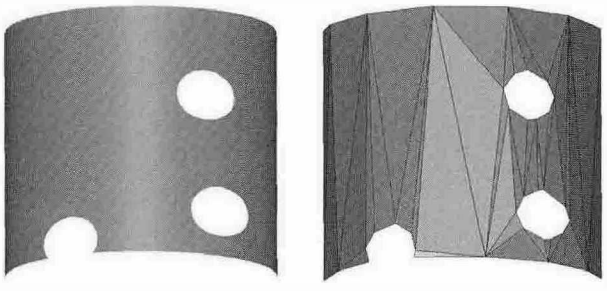


Fig. 13. Trimmed cylindrical surface patch and its tessellation.

design, we introduce a special treatment for such cylindrical and conical patches. From Fig. 13 it is noticed that such a triangular mesh consists of triangle strips (a series of triangles) which are connected at 0-faces each other. In other words, a boundary patch could be decomposed into a set of triangle strips connected through 0-faces.

Furthermore, a triangle strip is basically defined by

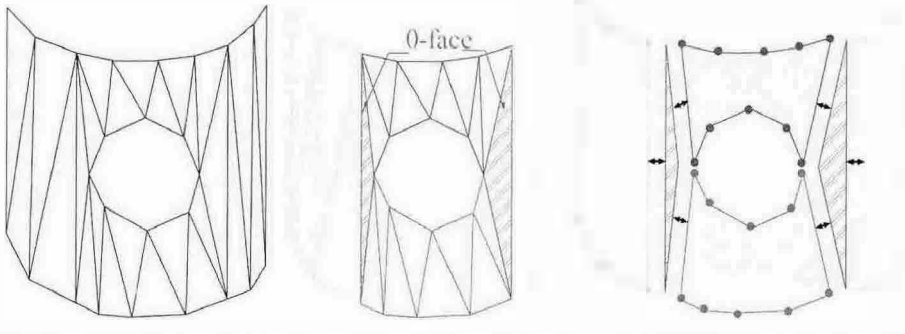


Fig. 14. Boundary patch with Polygonal Belt and 0-face. |left| Boundary patch. |center| Segmentation. |right| Polygonal belts.

two side curves on a boundary of the patch which are represented by a subdivision curve. So we call such a strip *polygonal belt*. We segment the boundary patch into a set of polygonal belts connected via 0-faces. Then we subdivide the polygonal belt by subdividing the two side subdivision curves and generate a strip by inserting edges between the side curves. Those inserted edges are aligned to be along the central axis as much as possible. So we need extra information of axis direction of the original surface.

We have not examined this method fully and there is no guarantee that this method is applicable to all the trimmed patterns of a surface. However, when it is applicable, it can generate desirable results.

5. Results

5.1. Examples

We first applied our method to the model Easy1 (Fig. 15). The model consists of 38 patches, all of which have attributes such as plane, cone, cylinder, sphere, but no parametric surfaces. In this figure, (1) shows the input CAD model. (2) and (3) are the initial subdivision meshes for different tessellation tolerances. The coarser initial control mesh (2) has 512 vertices and 548 triangles and the denser one (3) has 1048 vertices and 1144 triangles. (4) and (5) show the subdivision surfaces with efficient LOD rendering for (2) while (6) and (7) show those for (3). Especially, (5) and (7) show good performance of the subdivision scheme and LOD rendering we proposed for trimmed cylindrical surface.

Our method was applied also to the model Pump (Fig. 16). The model consists of 133 patches whose attributes are both analytic and parametric surfaces. In Fig. 16, (1) shows the input CAD model. (2) is the subdivision mesh. (3) and (4) show the LOD rendering. As shown in this figure, subdivision and LOD rendering at the trimmed and filet regions are made successfully.

As more complicated examples, we applied our method to the model Tractor (Fig. 17) and the model Sports Car (Fig. 18). These models are very large scaled which consist of 64 bodies, 5032 patches in

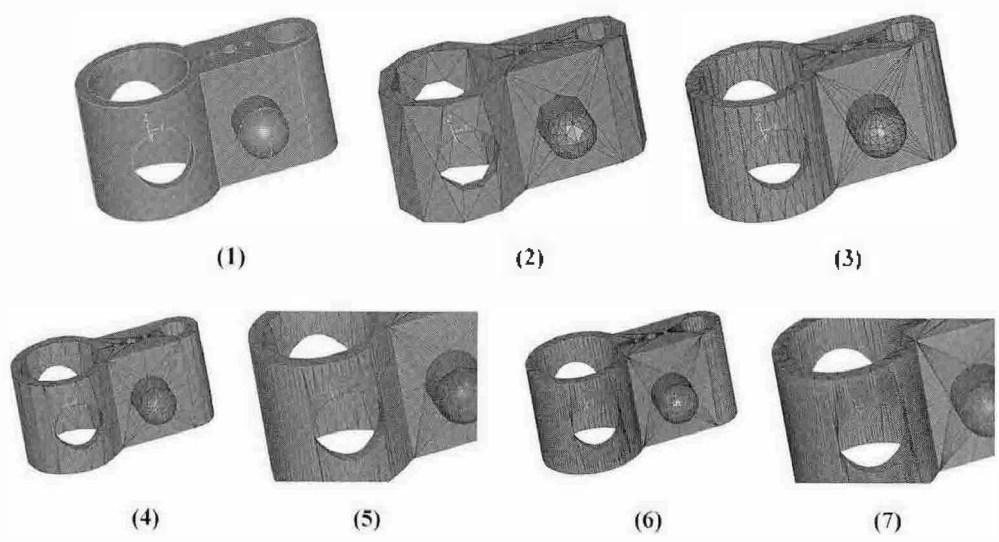


Fig. 15. Easy1: (1) input CAD model. (2)(3) initial control meshes at different coarseness generated through tessellation and subdivision surface fitting steps. (4)(5) LOD rendering of the subdivision surfaces for (2). Similarly, (6) and (7) show LOD rendering of (3).

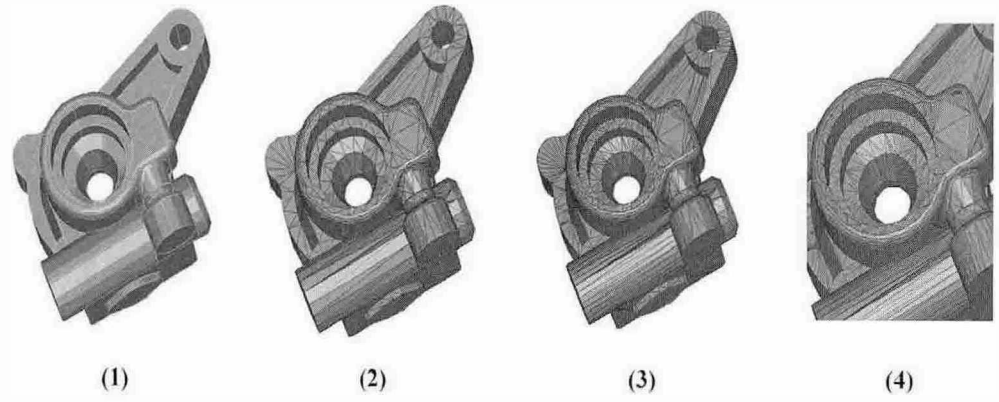


Fig. 16. Pump: (1) input CAD model. (2) initial control meshes after fitting. (3) and (4) the subdivision surfaces with LOD rendering. Subdivision surface fitting using subdivision limit positions.

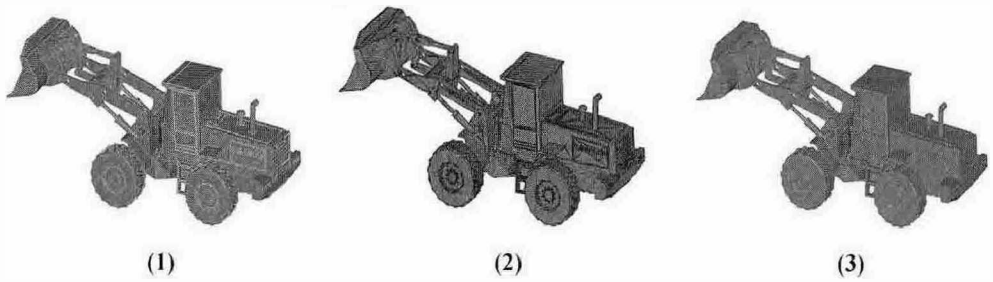


Fig. 17. Tractor: (1) input CAD model. (2) and (3) subdivision surfaces of the control mesh with LOD rendering (edge view). Flowchart of S-CODE system.

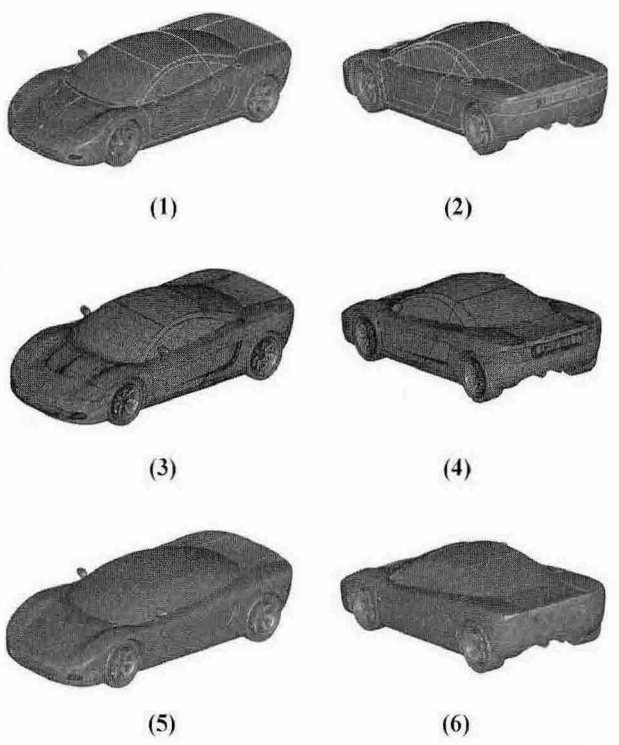


Fig. 18. Sports Car: (1) and (2) the input CAD model (viewing from another angle). (3) - (6) the subdivision surfaces of the control meshes with LOD rendering.

Tractor, and 134 bodies, 875 patches in Sports Car. As for Tractor, most patches are classified into analytic surfaces. In Fig. 17, (1) is the input CAD model, and (2) and (3) show the subdivision meshes with LOD rendering.

On the other hand, most CAD faces in Sports Car are free-form surfaces. In Fig. 18, (1) and (2) are the input CAD model (viewing from another angle). Figs. from (3) to (6) show the subdivision surfaces of the control mesh with LOD rendering. These two examples show that our subdivision scheme and LOD rendering are successful for large scaled objects.

In our method, both the size of S-CODE file and the surface approximation accuracy mostly depend on the coarseness of tessellation. There is a trade-off between those two, so it is quite difficult to decide automatically the optimal coarseness of tessellation and the maximum

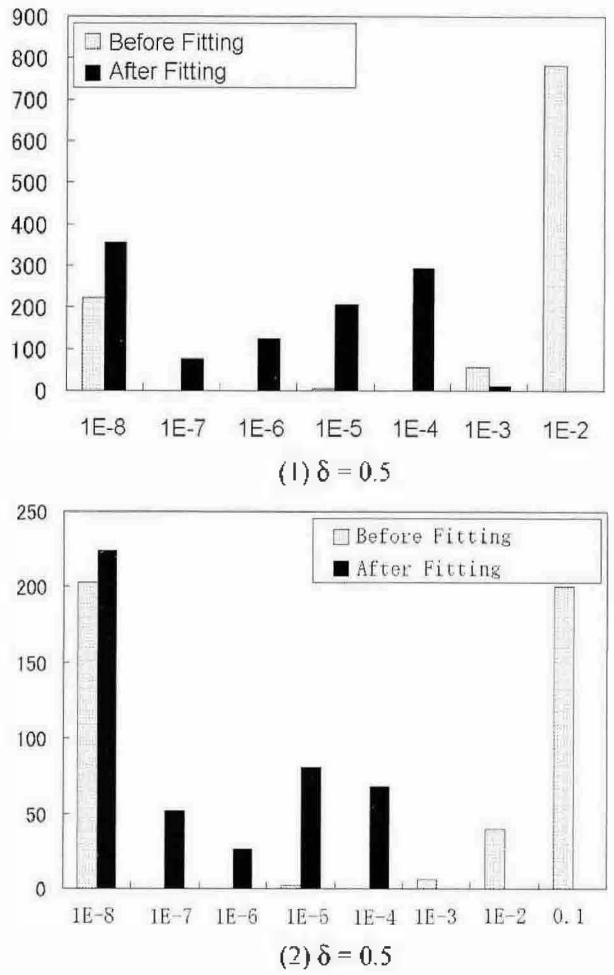
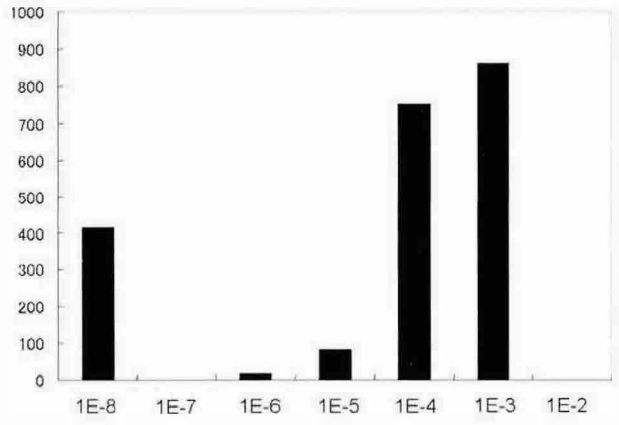


Fig. 19. Histograms of error distances.

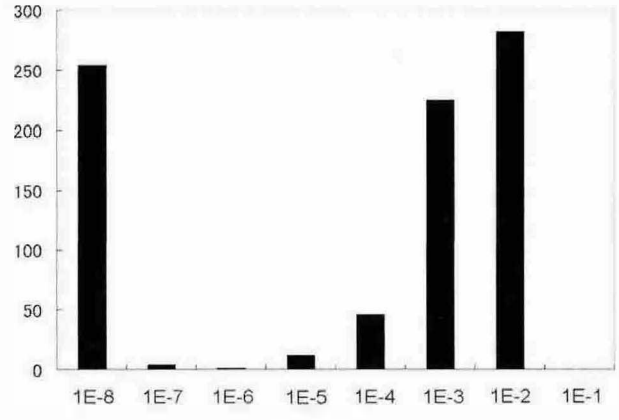
subdivision level.

5.2. Surface approximation accuracy

For measuring the surface approximation accuracy, we compute the gap between the input parametric CAD face P and approximating subdivision surfaces S. Specifically, the error distance is computed between subdivision limit position of vertices associated with the control mesh of subdivision surfaces and those nearest points on P. In order to evaluate the fitting performance, we compare the accuracy of the subdivision meshes Q not fitted to P and that of the subdivision



(1) $\delta = 0.5$



(2) $\delta = 0.5$

Fig. 20. Histogram of Error distances of the meshes subdivided once.

meshes R fitted to P.

For the case of Easy1 (Fig. 15) CAD faces are tessellated into Q with δ=0.5(see Fig. 3), Fig. 19 (1) shows the error distance for Q and R. Also, when the CAD faces are tessellated into R with δ=5.0 for the same model, the error distance for the meshes without fitting and a mesh with fitting are shown in Fig. 19 (2).

These two tables show the following two things:

- By the fitting procedure, subdivision limit positions exactly on the subdivision surfaces converge to the input CAD face.
- The larger the facet tolerance, which is given as the approximation distance (δ), the larger error the surface exhibits.

As mentioned above, we can measure the error between P and subdivision surfaces by computing the distance between subdivision limit positions and those nearest points on P. However, this approach just evaluates the error in narrow portions of subdivision surfaces near subdivision limit positions of the control vertices of the initial subdivision mesh. Therefore, we subdivide the control mesh to evaluate wider portions of subdivision surfaces. Fig. 20 shows the results using the two tessellation tolerances.

From these graphs, we find subdivision limit positions of vertices newly inserted by the subdivision scheme do not approach to P very well by comparison to those of the original vertices. The reason is that the fitting procedure makes use of only subdivision limit positions of vertices of the initial subdivision mesh. However, the error distance for newly inserted vertices can still be regarded as sufficiently accurate approximation for viewing the model.

As for a boundary patch, our subdivision scheme is based on the idea that analytic surfaces such as cylinder and cone can be represented by only perimeter curves. This subdivision scheme subdivides only boundary edges and never generates inner vertices. Moreover, in order to generate better shaped surfaces, the direction of inner edges is aligned along the axis. So the approximation accuracy is determined by those of vertices on the boundary, which is similar to the closed patches. However, due to this alignment our subdivision can no longer be uniform, and has lost the regularity of 4-to-1 subdivision. Therefore, this subdivision scheme is not as fast as our 4-to-1 subdivision scheme. On the other hand, the size of generated subdivision meshes is smaller than that of 4-to-1 subdivision.

5.3. Compression rates

For encoding and decoding system, how much the system can compress the data is one of the most important factors. Therefore, we evaluate compression ratio of the input CAD file (IGES file) to the S-CODE file and we also made comparison with STL file for comparison, which is generated by tessellating the CAD model with δ=0.1. This comparison must not be so fair, but may give the readers some idea about the compression rates. Table 2 shows the comparison with the various models as shown in Figs. 15 - 18. Note that

Table 2. The rate of compression with each model

model(δ)	CAD data	STL	S-CODE	Comp. rate
Easy1 (0.5)	130 KB	337 KB	9.53 KB	7.33%
Easy1 (5.0)			4.33 KB	3.33%
Pump (0.5)	757 KB	3031 KB	65 KB	8.58%
Tractor (1.0)	13,899 KB	85,746 KB	474 KB	3.4%
Sports Car (1.0)	11,221 KB	73,414 KB	256 KB	2.3%

we apply the compression method to each subdivision patch. Though the boundaries between patches can be duplicated, this overhead is small.

6. Conclusions

In this paper, S-CODE system is proposed as an application of subdivision meshes for approximating a CAD model and its LOD rendering. The approximated meshes are compressed to an S-CODE file whose size is very small compared to the original CAD data file. We deal with CAD faces basically by two types of subdivision methods, the closed patch and the boundary patch. Both of them are generated by subdivision surface fitting method. The fitting accuracy is sufficiently good for visualization purposes. In particular, we introduced a new numbering scheme for indexing mesh elements. Thanks to this scheme, we can retrieve subdivision meshes very efficiently. And in order to avoid the distortions which are visible for conical and cylindrical surfaces, strip based subdivision method is also introduced. The system is applied to several CAD models to demonstrate its performance.

Considering future research, since approximation accuracy is not always sufficiently good, we possibly use other approximation methods such as quasi-interpolation [10]. The idea of the polygonal belt and its subdivision needs more analysis to be completed. It will be useful to extend S-CODE to be able to decode STL based mesh models which are generated from CAD models. For this purpose we need a robust method to recover topological structure of the mesh and to segment it to a reasonable set of face elements.

Acknowledgements

This S-CODE Project was funded by IPA (Information Processing Association), Japan in 2001.

References

[1] Biermann, H. *et al.* (2000), "Piecewise smooth subdivision surfaces with normal control," SIGGRAPH 2000 Conference Proceedings, 113-120.

[2] Eberly, D. H. (2000), "3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics," Morgan Kaufmann Pub.

[3] Gailly, J.-l. and Adler, M., zlib. http://www.gzip.org/zlib/

[4] Halstead, M. *et al.* (1993), "Efficient fair interpolation using Catmull-Clark surfaces," SIGGRAPH 93 Conference Proceedings, 35-43.

[5] Hoppe, H. *et al.* (1994), "Piecewise smooth surface reconstruction," SIGGRAPH 94 Conference Proceedings, 295-302.

[6] Jenkins, S. (2001), "Implementing subdivision and multiresolution surfaces," Game Developer's Conference Course Note.

[7] Kobbelt, L. (2000), $\sqrt{3}$-subdivision, SIGGRAPH 2000 Conference Proceedings, 103-112.

[8] Lattice Technology Inc. (2003), XVL, http://www.lattice.co.jp.

[9] Lee, M. and Samet, H. (2000), "Navigating through triangular meshes implemented as linear quadtrees," *ACM Trans. Graphics*, **19**(2), 79-121.

[10] Litke, N. *et al.* (2001), "Fitting subdivision surfaces," IEEE Visualization 2001 Conference Proceedings, 319-324.

[11] Loop, C. (1987), "Smooth subdivision surfaces based on triangles," Master's thesis, Department of Mathematics, University of Utah.

[12] T. Pulli, K. and Segal, M. (1996), Fast rendering of subdivision surfaces, Proc. 7th Eurographics Workshop on Rendering, 61-70.

[13] Rossignac, J. (1999), "Edgebreaker: Connectivity compression for triangle meshes," *IEEE Trans. on Visualization and Computer Graphics*, 5(1), 47-61.

[14] Suzuki, H. *et al.* (1999), Subdivision surface fitting to a range of points, Pacific Graphics 99 Conference Proceedings, 158-167.

[15] Tech Soft America, HOOPS, http://www.hoops3d.com/

[16] Zorin, D., Schroeder P. and Sweldens, W. (1997), Interactive multiresolution mesh editing, SIGGRAPH 97 Conference Proceedings, 259-268.

Hiromasa Suzuki is a professor of RCAST (Research Center for Advanced Science and Technologies) at The University of Tokyo. His research interests include geometric modeling and and its applications to mechanical CAD/CAM systems. Suzuki received his doctor degree in precision machinery engineering from the University of Tokyo in 1986. He is a member of JSPE (Japan Society for Precision Engineers), JSME(Japan Society for Mechanical Engineers), IEEE and ACM.

Yosuke Takarada is a software engineer at CANON Inc. Japan. He received his master degree in precision machinery engineering from the University of Tokyo, Japan in 2003.

Shingo Takeuchi is a Ph.D pre-candidate of Mechanical Engineering at the University of Michigan. His research interests included geometric modeling and computer graphics. Currently, his research is related to disassembly sequence planning and design optimization. Takeuchi received his master degree in precision machinery engineering from the University of Tokyo in 2001.

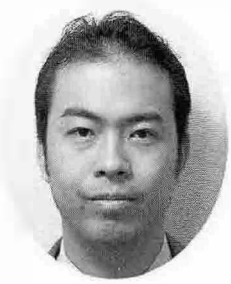
Isao Kawano is an engineer of ELYSIUM CO., LTD. that is an independent venture company that plans and develops various kinds of package software products by means of highly specialized techniques for 3D geometry handling and data translation.

Jun Hotta is a CEO of Zodiac Co., Ltd. His research interests include 3D geometric data and its applications to engineering communication. Hotta graduated from the Hiroshima University faculty of technology in 1981.

Hiromasa Suzuki

Yosuke Takarada

Shingo Takeuchi



Isao Kawano

Jun Hotta