**Proceedings of DETC'01**
**ASME 2001 Design Engineering Technical Conferences**
**Pittsburgh, Pennsylvania, September 9-12 2001**

**DETC2001/CIE-21305**

# SKETCH BASED MESH EXTRUSION WITH REMESHING TECHNIQUES

Charlie C.L.Wang Matthew M.F.Yuen

Department of Mechanical Engineering, The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
wangcl@ust.hk, meymf@ust.hk

## ABSTRACT

In this paper, we proposed a useful 3D mesh extrusion method for intuitive, efficient geometric modeling of free-form polygonal models. With our method, the user can sketch two strokes to extrude a polygonal mesh surface. Two remeshing techniques: partial mesh re-triangulation and mesh optimization are described in this paper first. After that, the extrusion algorithm with the remeshing techniques is introduced in detail. The method can be widely used in the modeling of free-form polygonal objects. And at the end of this paper, several examples are given.

**KEYWORDS:** Mesh extrusion, Mesh optimisation, Mesh surface, Sketch input, Geometric modeling.

## 1. INTRODUCTION

The goal of this research is to develop an efficient and intuitive tool for freeform geometric modeling. In current commercial geometric modeling softwares, there are different kinds of modeling tools. But all these tools are for precise design. They are not efficient or intuitive enough for conceptual design. It is important to develop an efficient and intuitive tool for conceptual design. Studies show that most designers nowadays still prefer to express their creative design idea through 2D sketches. Thus, the input method of this modeling tool is also very important.

The mesh extrusion method is such a modeling tool for conceptual design. It is based on a two-stroke input by users, using two remeshing techniques, to rapidly construct an extruded mesh surface. Our method uses the polygonal mesh to represent a surface, which is widely used for geometric modeling in general.

In this paper, we first introduce the two remeshing techniques that are used as the fundamental mesh processing algorithms of mesh extrusion. Then, the whole procedure of the mesh extrusion approach is presented in detail.

## 2. LITERATURE REVIEW

A lot of research has been focused on the mesh surface construction. Meyers et al.'s work (1992) was concerned with the problem of reconstructing the surfaces of three-dimensional objects, given a collection of planar contours representing cross-sections through the objects. Hoppe et al. (1994) presented a general method for automatic reconstruction of accurate, concise, piecewise smooth surface models from scattered range data. SKETCH system (Zeleznik et al., 1996) rapidly constructs approximate shape via direct mark based interaction. Teddy system (Igarashi et al., 1999) constructed a rounded freeform mesh model by finding the chordal axis of user input 2D closed stroke to build smooth surface around the axis. Another approach to mesh construction was the use of implicit surfaces (Bloomenthal and Wyvill, 1990; Markosian et al., 1999). The user specified the skeleton of the intended model and the system constructs smooth, natural-looking surfaces around it. By extending implicit surface modeling techniques, Markosian et al. (1999) presented a particle-based freeform surface modeling approach. In their system, a user interactively guides the particles, called skin, to grow over a given collection of polyhedral elements (or skeletons), yielding a smooth surface (though subdivision) that approximates the underlying skeletal shapes. In our mesh construction algorithm, we simply use the approach of Mayers et al. to construct mesh surface between two neighbouring base rings.

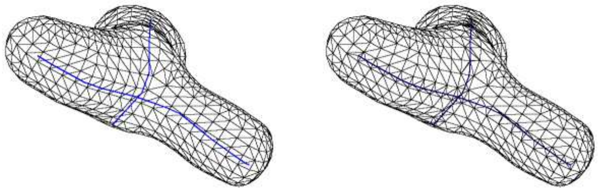
Suzuki et al. (2000) presented a 3D mesh-dragging method for intuitive, efficient geometric modeling of free-form polygonal models. With their method, the user can drag a part of a triangular mesh and change its position and orientation. Their method is based on an adaptive remeshing procedure. Similarly, our interactive mesh extrusion also proposes an intuitive, efficient geometric modeling of free-form polygonal models. Our approach is also based on two fundamental remeshing techniques: partial mesh re-triangulation and mesh optimization. Some other interactive modeling researches are related to the multi-resolution presentation of models: Zorin et al (1997) built a scalable interactive multi-resolution editing system based on mesh refinement and coarsification algorithms; and Khodakovsky and Schroder (1999) developed an algorithm based on Zorin's approach that can modify the fine level shape of a surface.

Several algorithms have been formulated to simplify surfaces and to solve the more general problem of multi-resolution modeling. Schroeder et al. (1992) described an algorithm that we term vertex decimation. Their method iteratively selects a vertex for removal, removes all adjacent faces and re-triangulates the resulting hole. Garland and Heckbert (1997) developed a surface simplification algorithm that can rapidly produce high quality approximations of polygonal models. The algorithm uses iterative contractions of vertex pairs to simplify models and maintain surface error approximations using quadric matrices. It can facilitate good approximations, both visually and with respect to geometric error. But in their algorithm, the element shape qualities are not respected. Hoppe et al. (1993) presented a method for solving the mesh optimization problem by defining an energy function that explicitly models the competing requirements of conciseness of representation and fidelity to the data. Their algorithm is based on obtaining the optimal solution of energy functions. It is a time consuming iteration scheme, and reference cloudy points are required. Frey and Borouchaki (1998) presented a surface mesh optimization method that is suitable for obtaining a geometric finite element mesh, given an initial arbitrary surface triangulation. The first step of their method involves constructing a $G^1$ continuous geometric support, associated with the initial surface triangulation, which represents an adequate approximation of the underlying surface geometry. The initial triangulation is then optimized with respect to this geometry as well as the element shape. Their procedure of mesh optimization is also a very time consuming iteration scheme, but not a real time mesh optimization technique. We address the problem of optimizing an arbitrary triangular mesh surface with respect to a prescribed criteria to obtain a unit surface mesh (i.e., where all edges have a normalized length close to the optimal value). Our algorithm proposes to optimize the mesh in "real time".

Our extrusion approach is closely related to the extrusion method in Teddy system (Igarashi et al., 1999). But we focus more on the fundamental techniques. Based on the two fundamental techniques: partial mesh re-triangulation and mesh optimization, we develop an efficient and intuitive freeform modeling tool – mesh extrusion.

## 3. PARTIAL MESH RE-TRIANGULATION

The work is to re-triangulate the triangles of a 3D polygonal object – $\Phi$ by painted curves on its surface. After re-triangulation, the painted curves become the edges of the triangles of $\Phi$ (as shown in Figure 1).



(a) before re-triangulation (b) after re-triangulation

Figure 1 Partial Mesh Re-triangulation

The whole procedure of re-triangulation consists of four steps: 1) calculate the intersections of the painted curves; 2) create new vertices and edges by painted curves; 3) triangulate triangles in $\Delta_{i,i=1,2,\ldots,n}$ one by one; 4) use the triangulation result to obtain the new object – $\Phi'$.

After we have painted several curves on the surface of $\Phi$, the painted curves are represented by several 3D line segments. But the intersections of painted curves are not stored in the data structure of our system. Thus, we need to obtain the intersections and insert them into the painted curves as nodes in the first step of subdivision.

In the second step of re-triangulation, we use the nodes of the painted curves to create new vertices of mesh object $\Phi$; and use the line segments of the painted curves to create new edges. Triangular edges crossed by the painted curves are divided into several edges. These new vertices and new edges should be stored in the data structure of $\Phi$ as they will be used in the third step of subdivision. The divided triangular edges are removed from $\Phi$.

The third step of re-triangulation subdivides the triangles in $\Delta_{i,i=1,2,\ldots,n}$ one by one; the procedure is shown in Figure 2 (from the left column to the middle column). Because edges and vertices have been created in step two, only new triangular faces are created in this step. Here the constrained Delaunay triangulation algorithm is performed. At the end of this step, we obtained new triangles $\Delta'_{i,i=1,2,\ldots,n}$ .

Step four is shown in Figure 2 (from the middle column to the right column). After replacing the triangles $\Delta_{i,i=1,2,\ldots,n}$ by $\Delta'_{i,i=1,2,\ldots,n}$ in $\Phi$, we obtain a new mesh object $\Phi'$. In $\Phi'$, the painted curves on $\Phi$ have become the edges of the triangular mesh.

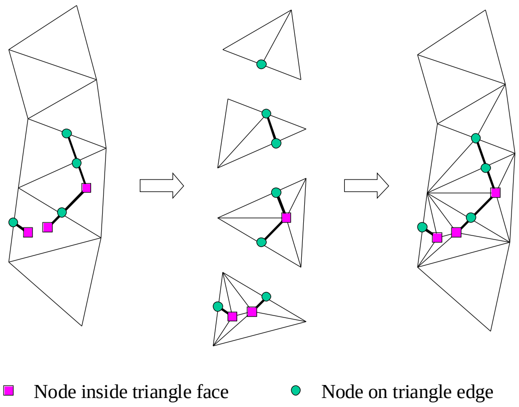
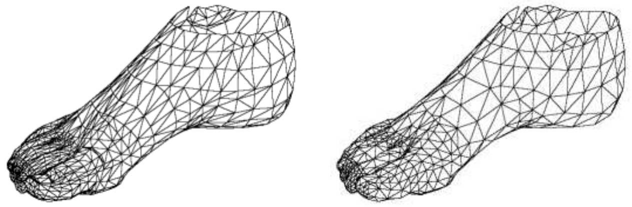


Figure 2 Re-triangulate $\Delta_{i,i=1,2,\ldots,n}$ to obtain $\Phi'$

## 4. MESH OPTIMIZATION

In this section, we propose to address the problem of optimizing an arbitrary triangular mesh surface with respect to the prescribed criteria to obtain a unit surface mesh (i.e., where all edges have a normalized length close to the optimal value, see Figure 3). Our algorithm proposes to optimize the mesh in real time.



(a) before optimization (b) optimization result

Figure 3 Example of Mesh Optimization

### 4.1 Criteria of Optimization

Before defining the criteria of optimization, we need a parameter to measure the shape of triangles in the given mesh surface, $\mathrm{M}$. Since our mesh optimization algorithm is an edge based optimization method, we define a shape factor function, $h(e)$, for any edge $e$ in $\mathrm{M}$.

**Definition 1** Shape factor function $h(e)$ is

$$h(e) = \frac{6 l_e}{\sum l_i}$$

where $l_e$ is the length of edge $e$, and $l_i$ is the edge length of triangles shared edge $e$.

Let $\eta_3$ be the (discrete) size map associated with the vertices of $\mathrm{M}$. Frey and Borouchaki (1998) define a continuous size map $\eta_3^c$ in $\Re^3$ by interpolating $\eta_3$ over $\mathrm{M}$.

**Definition 2** Mesh $\mathrm{M}$ conforms to the map $\eta_3$ if and only if

$$\forall e \in \mathrm{M}, \quad \frac{1}{\sqrt{2}} \le l_e \le \sqrt{2}$$

where $l_e$ is the length of edge $e$ with respect to $\eta_3$.

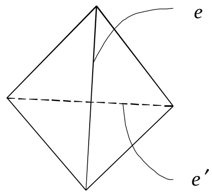


Figure 4 Dual edge of $e$ in $\mathrm{M}$

For any edge $e$ in $\mathrm{M}$, we define a dual edge, $e'$, as shown in Figure 4. The definition of the dual edge is used to describe the criteria for edge swapping. Berg et al. (1997) introduced the angle-optimal of a triangulation of the planar point set, which is derived from Thales's Theorem. In the same way, we define the angle-optimal of $\mathrm{M}$.

**Definition 3** We assume that mesh $\mathrm{M}$ has $n$ triangles. Consider the $3n$ angles of the triangles of $\mathrm{M}$, sorted by increasing value. Let $\alpha_1, \alpha_2, \ldots, \alpha_{3n}$ be the resulting sequence of the angles; hence, $\alpha_i \le \alpha_j$, for $i < j$. We call $A(\mathrm{M}) := (\alpha_1, \alpha_2, \ldots, \alpha_{3n})$ the angle-vector of $\mathrm{M}$. Let $\mathrm{M}'$ be a mesh surface that has the same vertices with $\mathrm{M}$ and let $A(\mathrm{M}') := (\alpha'_1, \alpha'_2, \ldots, \alpha'_{3n})$ be its angle-vector. We say that the angle-vector of $\mathrm{M}$ is larger than the angle-vector of $\mathrm{M}'$ if $A(\mathrm{M})$ is lexicographically larger than $A(\mathrm{M}')$, or, in other words, if there exists an index $i$ with $1 \le i \le 3n$ such that: $\alpha_j = \alpha'_j$ for all $j < i$, and $\alpha_i > \alpha'_i$. We denote this as $A(\mathrm{M}) > A(\mathrm{M}')$. A mesh surface, $\mathrm{M}$, is called angle-optimal if $A(\mathrm{M}) \ge A(\mathrm{M}')$ for all mesh surfaces, $\mathrm{M}'$, have the same vertices with $\mathrm{M}$.

**Definition 4** Define the minimum angle function $\alpha(e)$ of edge $e$,

$$\alpha(e)=\min\{\angle\alpha\in f_{j,j=1,2}\}$$

where $f_{j,j=1,2}$ are the triangles sharing edge $e$, and $\angle\alpha$ is the angle in $f_{j,j=1,2}$.

From Definition 1, 2, 3 and 4, we can define the criteria of mesh optimization as $\forall e\in \mathrm{M}$,

$$\frac{1}{\sqrt{2}}\le h(e)\le\sqrt{2} \qquad (1)$$

$$\alpha(e)<\alpha(e') \qquad (2)$$

Equation (1) is the criterion for edge split and edge collapse, Equation (2) is the criterion for edge swap. After the criteria is reached, we obtained a unit surface mesh – in which all edges have a normalized length closed to one. But after testing, we find that criteria (1) is too restrict to implement. Thus, we change it to $\frac{1}{2}\le h(e)\le\sqrt{2}$.

### 4.2 Optimization Operators

The purpose of mesh optimization is to modify the initial triangular mesh so as to produce a unit surface mesh in accordance to the criteria. Basically, it involves subdividing the longest edges, collapsing the shortest edges (the edge length is computed with respect to the criteria of optimization) and swapping some edges to reach angle-optimal. Practically, this stage involves edge split; edge collapse and edge swap operation (Figure 5).

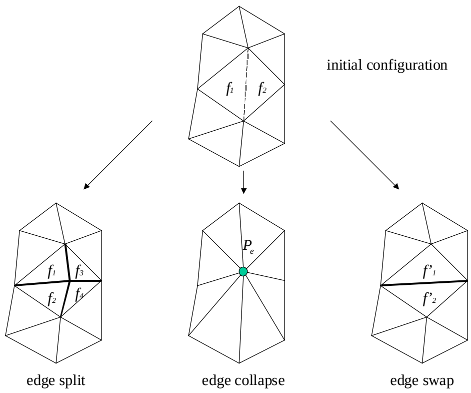


Figure 5 Optimization Operators

**Edge split:** For a mesh edge $e$, the edge split operation involves introducing the edge midpoint $P_m$ and in snapping $P_m$ onto a local spherical surface, $S_s$, which is determined by endpoints of edge $e$ and the endpoints of its dual edge $e'$ (introduced in section 4.3 in detail). The two triangles, $f_1$ and $f_2$, sharing edge $e$ are replaced by four new triangles, $f_{j,j=1,\ldots,4}$. One new vertex and four new edges are created. And all topological information of entities sharing edge $e$ and its endpoints should be alternated too.

**Edge collapse:** The edge collapse operation is based on the identification of the two endpoints of edge $e$, thus leading to a unique point $P_e$ that can be either one of the original edge endpoints. It also changes the position of $P_e$ to the interpolation point $P_m'$ of the midpoint of edge $e$ on local spherical surface $S_s$ (the determination of $P_m'$ is introduced in section 4.3). This operation removes the two triangles $f_1$ and $f_2$, their sharing edge $e$, and the two edges sharing the deleted vertex $P_d$ in $f_1$ and $f_2$; replaces the endpoint of the edges sharing $P_d$ to $P_e$.

**Edge swap:** The objective of the edge swap operation is to decide whether the current configuration is better (regarding the element shape – in detail criteria (2) introduced in last section) than the alternate configuration (Figure 5, right-hand side). The two triangles ($f_1$ and $f_2$) sharing edge $e$ are replaced by two new triangles ($f_1'$ and $f_2'$) sharing the dual edge $e'$ of $e$. All topological information of entities sharing endpoints of edge $e$ and endpoints of $e'$ should be alternated.

### 4.3 Geometric Interpolation

For the edge split operator, the coordinate of the newly created vertex needs to be determined; for the edge collapse operator, the new coordinate of either of the endpoints of edge $e$ should also be determined. As we mentioned in the last section, we use a local spherical surface, $S_s$, to interpolate those points.

**Local Spherical Surface** $S_s$**:** Let $P_1(x_1,y_1,z_1)$ and $P_2(x_2,y_2,z_2)$ be the two endpoints of edge $e$; and let $P_3(x_3,y_3,z_3)$ and $P_4(x_4,y_4,z_4)$ be the two endpoints of edge $e'$, a local spherical surface, $S_s$**,** is determined (Figure 6).

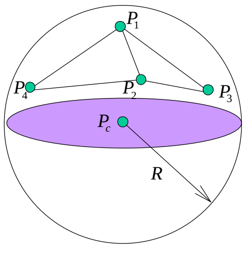


Figure 6 Local Spherical Surface $S_s$ Determined by Endpoints of Edge $e$ and $e'$

The equation of $S_s$ is

$$\begin{vmatrix} x^2+y^2+z^2 & x & y & z & 1 \\ x_1^2+y_1^2+z_1^2 & x_1 & y_1 & z_1 & 1 \\ x_2^2+y_2^2+z_2^2 & x_2 & y_2 & z_2 & 1 \\ x_3^2+y_3^2+z_3^2 & x_3 & y_3 & z_3 & 1 \\ x_4^2+y_4^2+z_4^2 & x_4 & y_4 & z_4 & 1 \end{vmatrix} = 0\,. \tag{3}$$

**Interpolate Point on $S_s$:** The midpoint, $P_m$, of edge $e$ is $P_m = (P_1 + P_2)/2$. Let the unit normal of points $P_1$ be $n_1$ and the unit normal of $P_2$ be $n_2$. We approximate the normal of $P_m$ as $n_m = (n_1 + n_2)/2$. Thus, a line equation, which passes through $P_m$ with direction $n_m$ is obtained by

$$P(t) = P_m + tn_m\,. \tag{4}$$

Equations (3) and (4) can determine the two intersections. The closest intersection point to $P_m$, named $P'_m$, is the interpolation point on $S_s$. In edge split, we snap point $P_m$ to the position of $P'_m$; and in edge collapse, we change the position of $P_e$ to the interpolation point $P'_m$.

### 4.4 Scheme of Mesh Optimization

In order to select an edge to perform edge collapse and edge split during a given iteration, we need to assign them some notations. The shape factor function $h(e)$ of edge $e$ mentioned in definition 1 of section 4.1 defines the cost. And to the edge swap operator, the notation of cost is defined by the minimum angle function $\alpha(e)/\alpha(e')$ of edge $e$ mentioned in definition 4 of section 4.1.

**Algorithm Summary:** Our optimization algorithm is build around the edge operators described in section 4.2, and criteria defined in section 4.1. The current implementation represents models that use an adjacency graph structure, which includes vertices, edges, and faces; they are all explicitly represented and linked together. Each vertex maintains a list of the edges of which it is a member. The algorithm itself can be summarized as follows:

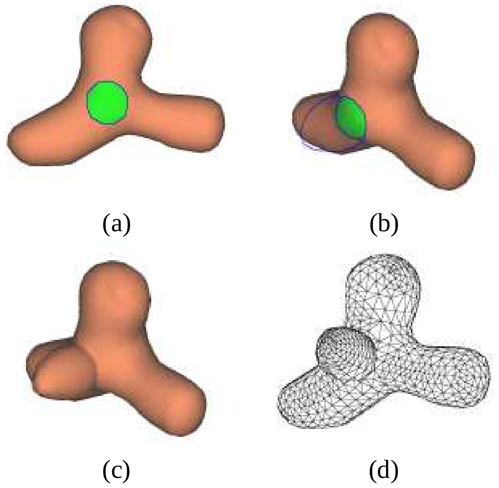
1. Compute the shape factor function $h(e)$, and the minimum angle function $\alpha(e)$ of each edge $e$;
2. Place all edges in a maximum heap $H_s$ keyed on the $h(e)$ of edge – the edge with maximum $h(e)$ is at the top of $H_s$;
3. Iteratively split the edge $e$ at the top of $H_s$; remove the edge $e$ and insert new edges created by the edge split operator. When $h(e)$ of the edge at the top of $H_s$ reaches $h(e) \le \sqrt{2}$, the iteration stops.
4. Place all edges in a minimum heap $H_c$ keyed on the $h(e)$ of edge – the edge with minimum $h(e)$ is at the top of $H_c$;
5. Place all edges in a minimum heap $H_a$ keyed on the $\alpha(e)/\alpha(e')$ of edge – the edge with minimum $\alpha(e)/\alpha(e')$ is at the top of $H_a$;
6. Remove the edge $e$ at the top of $H_c$, collapse edge $e$; then remove it from $H_a$ and update the costs of all valid edges involving $e$ in $H_c$ and $H_a$;
7. Swap the edge $e$ at the top of $H_a$, update the cost of $e$ in $H_c$; update the costs of all valid edges involving $e$ in $H_c$ and $H_a$;
8. If $h(e)$ of the edge at top of $H_c$ fails to reach $h(e) \ge \frac{1}{2}$; or if $\alpha(e)/\alpha(e')$ of the edge at top of $H_a$ fails to reach $\frac{\alpha(e)}{\alpha(e')} \ge 1$, return to (6).

**Preventing Mesh Inversion:** Edge operators do not preserve the orientation of the faces in the area of the edge operations necessarily. For instance, it is possible to collapse an edge or swap an edge and cause some neighboring faces to fold over on each other. It is usually best to avoid this type of mesh inversion. We use essentially the same scheme used by others before (Garland, 1997). When considering a possible edge collapse or swap, we compare the normal of each neighboring face before and after the operation. If the normal flips, this operation would be either heavily penalized or disallowed.

**Control Gap Tolerance:** The gap between the approximated mesh surface and the "true surface" is an important tolerance. It must be controlled during the process of mesh optimization. Let $D(P_m)$ be the distance from the midpoint $P_m$ of edge $e$ to the local spherical surface $S_s$**;** and $D(P'_m)$ be the distance from the midpoint $P'_m$ of edge $e$ to $S_s$. While considering a possible edge swap, we compare $D(P_m)$ and $D(P'_m)$. If $D(P_m) > \alpha\, D(P'_m)$, edge swap operation should be disallowed (where $\alpha$ is a coefficient to control the accuracy of approximation, i.e. $\alpha = \sqrt{2}$ ).

## 5. MESH EXTRUSION

The extrusion is a two-stroke operation: a closed stroke on the surface of object $\Phi$ and a 2D stroke depicting the silhouette of the extruded surface. Firstly the user draws a closed stroke on the object surface (Figure 7a); then the user rotates the model to bring the closed stroke sideways and draw a silhouette line to extrude the surface (Figure 7b). A sweep operation is applied on $\Phi$ to construct the 3D shape by moving the closed surface line along the skeleton of the silhouette. Figure 7c shows the result of the extrusion, and Figure 7d shows the mesh representation of the extrusion result. The direction of the extrusion is perpendicular to the object surface, but not parallel to the screen.


(a) (b)

(c) (d)

Figure 7 Extrusion

The algorithm of extrusion consists of four steps: 1) Create base ring; 2) Project extruding stroke; 3) Sweep base ring; 4) Sew adjacent rings.

**5.1 Create base ring**

The base ring $\mathrm{B}$ is defined as a closed curve, which lies on the surface of object $\Phi$, $\mathrm{B} = \{e_i, e_i \in \Phi\}$, where $e_i$ are the connected triangles' edges. In the algorithm of the extrusion, it should be created by the closed stroke input of users first.

When users draw a closed stroke on the object surface, the system calls the partial re-triangulation algorithm to re-triangulate the object by the closed stroke. After the re-triangulation, the closed stroke has become the triangle edges of $\Phi$. These triangle edges are $e_i$ in $\mathrm{B}$, they form the base ring $\mathrm{B}$. The area inside $\mathrm{B}$ is highlighted in Figure 7c.

**5.2 Project extruding stroke**

As shown in Figure 8, the 2D extruding stroke $\Omega$ is projected onto a plane to obtain the projected 3D extruding stroke $\Omega_p$. Thus, the major problem of this section is how to find the projection plane.

Two constrains are defined to determine the projection plane,

**Constrain 1** The plane passes through two points $P_1^*(x_1, y_1, z_1)$ and $P_2^*(x_2, y_2, z_2)$, where $P_i^* \in \mathrm{B}\ (i = 1,2)$ are the closest points to the two endpoints of $\Omega$.

**Constrain 2** The plane lies parallel to the normal $\vec{n}_{\mathrm{B}} = l_{\mathrm{B}}\vec{i} + m_{\mathrm{B}}\vec{j} + n_{\mathrm{B}}\vec{k}$ of the base ring $\mathrm{B}$.

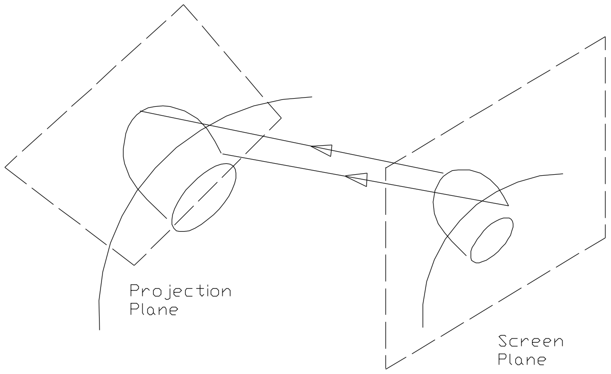


Figure 8 Project Extruding Stroke

Under the above two constraints, the plane faces towards the camera as much as possible. If the normal of each vertex on $\mathrm{B}$ is $\vec{n}_i = l_i\vec{i} + m_i\vec{j} + n_i\vec{k}$, the normal of $\mathrm{B}$ can be obtained by

$$\vec{n}_{\mathrm{B}} = (\frac{1}{n}\sum_{i=1}^{n} l_i)\vec{i} + (\frac{1}{n}\sum_{i=1}^{n} m_i)\vec{j} + (\frac{1}{n}\sum_{i=1}^{n} n_i)\vec{k} \tag{5}$$

where $n$ is the number of vertices on $\mathrm{B}$. And then, the equation of the projection plane can be written as

$$\begin{vmatrix} x & y & z & 1 \\ x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ l_{\mathrm{B}} & m_{\mathrm{B}} & n_{\mathrm{B}} & 0 \end{vmatrix} = 0. \tag{6}$$

**5.3 Sweep base ring**

After the system projects the 2D extruding stroke onto the plane, producing a 3D extruding stroke. Copies of the base ring are created along the extruding stroke in such a way as to be almost perpendicular to the direction of the extrusion, and are resized to fit within the stroke (Figure 9). This is done by advancing two points (left and right) along the extruding stroke starting from both ends (Figure 10).

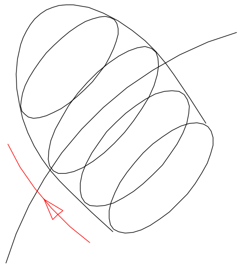


Figure 9 Sweep Base Ring

And in each step, the system chooses the best of the following three possibilities: advance the left point – case one, the right point – case two, or both – case three. The goodness value increases when the angle between the line connecting the points and the direction of the stroke at each pointer is close to 90 degrees. In other words,

$$\alpha_1 = \max\{|90 - \angle P_5P_3P_1|, |90 - \angle P_4P_2P_3|\},$$

$$\alpha_2 = \max\{|90 - \angle P_3P_1P_4|, |90 - \angle P_6P_4P_1|\},$$

$$\alpha_3 = \max\{|90 - \angle P_5P_3P_4|, |90 - \angle P_6P_4P_3|\},$$

where $\alpha_1$, $\alpha_2$, and $\alpha_3$ respectively represent the angle of case one, two, and three. Thus, the possibility of the three cases – $P_i^*$ is

$$P_i^* \propto \frac{1}{\alpha_i}, \quad i = 1,2,3. \tag{7}$$

This process is completed when the left and right points meet.

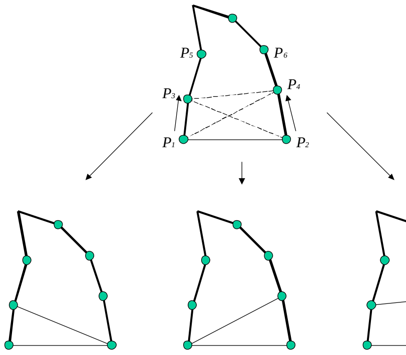


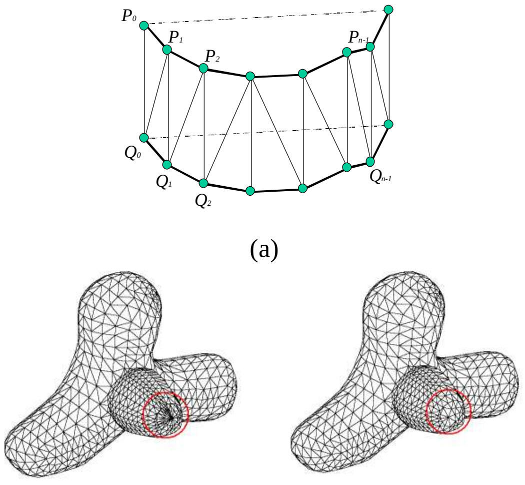
Figure 10 From Left to Right: Case 1, 2, and 3

After the two points are advanced, the left point and the right point are $P_1'$ and $P_2'$. In case one, $P_1' = P_3$ and $P_2' = P_2$; in case two, $P_1' = P_1$ and $P_2' = P_4$; and in case three, $P_1' = P_3$ and $P_2' = P_4$. Assume that the normal of the projection plane is $\vec{n}_p$, then the point $P_1^*$ and three vectors: $\vec{n}_p$, $\overrightarrow{P_1^*P_2^*}$, and $\vec{n}_p \times \overrightarrow{P_1^*P_2^*}$ build an orthogonal coordinate $O^*$; and point $P_1'$ and three vectors: $\vec{n}_p$, $\overrightarrow{P_1'P_2'}$, and $\vec{n}_p \times \overrightarrow{P_1'P_2'}$ build an orthogonal coordinate $O'$. Briefly, the procedure of copying the base ring B is to translate the vertices on B from coordinate $O^*$ to coordinate $O'$, and scale them by ratio $\left\|\overrightarrow{P_1'P_2'}\right\| \Big/ \left\|\overrightarrow{P_1^*P_2^*}\right\|$.

### 5.4 Sew adjacent rings

In this step of mesh surface extrusion, the original polygons surrounded by the base ring B are deleted first, and then the new polygons are created by sewing the neighboring copies of the base ring together (Meyers et al., 1992) (Figure 11a).

After sewing the adjacent rings, a new 3D polygonal object is created as shown in Figure 11b. Analyzing the mesh representation of the 3D polygonal object, we find that there are many short edges and small triangles at the top of the extruded mesh surface (circled by red pen). The mesh optimisation algorithm introduced in section 4 is used to remove these short edges and small triangles; the result is shown in Figure 11c.



(a)

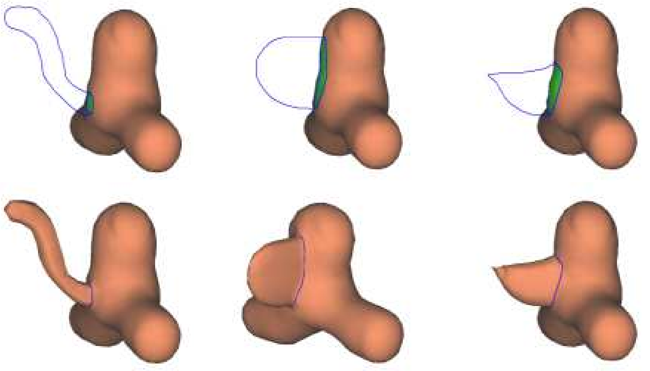
(b) (c)

Figure 11 Sew Adjacent Rings

## 6. CONCLUSION AND DISCUSSION

In this paper, we propose a useful 3D mesh extrusion method for intuitive, efficient geometric modeling of freeform polygonal mesh surface. With our method, the user can sketch two strokes to extrude a polygonal mesh surface. The fundamental algorithms of this approach are partial mesh re-triangulation and mesh optimization. They are introduced in this paper first. After presenting the two fundamental algorithms, the extrusion algorithm is introduced in detail. The method can be widely used in the modeling of freeform polygonal objects.

Users can create a wide variety of shapes using this mesh extrusion tool, as shown in Figure 12.



(a) long (b) thin (c) sharp

Figure 12 Examples of Mesh Extrusion

The system uses the same algorithm to dig a cavity on the surface (Figure 13). The implementation of extrusion does not support holes that completely extend to the other side of the object. Further research can focus on how to solve such problems.

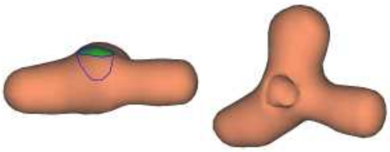


Figure 13 Dig a Cavity

## ACKNOWLEDGMENTS

The authors would like to acknowledge the comments made by Dr. Tai Chiew-Lan, Mr. Terry Chang, and the support of the CAD/CAM Facility of the Hong Kong University of Science and Technology.

## REFERENCES

Bloomenthal, J., Wyvill, B., “Interactive techniques for implicit modeling”, *1990 Symposium on Interactive 3D Graphics*, pp. 109-116, 1990.

Bloomenthal, J., *Introduction to implicit surfaces,* San Francisco, C.A.: Morgan Kaufmann Publishers, Inc., 1997.

Frey, P.J., Borouchaki, H., “Geometric surface mesh optimization”, *Computing & Visualization in Science*, vol.1, no.3, Nov. 1998, pp.113-21. Publisher: Springer-Verlag, Germany.

Garland, M., Heckbert, P.S., “Surface simplification using quadri error metrics”, *SIGGRAPH 97 Conference Proceedings*, pp. 209-16, 1997.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., “Mesh optimization”, *SIGGRAPH 1993 Conference Proceedings*, 1993, pp.19-26.

Hoppe, H., DeRose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schweitzer, J., Stuetzle, W., “Piecewise smooth surface reconstruction”, *SIGGRAPH 94 Conference Proceedings*, 1994, pp.295-302.

Igarashi, T., Tanaka, H., Matsuoka, S., “Teddy: A Sketching Interface for 3D Freeform Design”, *SIGGRAPH 1999 Conference Proceedings*, 1999.

Keppel, E., “Approximating complex surfaces by triangulation of contour lines”, *IBM Journal Res. Develop,* Jan. 1975, pp. 2-10.

Khodakovsky A. and Schroder P., “Fine level feature editing for subdivision surfaces”, *Proceedings Fifth Symposium on Solid Modeling and Applications*, Jun. 1999, pp.203-11. N.Y.: ACM Press, USA.

Markosian, L., Cohen, J.M., Crulli, T., Hughes, J., “Skin: a constructive approach to modeling free-form shapes”, *SIGGRAPH 99 Conference Proceedings*, 1999.

Meyers, D., Skinner, S., Sloan, K., “Surface from Contours”, ACM Transaction on Graphics, vol. 11, no. 3, pp. 228-258, 1992.

Rademacher, P., “View-dependent geometry”, *SIGGRAPH 99 Conference Proceedings*, pp.439-446, 1999. New York, NY, USA.

Schroeder, W.J., Zarge, J.A., Lorensen, W.E., “Decimation of triangle meshes”, *Computer Graphics*, v26, n2, pp.65-70, 1992, USA.

Suzuki, H., Sakurai, Y., Kanai, T., Kimura, F., “Interactive mesh dragging with an adaptive remeshing technique”, *Visual Computer*, vol.16, no.3-4, 2000, pp.159-76. Publisher: Springer-Verlag, Germany.

Wang, C.C.L., Yuen, M.M.F., "A generic algorithm of mesh optimization", *International Journal of Advanced Manufacturing Technology*, to appear in 2001. Publisher: Springer-Verlag, UK.

Zeleznik, R.C., Herndon, K.P., Hughes, J.F., “SKETCH: An interface for sketching 3D scenes”, *SIGGRAPH 1996 Conference Proceedings*, pp. 163-170, 1996.

Zorin, D., Schroder, P., Sweldens, W., “Interactive Multiresolution Mesh Editing”, *SIGGRAPH 97 Conference Proceedings*, 1997.