Computer-Aided Design and Applications

http://www.cadanda.com

Constructing Triangular Meshes of Minimal Area

Wenyu Chen[1], Yiyu Cai[2] and Jianmin Zheng[3]

[1]Nanyang Technological University, Singapore, ciweiyu@pmail.ntu.edu.sg
[2]Nanyang Technological University, Singapore, MYYCai@ntu.edu.sg
[3]Nanyang Technological University, Singapore, ASJMZheng@ntu.edu.sg

ABSTRACT

This paper is concerned with the problem of constructing an aesthetically pleasing triangular mesh with a given closed polygonal contour in three dimensional space as boundary. Triangular meshes of minimal area from all triangular meshes with the prescribed boundary are suggested as the candidates for this problem. An iterative algorithm of constructing such a triangular mesh from a given polygonal boundary is presented. Experimental examples show that the proposed algorithm is reliable and effective. Some related theoretical issues, possible extensions and applications are also discussed.

Keywords: geometric modeling, minimal surfaces, plateau problem, dirichlet approach.
DOI: 10.3722/cadaps.2008.508-518

1. INTRODUCTION

One of the main tasks in geometric modeling is the generation of aesthetically pleasing surfaces [14]. The surfaces are usually created from some inputs such as a set of 3D points or curves that serve as constraints or guidance for the surfaces. However, the selection of the most appealing surface or the most reasonable surface from the given constraints is subjective. There is no best answer for all situations. In practice, many energy functionals have been used, which are defined in terms of elastic membranes or thin plates or geometric invariants like curvatures. The surface with the minimal energy is selected.

This paper studies the problem of how to construct visually pleasing shapes from prescribed boundary and minimal surfaces are considered to be the natural solutions. Fig. 1 shows two famous minimal surfaces: the Helicoid and Catenoid surfaces. Minimal surfaces refer to the surfaces that minimize surface area. The physical models of minimal area surfaces can be made by dipping a closed, curved wire frame into a solution of soap and water and withdrawing it. A soap film is formed, which is a minimal surface whose boundary is the wire frame. The "naturalness" of minimal surfaces may be partially explained by this fact from physics: the surface tension that governs the film's shape is proportional to the area, the film tries to minimize the tension everywhere subject to the fixed boundary constraint, and thus the shape tends to form the surface of minimal area among nearby surfaces with the same boundary. Due to their special properties, minimal surfaces often become the candidates of ideal models in many applications of different fields such as molecular engineering, materials science, and architecture. For example, minimal surfaces were used in architecture for light roof constructions, form-finding models for tents, nets and air halls. In computer graphics, polyhedra of minimal surface area were suggested as natural candidates for object models [17] and the triangular tiles of minimal surface area were used to interpolate parallel slices [10].

Mathematically, minimal surfaces are characterized as surfaces whose mean curvature vanishes everywhere, reflecting the fact that there is no pressure differential across the surface. Finding a surface that minimizes the area is actually a problem of calculus of variations [12]. In particular, the problem of finding the minimal surface for a given boundary curve is known as the Plateau problem after the Belgian Physicist Plateau who carried out extensive experiments with soap films in the mid-nineteenth century. Out of his investigations there developed a conjecture that every closed, non-self-intersecting curve can be spanned by a minimal surface. The conjecture was mathematically proved in 1930 by Rado [19] and in 1931 by Douglas [7] independently. In general, exact solutions are usually complicated and difficult to find. Many numerical methods have been developed to approximate the exact minimal surfaces. For example, Douglas used a finite element method to find a numerical solution of the Plateau problem [6] and Wilson used a boundary element method to produce an approximate minimal surface [26]. Wang *et al* [25] combined Trefftz finite element formulation with radial basis functions and the analogue equation method to analyze minimal surface problems. Different functional energies have been used in developing numerical methods. Area functional, mean curvature flow and the Dirichlet energy are the typical energies. Tsuchiya [22-24] proposed two numerical methods: one minimized the surface area and the other minimized the Dirichlet energy. Both solutions converged to the minimal surface in a suitable function space. Dziuk [9] used the mean curvature flow to compute stable minimal surfaces by a semi implicit finite element scheme. The minimal surfaces spanned by a polygon were studied by Hinze and a numerical method was proposed based on a theoretical result that the minimal surfaces spanning the polygon correspond in a one to one manner to the critical points of Shiffman's function [11].

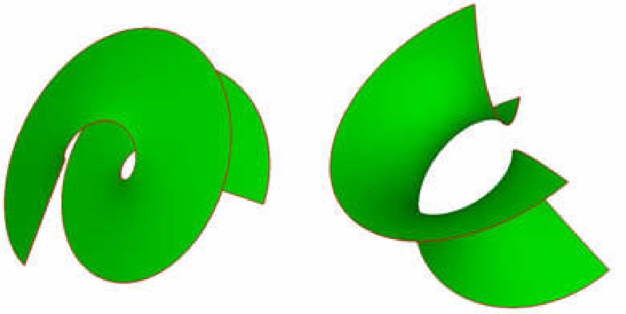

Fig. 1: The Helicoid and Catenoid surfaces.

Polynomial approximation to minimal surfaces is also of interest. Monterde *et al* studied the Plateau-Bézier problem that finds the surface of minimal surface area from among all Bézier surfaces with prescribed border [1,4,15]. Given three or four Bézier curves, the triangular Bézier patch or tensor-product Bézier patch which minimizes the Dirichlet energy can be found. It is shown that the resulting Bézier surface patch does not minimize area in general but has the area close to the minimum.

This paper presents a new method for constructing a triangular mesh of minimal area from a given polygonal boundary. Unlike previous numerical approaches that are based on sophisticated mathematics, the new approach is in the fashion of digital geometry processing [5,20,21], which is conceptually simple and easy to implement. A similar work that one referee points out to us was done by Pinkall and Polthier who presented a numerical minimization procedure to find discrete minimal surfaces bounded by a number of boundary curves [18]. The method begins with an initial mesh and iteratively updates the mesh by minimizing the Dirichlet integral. Our method has three basic processes: area minimizing, Laplacian fairing and edge swapping. The first two processes are used to optimize the geometry of the mesh and the last one is used to adjust the connectivity of the triangular mesh. The paper also gives a simple initialization step which can control the level of detail of the resulting mesh. The experimental results have demonstrated that the new approach performs very stably and effectively. Since nowadays triangular meshes are widely used in computer aided design and computer graphics because of their simplicity and powerful capability to model complicated shapes, the new method can find applications where smooth, visual appealing shapes are required. In addition, the method can also be used as a visualization tool in minimal surface study [3].

The rest of the paper is structured as follows. Section 2 reviews some important concepts and properties relevant to minimal surfaces. They serve as guidance for developing our new algorithm that is described in detail in Section 3. Section 4 provides some examples to demonstrate how well our algorithm can generate visual pleasing shapes. Finally in Section 5 some theoretical and practical issues are discussed.

2. MINIMAL SURFACES

Let Ω, a closed subset of R^2, be the parameter domain of the surfaces, with boundary $\partial\Omega$. Let Γ be the given 3D curve defined over $\partial\Omega$. The Plateau problem is to find a parametric surface $r(u,v),(u,v) \in \Omega$, which is the solution of the minimization problem:

$$\min_{r} \iint_{\Omega} \left| r_u(u,v) \times r_v(u,v) \right| dudv = \min_{r} \iint_{\Omega} \sqrt{EG - F^2}\, dudv \tag{1}$$

with $r(u,v)\,|_{\partial\Omega} = \Gamma$, where E, F and G are the coefficients of the first fundamental form of $r(u,v)$.

As can been seen, the area expression in Eqn. (1) is in general complicated. A lot of approaches in numerical approximation of minimal surfaces do not minimize the area functional directly. Instead, they try to minimize the following functional called the Dirichlet functional (or thin-plate energy of surface $r(u,v)$ in geometric modeling):

$$\iint_{\Omega} (E + G)\, dudv = \iint_{\Omega} (r_u^2(u,v) + r_v^2(u,v))\, dudv \tag{2}$$

The area functional and the Dirichlet functional have the following relation:

$$\iint_{\Omega} \sqrt{EG - F^2}\, dudv \le \iint_{\Omega} \sqrt{EG}\, dudv \le \frac{1}{2} \iint_{\Omega} (E + G)\, dudv$$

Obviously, both functionals are the same if and only if $E = G$ and $F = 0$, which implies that the surface $r(u,v)$ is a conformal mapping. In addition, while the area functional is independent of the parameterization of the surface, the Dirichlet one depends on the parameterization. However, the Dirichlet functional is easier to manage and there holds an important result: both the area and Dirichlet functional have the same extremals in the unrestricted case [16]. In the Bézier case (i.e., the minimal surfaces are restricted to polynomial surfaces), the Dirichlet extremals are an approximation to the extremals of the areal functional [15].

The variational derivative of the Dirichlet functional corresponds to the Laplacian and can be expressed as

$$\Delta r(u,v) = r_{uu} + r_{vv}$$

where Δ is the Laplacian operator. Therefore, if a surface $r(u,v)$ is harmonic, i.e., $\Delta r(u,v) = 0$, it minimizes the Dirichlet energy. Furthermore, if $r(u,v)$ is also conformal, then it is a minimal surface.

3. CONSTRUCTION OF OPTIMAL TRIANGULAR MESHES

We now describe our problem (the discrete version of the Plateau problem). Suppose that we are given a simple polygon Γ with n vertices and an integer $m(\ge n)$ that has the same parity as n. There are an infinite number of triangular meshes with m triangles spanning Γ. Our task is to find one triangular mesh from the set of such triangular meshes, which has the minimal area. Note that the requirement of the same parity of m and n is due to the Euler-Poincaré formula. In addition, though we can construct a triangular mesh with $n-2$ triangles for the given boundary Γ, we should in general have sufficient number of triangles to make the mesh look smooth. Therefore in this paper we ignore the case of $m = n-2$ and always assume that $m \ge n$. When $m \ge n$, the triangular mesh will contain some vertices other than the given ones of the boundary. The coordinates of the new vertices provide degrees of freedom for optimizing the shape of the triangular mesh. Number m can also be viewed as a control for levels of detail in approximation of a continuous minimal surface.

A triangular mesh contains two aspects of information: geometry and connectivity. Geometry is defined by the coordinates of vertices of the mesh and it tells the location of the mesh in 3D space. Connectivity defines how the vertices are joined to form the mesh. Our objective is to create a triangular mesh which is optimal in both geometry and connectivity. Optimizing geometry and connectivity simultaneously is a very difficult problem. Our strategy is to separate geometry and connectivity. For given connectivity of vertices of the mesh, we optimize geometry (i.e., the coordinates of vertices) and for fixed vertices of the mesh, we find an optimal triangulation. In the paper the former will be implemented by the processes of area minimizing and Laplacian fairing and the latter by the process of edge swapping. These three processes are the main ingredients of our proposed algorithm. Below they will be explained first and then the algorithm is presented.

3.1 Area Minimizing

Let a triangular mesh M be represented as a triple $< I, P, T >$, where $I = \{1, 2, ..., N\}$ is its vertex index set, $P : I \to R^3$ is a mapping from the vertex indices to their locations in 3D space, T is its triangle set, and each triangle $t \in T$ is represented as an ordered vertex index triple $t =< i, j, k >$ meaning that the triangle is defined by vertices $P(i), P(j)$ and $P(k)$. Without causing ambiguity, we use P_i to replace $P(i)$ for simplicity. In triple $< I, P, T >$, we assume that the first n vertices $P_i, i = 1, \cdots, n$ are the given vertices on the boundary Γ.

The area A of mesh M is just the sum of all triangles' areas:

$$A=\sum_{t=<i,j,k>\in T}\frac{1}{2}\mid P_jP_k\times P_jP_i\mid=\sum_{t=<i,j,k>\in T}\frac{1}{2}\sqrt{(P_jP_k\times P_jP_i)^2} \tag{3}$$

With the boundary vertices fixed, area A is a function of vertices $P_{n+1},\cdots,P_N$. The process of *area minimizing* is to find appropriate positions for $P_{n+1},\cdots,P_N$ such that the area functional A will be minimized.

Let $NT(i)\subseteq T$ be the set of all the triangles that contain vertex P_i and $\frac{\partial A}{\partial P_h}=\left(\frac{\partial A}{\partial(P_h)_x},\frac{\partial A}{\partial(P_h)_y},\frac{\partial A}{\partial(P_h)_z}\right)^T$. By some simplification, we have for $h=n+1,\cdots,N$

$$\begin{aligned}\frac{\partial A}{\partial P_h}&=\frac{1}{2}\sum_{t=<i,j,k>\in T}\frac{\partial}{\partial P_h}\sqrt{(P_jP_k\times P_jP_i)^2}\\&=\frac{1}{2}\sum_{t=<h,j,k>\in NT(h)}\frac{1}{2}\frac{\frac{\partial}{\partial P_h}(P_jP_k\times P_jP_h)^2}{\sqrt{(P_jP_k\times P_jP_h)^2}}\\&=\frac{1}{2}\sum_{t=<h,j,k>\in NT(h)}\frac{(P_jP_k)^2P_jP_h-(P_jP_k\cdot P_jP_h)P_jP_k}{\sqrt{(P_jP_k\times P_jP_h)^2}}\end{aligned} \tag{4}$$

Setting all these derivatives to zero leads to $(N-n)$ equations with $(N-n)$ variables. The solution renders us an optimal mesh. However, the equations are non-linear. It is difficult to solve such a non-linear system. For a mesh with a large data set, the situation even becomes worse.
Here we propose a local mechanism and iteratively approximate the solution. Rewrite Eqn. (4) as

$$\frac{\partial A}{\partial P_h}=\frac{1}{2}CP_h+\frac{1}{2}\sum_{t=<h,j,k>\in NT(h)}\frac{(P_jP_k\cdot P_j)P_jP_k-(P_jP_k)^2P_j}{\sqrt{(P_jP_k\times P_jP_h)^2}}$$

where

$$C=\sum_{t=<h,j,k>\in NT(h)}\frac{(P_jP_k)^2\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}-(P_jP_k)(P_jP_k)^T}{\sqrt{(P_jP_k\times P_jP_h)^2}}$$

is a 3×3 matrix. Letting $\frac{\partial A}{\partial P_h}=0$, we have

$$P_h=-C^{-1}\sum_{t=<h,j,k>\in NT(h)}\frac{(P_jP_k\cdot P_j)P_jP_k-(P_jP_k)^2P_j}{\sqrt{(P_jP_k\times P_jP_h)^2}}$$

The above equation cannot be considered to be an explicit solution for P_h because the right-hand side of the equation also contains P_h. However, it gives us a way to update vertex P_h in the fashion of signal processing [21]. That is, we compute the new vertex $\bar{P}_h$ from the old P_h and its 1-ring neighboring vertices by

$$\bar{P}_h=-C^{-1}\sum_{t=<h,j,k>\in NT(h)}\frac{(P_jP_k\cdot P_j)P_jP_k-(P_jP_k)^2P_j}{\sqrt{(P_jP_k\times P_jP_h)^2}} \tag{5}$$

When formula Eqn. (5) applies to all the interior vertices $P_{n+1},\cdots,P_N$ once, this completes one iteration. The process continues until the area change by one iteration is smaller than a prescribed tolerance.

3.2 Laplacian Fairing

We can also examine the Dirichlet approach. Since the Dirichlet functional Eqn. (2) depends on the parameterization of the surface, it cannot be used directly for a mesh model. However, we can consider its variational derivative—the Laplacian. For a triangular mesh, a discrete Laplacian should be used. Let $N(i)\subseteq I$ be the index set of the 1-ring neighboring vertices of vertex i. The Laplacian operator $\Delta()$ can be approximated at each vertex by the umbrella operator:

$$\Delta(P_i) = \sum_{j \in N(i)} w_{ij}(P_j - P_i)$$

where the weights w_{ij} are positive numbers that sums to one for each i. There are many ways to choose the weights based on the neighborhood structures. In this paper, we choose

$$w_{ij} = \frac{S_{ij}}{\sum_{k \in N(i)} S_{ik}}$$

where S_{ij} is the area of the two triangles that share the edge connecting P_i and P_j.

Now for all interior vertices $P_i, i = n+1, \cdots, N$ of mesh M, let their Laplacian $\Delta(P_i)$ equal zero. This results in $N-n$ equations with $N-n$ unknowns. Unfortunately, the equations are non-linear due to the fact that our chosen Laplacian operator is non-linear. Then we take a similar approach that we used in Section 3.1. We update P_i to $\overline{P}_i$ by averaging its old neighboring vertices:

$$\overline{P}_i = \sum_{j \in N(i)} w_{ij} P_j.$$

where w_{ij} are computed from old P_i and P_j. When all the interior vertices are updated, this completes one iteration. Similarly, the iteration continues until the area change is smaller than the prescribed tolerance. Since the above updating is similar to Laplacian filtering, the process is called *Laplacian fairing*.

3.3 Edge Swapping

Given a set of points in 3D space, there are many ways to connect them to form a triangular mesh. Obviously, the number of possible triangulations is huge. Not all of them possess equally pleasing shapes and for a particular application some will be much more acceptable than others. This suggests that we need to find an optimal triangulation in some sense.

Here we intend to improve a given mesh by changing its connectivity via a simple local transformation, called the *edge swapping*. Refer to Fig. 2. The edge swapping transforms two triangles sharing one edge shown on the left into another two triangles shown on the right. Note that during this transformation, the four vertices remain unchanged. What has changed is the connectivity among the vertices. The edge swapping should only be performed if it improves the mesh. For our application, the improvement is measured by the area reduction.

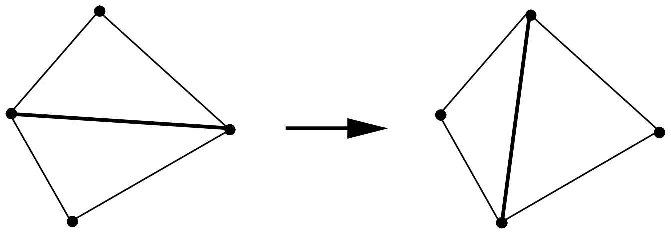

Fig. 2: Edge swapping.

For given mesh M, our edge swapping algorithm is performed using Lawson's local optimization approach [8]. It visits each interior edge of M and checks whether the edge swapping reduces the area of the mesh. If an area reduction does result, the algorithm swaps the edge. After the algorithm visits all edges in this way, it completes one iteration. If the algorithm made any swaps during the first iteration, it conducts a second iteration of edge visits. This process continues until no swap is made in one iteration.

This process is simple and fast. It always terminates in a finite number of iterations. However, it is essentially a "best-first" algorithm, which usually has the drawback that it may return a local optimum. To get globally optimal solutions, simulated annealing technique may be considered [2], but the computational cost is much higher.

3.4 Algorithm

Note that area minimizing, Laplacian fairing and edge swapping cannot be applied to our boundary condition directly because these three processes all assume that there has already existed a triangular mesh. Therefore we need an initialization step to create an initial triangular mesh from our input.

Here we give a simple way to create our initial triangular mesh. Given the boundary $P_1 \cdots P_n$ and a triangle number m, we first compute the central point of the polygon by averaging the vertices: $P_c = \sum_{i=1}^{n} P_i/n$. Then for each vertex P_i, we connect it to P_c and also add s interior vertices P_i^j along line segment P_iP_c:

$$P_i^j = P_i + \frac{j}{s+1}(P_c - P_i), j = 1, 2, \cdots, s$$

where $s = \left\lfloor \frac{m-n}{2n} \right\rfloor$ stands for how many rings will be added. Next, for each j, we join $P_1^j, P_2^j, \cdots, P_n^j$ by the order to form a polygon. This results in a mesh which contains triangles and possibly quadrilaterals. For the quadrilateral facets such as $P_i^j P_{i+1}^j P_{i+1}^{j+1} P_i^{j+1}$, we split them into two triangles by inserting a diagonal edge such as $P_i^j P_{i+1}^{j+1}$. So far we have created a triangular mesh which consists of $(2s+1)n$ triangles (refer to Fig. 3(a)). If $(2s+1)n$ does not equal to m, we need to further insert $m-(2s+1)n$ triangles to match m. This can be done by arbitrarily choosing $(m-(2s+1)n)/2$ triangles (for example, those on the outer ring) and split each of them into 3 sub-triangles with a new vertex at the center of the triangle (see Fig. 3(b)).

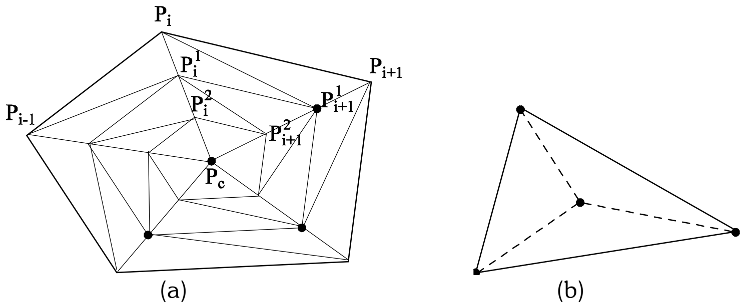

Fig. 3: (a) Creating an initial mesh with $s = 2$ and (b) refining an individual triangle.

It should be pointed out that this initialization method is simple and easy to create a mesh with required triangle number, but it is not optimized and can be improved by taking the shape of the boundary into consideration and/or employing some optimization criteria such as those used in Delaunay triangulation. The current initialization method may generate an initial mesh which is far from satisfactory, but our subsequent techniques are able to correct it. The experimental examples have demonstrated that our proposed algorithm can always return the triangular meshes of minimal area. Fig. 4 shows one of such examples, in which the initial mesh contains self-intersection. In addition, the initialization step also sets the topology of the final minimal surface besides providing an initial mesh. This is because the subsequent processes do not change the topology of the mesh as a surface. Therefore if a special topology is expected, a different initialization is needed.

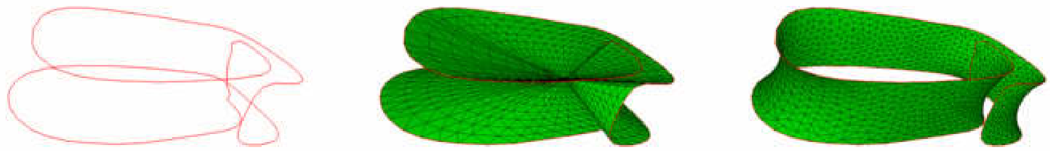

Fig. 4: Left: the input boundary; middle: the generated initial mesh; right: the final minimal area surface.

Once an initial triangular mesh is created, the edge swapping, Laplacian fairing and area minimizing are used to improve the connectivity and positions of vertices. It can be observed that Laplacian fairing works as a filtering and its numerical performance is much more stable than that of area minimizing. This suggests that we should apply Laplacian fairing first and then use area minimizing. Therefore we propose our algorithm as follows:

Step 1. Initialization
Edge swapping
Step 2. DO {

Laplacian fairing
Edge swapping
} WHILE (area change $> \varepsilon_1$)

Step 3. DO {
Area minimizing
Edge swapping
} WHILE (area change $> \varepsilon_2$)

In the above algorithm, ε_1 and ε_2 are two prescribed tolerances that control when the iterations should stop. Step 1 creates an initial mesh with the given boundary polygon. Step 2 is kind of discrete Dirichlet approach which gives a good approximation to the solution. Step 3 further refines the approximation by minimizing the area functional. In all these steps, the edge swapping is added to improve the connectivity of the mesh.

4. EXPERIMENTAL EXAMPLES

This section provides some examples to demonstrate the algorithm. In particular, the first two examples are designed to check the validity of the algorithm. Their input boundary polygons were generated from the classic Helicoid and Catenoid minimal surfaces. In addition, we also examine the mean curvature of the resulting triangular meshes. The concept of mean curvature for a mesh surface and its computational formula are from *discrete* differential geometry [13]. The absolute value of mean curvature k_i at vertex P_i on a mesh is computed by

$$|k_i| = \frac{1}{4A_i}\left|\sum_{j\in N(i)} (cot\alpha_{ij} + cot\beta_{ij})(P_j - P_i)\right|$$

where A_i is is the sum of areas of triangles that contain vertex P_i, and α_{ij}, β_{ij} are two angles corresponding to edge P_iP_j (refer to Fig. 5 for illustration).

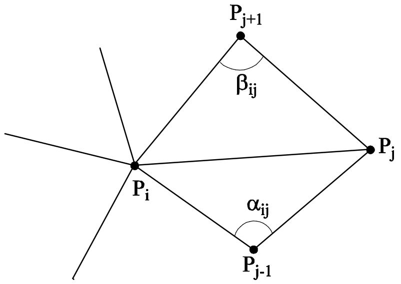

Fig. 5: Illustration of symbols in the mean curvature formula.

The input polygon in the first example is shown in Fig. 6(a), which consists of 186 points. There points were obtained by sampling the boundary of a patch of the Helicoid surface. The patch is defined by parametric equations

$$x = u\cos v, \quad y = u\sin v, \quad z = v, \qquad (u,v) \in [2,10]\times[-1,10]$$

and has an area of 536.7. Now we use our algorithm to find an open triangular mesh with the input polygon as boundary. The triangular mesh should contain a required number of triangles and have the minimal area. In this example, we let the triangle number be 2050 and two tolerance $\varepsilon_1, \varepsilon_2$ be 0.01. The results are shown in Fig. 6, where (b), (c) and (e) are the outputs of Step 1 (initialization), Step 2 (Laplacian fairing) and Step 3 (area minimizing), respectively. The areas of the triangular meshes in Fig. 6(c) and (e) are 536.17 and 536.07, which are even smaller than the actual area of the Helicoid patch. This can be explained by the fact that our input polygon is only an approximation to the exact patch boundary. The maximum absolute values of mean curvature of the meshes in Fig. 6(c) and (e) are 0.1 and 0.0007. Fig. 6(d) and (f) are the mean curvature images of (c) and (e), in which color blue stands for low mean curvature and color red for high mean curvature. It is clearly seen from the curvature images that the mesh shown in Fig. 6(c) is further improved by area minimizing which outputs Fig. 6(e). Both the area and the mean curvature indicate that the triangular mesh in Fig. 6(e) is a good solution.

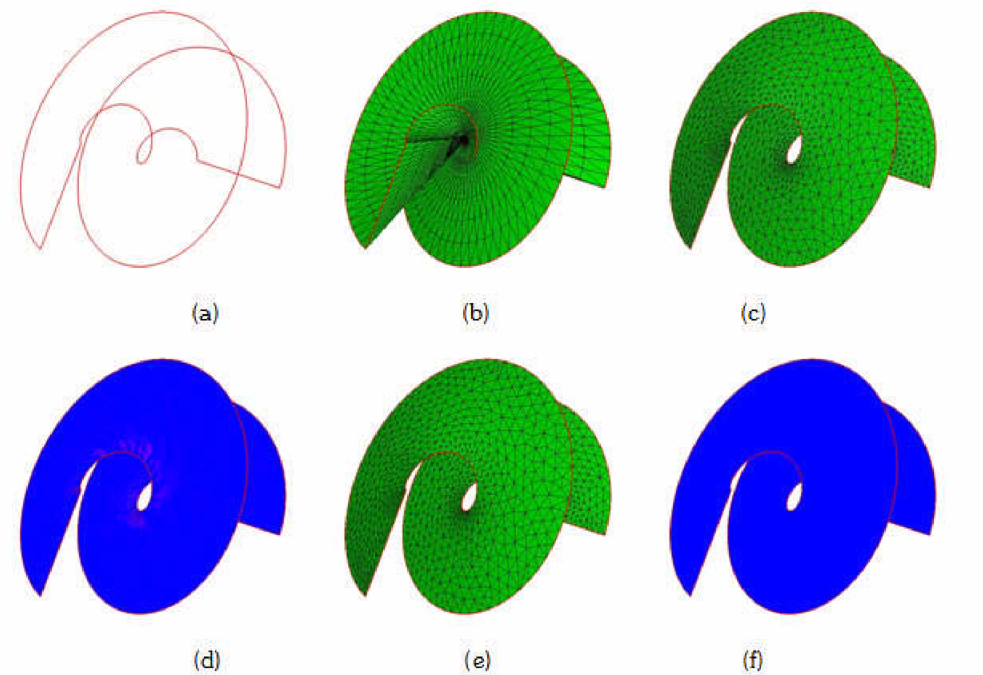

Fig. 6: Reconstructing the Helicoid surface from a polygon.

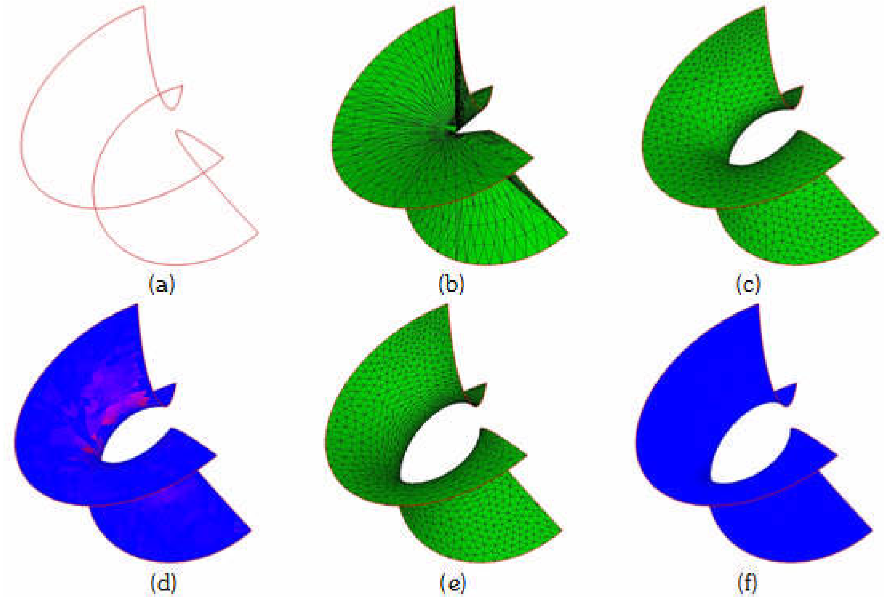

Fig. 7: Reconstructing the Catenoid surface from a polygon.

The second example is to reconstruct the Catenoid surface from a given polygon as boundary. The polygon contains 156 points that were originally sampled from the boundary of a Catenoid patch defined by $x = \cosh v \cos u, y = \cosh v \sin u$, $z = v, (u,v) \in \left[-\frac{3\pi}{5}, \frac{3\pi}{5}\right] \times [-2,2]$. The area of the patch is 58.98 . We apply our algorithm to the polygon with the triangle number of 2040 and $\varepsilon_1 = \varepsilon_2 = 0.01$. The results are shown in Fig. 7, where (a) is the input polygon, (b) is the mesh outputted from Step 1, (c) is the result from Step 2, which has an area of 58.95 and the maximal mean curvature of 0.23, (e) is the final mesh outputted from Step 3, which has an area of

58.2 and the maximal mean curvature of 0.004, (d) and (f) are the mean curvature images of (c) and (e), respectively. These statistics demonstrate the effectiveness of the algorithm.

Two more examples are shown in Fig. 8, where the left column shows two input polygons consisting of 160 and 97 vertices, respectively, the middle and right columns show the final minimal area meshes of different triangle numbers. The statistics are shown in Tab. 1 and Tab. 2. It can be seen that the area of output triangular meshes decreases when the level of detail increases.

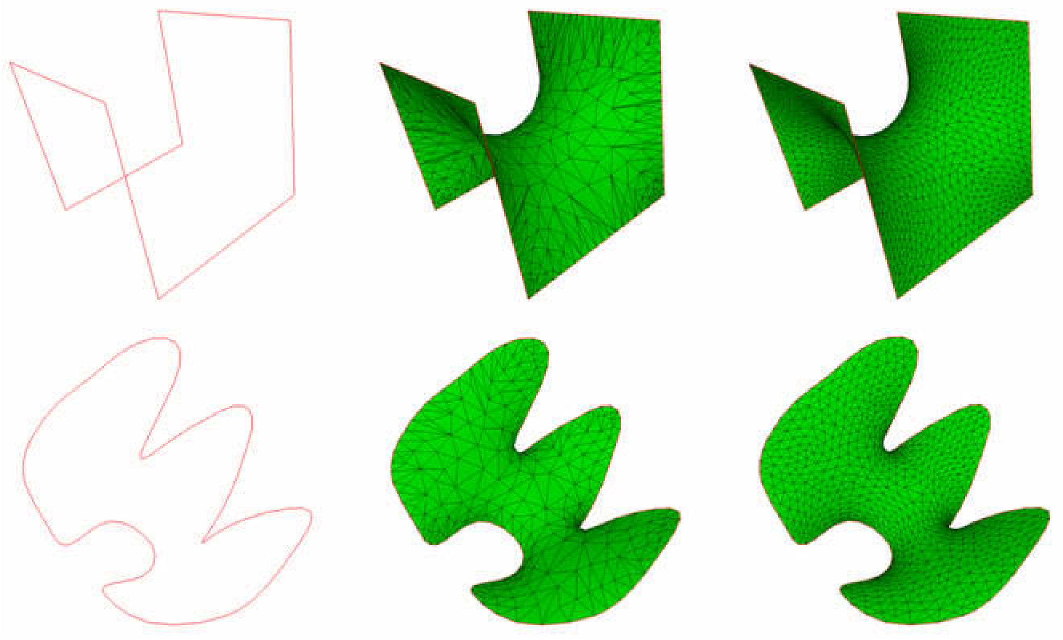

Fig. 8: Construction with different levels of detail.

	resulting mesh on the middle	resulting mesh on the right
triangle number	800	2400
area	9.9	9.87
max mean curvature	0.004	0.001

Tab. 1: Statistics for the example shown in the top row of Fig. 8.

	resulting mesh on the middle	resulting mesh on the right
triangle number	485	1455
area	1191.3	1187.8
max mean curvature	4.8×10^{-5}	1.1×10^{-5}

Tab. 2: Statistics for the example shown in the bottom row of Fig. 8.

5. DISCUSSION

In this paper, we have proposed an algorithm to construct triangular meshes of minimal area from all possible triangular meshes with the prescribed boundary and number of triangles. The core techniques of the algorithm are three processes: area minimizing, Laplacian fairing, and edge swapping. They are combined to provide an automatic approach. The algorithm has been shown by the examples to be reliable and effective. On the other hand, since these three processes can be done very fast, they can be used in interactive environments. Especially in digital geometry modeling, after users sketch the boundary and specify the level of detail, the three processes can then be interactively performed in various orders to achieve users' specific requirement.

Although the paper focuses on the construction of triangular meshes from a single contour, it is possible to extend the idea and the three processes to finding the minimal surfaces from multi-contours or with other constraints. Fig. 9 shows one example, where the input consists of two contours and the minimal surface interpolating the contours is constructed using our algorithm. Here we only need one extra process. That is to generate an initial triangular mesh from the given contours. Applications of the algorithm to other fields are also possible and we are currently studying the application to mesh fairing.

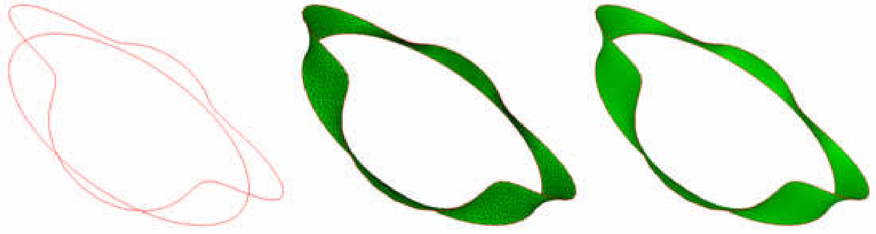

Fig. 9: Left: two input contours; middle: the resulting mesh; right: the shading image.

This research also generates some theoretical problems, one of which is the convergence analysis of the algorithm. Though the algorithm succeeds for all our testing examples, we do not know exactly what conditions guarantee the convergence. Meanwhile, this problem is also related to the existence and uniqueness properties of our minimal surface problem. The existence of triangular meshes of minimal area can be proved. In fact, the area functional Eqn. (3) can be considered as a continuous real function defined on $R^{3(N-n)}$ since we have $N-n$ free vertices $P_{n+1},\cdots,P_N$ and each vertex has three coordinates. The area cannot be negative and thus it is bounded from below. Furthermore, when we look for a minimum, we can restrict the function to a suitable compact subset such that if a vertex goes outside of the compact subset, then the area functional becomes greater than some bound. Thus we affirm from calculus that a minimum exists and it is attained. However, the uniqueness of minimal area triangular meshes is generally not true. This can be seen from a special situation where the given polygonal boundary lies on a plane. Then the area of the region bounded by the polygon is minimal. If a triangular mesh reaches the minimal area, it must lie on the plane. If we move one vertex of the triangular mesh on the plane a little bit, obviously we obtain a new triangular mesh but we do not change the total area. This implies that we could have many triangular meshes which all have the minimal area. These theoretical properties of our minimal surface problem suggest that our algorithm should be improved such that it converges not only to a minimal area triangular mesh, but also to one that possesses a good quality connectivity in some sense. This is a challenging and worthwhile goal.

6. ACKNOWLEDGEMENTS

This work was partially supported by the A*STAR SERC Grant (No 0621020034) of Singapore. We are grateful to the reviewers for their helpful suggestions and for pointing out reference [18] which we missed in the earlier version of the paper.

7. REFERENCES

[1] Arnal, A.; Lluch, A.; Monterde, J.: Triangular Bézier surfaces of minimal area, Proceedings of Computational Science and Its Applications - ICCA, 2003, 366-375.

[2] Baszenski, G.; Schumaker, L.: Use of simulated annealing to construct triangular facet surfaces, Curves and Surfaces, Academic Press, Boston, 27-32.

[3] Callahan, M.; Hoffman, D.; Hoffman, J.: Computer Graphics tools for the study of minimal surfaces, Communications of the ACM, 1988, 648-661.

[4] Cosin, C.; Monterde, J.: Bézier surfaces of minimal area, Proceedings of Computational Science-ICCS, 2002, 72-81.

[5] Desbrun, M.; Meyer, M.; Schröder, P.; Barr, A.: Implicit fairing of arbitrary meshes using diffusion and curvature flow, Proceedings of ACM SIGGRAPH 1999, ACM, 317-324.

[6] Douglas, J.: A method of numerical solution of the plateau problem, Annals of Mathematics, 1928, 180-188.

[7] Douglas, J.: Solution of the problem of Plateau, Trans. Am. Math. Soc., 1931, 263-321.

[8] Dyn, N.; Levin, D.; Rippa, S.: Data dependent triangulations for piecewise linear interpolation, IMA J. Numerical Analysis, 1990, 137-154.

[9] Dziuk, G.: An Algorithm for Evolutionary Surfaces, Num. Math., 1991, 603-611.

[10] Fuchs, R.; Kedem, M.; Uselton, S.: Optimal surface reconstruction from planar contours, CACM, 1977, 693-702.

[11] Henze, M.: On the numerical approximation and computation of minimal surface continua bounded by one-parameter families of polygonal contours, Appl. Numer. Math., 1997, 89-116.

[12] Lawson, H.: Lectures on Minimal Submanifolds, Math. Lecture Series, Publish or Perish, Inc., Wilmington, DE, 1980.

[13] Meyer, M.; Desbrun, M.; Schröder, P.; Barr, A. H.: Discrete differential-geometry operators for triangulated 2-manifolds, 2003, 35-57.

[14] Moreton, H.; Séquin, C.: Functional minimization for fair surface design, Proceedings of ACM SIGGRAPH, 1992.

[15] Monterde, J.: Bézier surfaces of minimal area: The Dirichlet approach, Computer Aided Geometric Design, 2004, 117-136.

[16] Nitsche, J.: Lectures on Minimal Surfaces, 1, Cambridge Univ. Press, Cambridge, 1989.

[17] O'Rourke, J.: Polyhedra of minimal area as 3D object models, International Joint Conference on Artificial Intelligence, 2, 1981, 24-28, 1981.

[18] Pinkall, U.; Polthier, K.: Computing discrete minimal surfaces and their conjugates, Experim. Math., 2, 1993, 15-36.

[19] Rado, T.: On Plateau's problem, Ann.Math., 1930, 457-469.

[20] Schröder, P.; Sweldens, W.: Digital Geometry Processing, Course Notes, ACM SIGGRAPH, 2001.

[21] Taubin, G.: A signal processing approach to fair surface design, Proceedings of ACM SIGGRAPG 1995, ACM.

[22] Tsuchiya, T.: On two methods for approximating minimal surfaces in parametric form, Math. Comput., 1986, 517-529.

[23] Tsuchiya, T.: Discrete Solution of the Plateau-problem and its convergence, Mathematics of Computation, 1987, 157-165.

[24] Tsuchiya, T.: A Note on Discrete Solutions of the Plateau-Problem, Mathematics of Computation, 1990, 131-138.

[25] Wang, H.; Qin, Q.; Arounsavat, D.: Application of hybrid Trefftz finite element method to non-linear problems of minimal surface, International Journal for Numerical Methods in Engineering, 2007, 1262-1277.

[26] Wilson, J. W.: On discrete Dirichlet and plateau problems, Num. Math, 1961, 359-373.