SURFACE FLATTENING BASED ON ENERGY MODEL

Charlie C.L.Wang[*1] **Shana S. -F.Smith**[2] **Matthew M.F.Yuen**[1]

[1]Department of Mechanical Engineering, The Hong Kong University of Science and Technology,

Clear Water Bay, Kowloon, Hong Kong

[2]Department of Industrial Education and Technology, Iowa State University,

Ames, Iowa 50011-3130, USA

Abstract

This paper presents a method for three-dimensional surface flattening, which can be efficiently used in three-dimensional computer aided garment design. First, facet model is used to present a complex surface. Then, a spring-mass model based on energy function is used to flatten the 3D mesh surfaces into 2D patterns. The surface elastic deformation energy distribution is depicted by a color graph, which determines a surface cutting line. The method presented here can efficiently solve flattening problems for complex surfaces. The accuracy of a developed surface can easily be controlled locally. Thus, compared to earlier methods, this method provides more flexibility for solving CAD and CAM problems.

Keywords: surface flatten, energy-based model, spring-mass system, computer aided garment design

1. Introduction

The fashion industry uses two-dimensional CAD tools at present. It is expected that three-dimensional tools will be used for garment design to improve the efficiency of pattern generation and for more attractive design presentation. Surface flattening plays a prominent role in the flow of the three-dimensional computer aided garment design.

As an example, Figure 11 shows the flow of the 3D Garment CAD system, which is developed by the Hong Kong University of Science and Technology. Firstly, we import several patterns from other two-dimensional garment CAD systems. After a sewing simulation process [1], we obtain the 3D garment patterns. This 3D garment can be modified in 3D model space by free-form deformation (FFD) tools [2] to obtain the new 3D

[*] Corresponding Author: Charlie C. L. Wang; E-mail: wangcl@ust.hk; Tel: (852) 2358-8095; Fax: (852) 2335-9298

garment. However, what fashion industries need is not 3D simulation but 2D patterns that can be used in the manufacturing processes. Thus, we need surface flattening technology to produce this model. In Figure 11, we use short pants patterns as original input. The patterns are sewed together into a 3D garment. After the garment is modified into a longer and thinner one in the 3D model space, we flatten it back into 2D patterns, which can be used in fashion industry.

Our paper aims to describe an approach to obtain the 2D patterns from a given 3D mesh surface.

2. Related Work

Developable surfaces have been studied for a long time. Carmo (1976) [3] defined a developable surface. For a one-parameter family of lines $\{\alpha(t), \beta(t) : t \in I\}$, a ruled surface *X* is defined.

$$X(t,v) = \alpha(t) + v\beta(t), \quad t \in I, \quad v \in R$$

A ruled surface is developable if

$$\left\langle \beta \times \frac{d\beta}{dt}, \frac{d\alpha}{dt} \right\rangle = 0$$

In other words, *X* is developable if β, $\frac{d\beta}{dt}$ and $\frac{d\alpha}{dt}$ are coplanar for all points on *X*. The simplest examples of developable surfaces are cylinders and cones, and the simplest non-developable surfaces are spheres. Every surface enveloped by a one-parameter family of planes is a developable surface. If two parameters are used to design a surface, it is difficult to reach the condition of a developable surface. However, most CAD software uses two parameters to design surfaces. Thus, the flattening of two-parameter surfaces is very important.

Shimada and Tada (1991) [4] presented a generic surface development algorithm. This algorithm is based on a meshed surface. In their algorithm, a dynamic programming method is used to develop a curved surface. An objective curved surface is decomposed into regions of adjacent strips. Then each region is developed, in turn, into a flattened shape. The whole shape is derived by solving a multi-stage decision process. However, the method of division (cutting) is too simple to deal with a complex surface.

Parida and Mudur (1993) [5] gave an algorithm to develop complex surfaces. Their algorithm first obtains an approximate planar surface, and then reorients cracks and overlapping parts in the developed plane to satisfy orientation constraints. The Parida and Mudur algorithm might generate many cracks and calculation errors.

Surface development is also a key problem in texture mapping. Ma and Lin (1988) [6] and Maillot et al. (1993) [7] presented optimization-based surface development methods for texture mapping. In their methods, triangular surfaces are first mapped onto a plane, and then final 2D shapes are obtained through an optimization calculation procedure. The methods of Ma and Lin (1988) and Maillot et al. (1993) are useful for simple surfaces, but have difficulty applying to complex trimmed surfaces. In particular, their methods cannot control accuracy for local parts of surfaces.

Sun and Fiume (1996) [8] presented a method to solve the surface flattening problem from another point of view. They formulated a new developable surface representation technique suitable for use in interactive computer graphics. The feasibility of their method was demonstrated by modeling a hanging scarf, ribbons, and bows. However, in most CAD systems, it is difficult to use their model to formulate a surface.

Fan et al. (1998) [1] showed a general method for 2D to 3D and 3D to 2D transformation, which can be efficiently used in three-dimensional computer aided garment design. The method utilizes a uniform triangular spring-mass deformable model. However, if the surface is not wrapped from two-dimensional patterns, it is difficult to obtain a flattened surface without any cutting.

McCartney et al. (1999) [9] presented an algorithm for flattening 3D surfaces described as a list of triangles. The algorithm incorporates an energy model in terms of the strain energy required to deform edges of the triangular mesh. The problem of arbitrarily siting darts or gussets to assist in the garment fitting process is also considered. But in their model, energy is only used to measure the movement of points, not for generating a force to move points.

This paper solves the surface flattening problem using an innovative method. First, we use a triangular mesh to represent a surface. Then, an energy based model is established to flatten a 3D surface into a 2D pattern. The accuracy of any local area of the surface can be easily controlled in the process of development. We use an interpolation function to show the energy distribution of a developed surface. Surface cutting is included in the algorithm to reduce error for the resulting developed surface. Our method can efficiently develop a complex trimmed surface, which is usually quite difficult to be developed by earlier methods.

3. Definition of Energy Function

In this paper, we use an energy model - planar triangular spring-mass system to obtain the flattening result of a 3D mesh surface. The procedure of surface flattening is a planar triangular mesh deformation process directed by the energy function of the spring-mass system.

A spring-mass system is established for the deformation of planar triangular mesh - Φ. Φ is a pair (K, P), where K represents the connectivity of the vertices, thus determining the topological type of the mesh; $P = \{P_1, \ldots, P_m\}$, $P_i \in \Re^2$ is a set of vertex position defining the shape of the mesh in $\Re^2$ (its geometric realization); the given 3D mesh surface is another pair (K, Q), where $Q = \{Q_1, \ldots, Q_m\}$, $Q_i \in \Re^3$, each vertex Q_i corresponding to a planar vertex P_i. Most of the physical parameters for the spring-mass system are derived from their corresponding geometric parameters. For example, forces, elastic deformation energy and masses are determined by the relative displacement between mesh nodes and the area of triangles. The difference between the current position of mesh and the final shape of the 2D pattern can be considered as the elastic deformation energy stored in the planar spring-mass system. One example of a spring-mass system is shown in Figure 1.

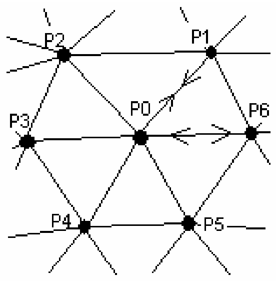


Figure 1 The spring-mass system

In this example, nodes P_i are masses and the links between masses P_i and P_j are springs. During deformation, if the distance between P_i and P_j on the planar surface is larger than the distance on the corresponding spatial surface, we apply an attraction force between them (e.g., P_0 and P_1 in Figure 1); otherwise there will be repellent force between them (e.g., P_0 and P_6 in Figure 1).

The elastic deformation energy function and tensile force on one single mass P_i are defined as

$$E(P_i) = \sum_{j=1}^{n} \frac{1}{2} C \left(\left| P_i P_j \right| - d_j \right)^2 \tag{1}$$

$$\vec{f}(P_i) = \sum_{j=1}^{n} C \left(\left| P_i P_j \right| - d_j \right) \vec{n}_{p_i p_j} \tag{2}$$

where C is the spring constant, $\left| P_i P_j \right|$ is the current distance between P_i and P_j on the planar surface, d_j is the distance between Q_i and Q_j, $\vec{n}_{p_i p_j}$ is the unit vector pointing from P_i to P_j, n is the number of nodes connected to P_i. Forces are generated by the triangle edge difference between the planar surface and the given spatial surface.

Recall that the goal of surface flattening is to obtain a planar triangular mesh Φ that provides a good fit to the given 3D mesh surface. We are going to find a simplical complex K and a set of vertex positions P defining a planar mesh $\Phi = (K, P)$ that minimizes the energy function

$$E(\Phi) = \sum_{i=1}^{m} E(P_i) \tag{3}$$

For some applications we want to control the flattening accuracy locally, different spring constants can be defined interactively for different parts of the surface. The higher the accuracy desired, the larger the value of the spring constants “ C ” is given.

While releasing the energy in our spring-mass model, most of our physical parameters such as system forces, masses and elastic deformation energy are defined by their corresponding geometric parameters (detail in section 5). Thus, our model is a spring-mass model in terms of geometry.

4. Measurement of Accuracy

Two criteria are used to evaluate the accuracy of a resulting developed surface.

Area Accuracy: During the developing process, the area of the surface may change. The final relative area difference is

$$E_S = \frac{\sum |A - A'|}{\sum A} \tag{4}$$

where A is the actual area of one patch on the surface before development and A' is the area of its corresponding patch after development.

The area of a surface $S(u,v)$ is calculated by $A = \iint_S \sqrt{EG - F^2}\, dudv$ [3], where $E = \frac{\partial S}{\partial u} \cdot \frac{\partial S}{\partial u}$, $G = \frac{\partial S}{\partial v} \cdot \frac{\partial S}{\partial v}$, $F = \frac{\partial S}{\partial u} \cdot \frac{\partial S}{\partial v}$. But in our approach, the facet model is used to present a complex surface. Thus, the analytic form of the surface ƺ cannot be determined. Here we simply approximate A by summing the area of each triangle in the facet model,

$$A = \sum_{i=0}^{n-1} A_i \tag{5}$$

where A_i is the area of the ith triangle in the facet model, and n is the triangle number in the facet model.

Shape Accuracy: Shape error can be represented by the difference between curve length on the original surface and the corresponding edge length on the developed surface. The final relative edge length difference is

$$E_C = \frac{\sum |L - L'|}{\sum L} \tag{6}$$

where L is the actual length of a curve segment on the original surface before development and L' is the corresponding edge length on the developed surface after development. Suppose curve segment L is contained in curve $C = S(u(t), v(t))$ on surface S, and then L can be calculated by $L = \int_{t_0}^{t_1} \sqrt{EU^2 + 2FUV + GV^2}\,dt$ [3], where $U = \frac{du(t)}{dt}$, $V = \frac{dv(t)}{dt}$, and (t_0, t_1) is the parametric range of segment L on curve C. For the same cause as the surface ʔ, the analytic form of the curve C cannot be determined too. Here we simply approximate L by summing the length of each triangle edge in the facet model,

$$L = \sum_{i=0}^{m-1} L_i \tag{7}$$

where L_i is the length of the *i*th triangle edge in the facet model, and m is the triangle number in the facet model.

Both the area accuracy and the shape accuracy are used as termination criteria for the flattening procedure.

5. Energy Release

The Lagrange Equation is usually applied in an energy model to release energy [10]. The equation is as follows.

$$M\ddot{q} + D\dot{q} + Kq = g_q + f_q \tag{8}$$

where M, D and K are spring mass, damping and stiffness matrices, g_q is the inertial force arising from the dynamic coupling between the local and global degrees of freedom, and f_q is the generalized external force associated with the degrees of freedom of the model.

In our spring-mass system, g_q and f_q are equal to zero and the damping item is ignored. Thus, for our spring-mass system the Lagrange Equations of Motion can be simplified to

$$M\ddot{q} + Kq = 0 \tag{9}$$

where Kq is equal to $-\vec{f}$ of equation (2).

We use Euler's method explicitly to solve equation (9). We assume that the acceleration of mass P_i is constant when Δt is very small. The balance of the whole spring-mass system is composed of the balances for each single node. For each node P_i, equation (9) can be changed to

$$m_i = \frac{\rho}{3}\sum A_k \tag{10}$$

$$\ddot{q}_i(t) = \frac{f_i(t)}{m_i} \tag{11}$$

$$\dot{q}_i(t+\Delta t) = \dot{q}_i(t) + \Delta t \ddot{q}_i(t) \tag{12}$$

$$q_i(t+\Delta t) = q_i(t) + \Delta t \dot{q}_i(t) + \frac{\Delta t^2}{2}\ddot{q}_i(t) \tag{13}$$

where m_i is the mass of P_i, ρ is the area density of the surface, A_k is the model space area of triangle k which contains nodes P_i, $\ddot{q}_i(t)$ is the acceleration of node P_i, $f_i(t)$ is the tensile force on node P_i, $\dot{q}_i(t)$ is the velocity of node P_i and $q_i(t)$ is the position of node P_i at time t.

The area density, ρ, here is not the real density of the surface. In most physics-based models, ρ and C are just scale factors that make the deformation more appropriate [10], e.g. $\rho = \frac{1}{\min\{m_i\}}$ and $C = 0.5$. The area density, ρ, and the spring constant, C, can also be obtained by testing the mechanical properties. To the time step, Δt, usually we use $\Delta t = 0.01$; it always leads to stable result with acceptable speed.

Penalty Function: During surface development, overlap may easily occur. In Figure 2 if P_1 moves to the right side of edge P_2P_3 (where $\vec{V}$ shows the moving direction of P_1), $\Delta P_1P_2P_3$ overlaps other triangles.

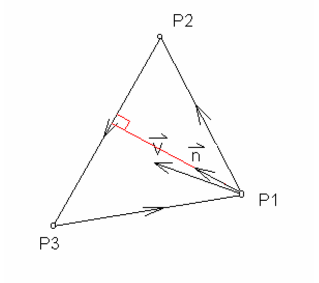


Figure 2 Overlap will happen

To prevent an overlap, a penalty is added to moving masses. Adapting Provot's method (1995) [11], we define the penalty function for mass P as follows.

$$\vec{F}_{penalty} = -\sum_{j=1}^{m} C_{penalty} \left| h_j - h_j^* \right| \cdot \vec{n} \qquad \begin{cases} C_{penalty} = 1 & \left(h_j \le h_j^*\right) \\ C_{penalty} = 0 & \left(h_j > h_j^*\right) \end{cases} \tag{14}$$

where h_j is the current distance from P to its opposite edge on the planar surface (e.g. in Figure 3, from P to edges $\overline{P_1P_2}$, $\overline{P_2P_3}$, $\overline{P_3P_4}$, $\overline{P_4P_5}$, $\overline{P_5P_6}$ and $\overline{P_6P_1}$), h_j^* is the Euclidean distance from Q to its *j*th opposite edge on the original surface, Q is the corresponding vertex of P on the given 3D mesh surface, $\vec{n}$ is a unit vector pointing from P to its opposite edge on the planar surface (as shown in Figure 2), and m is the number of the opposite edges of P.

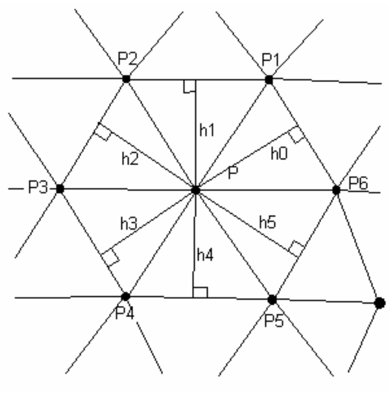


Figure 3 Penalty functions

Penalty function $\vec{F}_{penalty}$ is a vector function. When mass P_i moves to a position that is too close to its opposite edge, the penalty function $\vec{F}_{penalty}$ is applied to the position of P_i to prevent overlapping (as shown in the following equation).

$$q'_{i_t} = q_{i_t} + \vec{F}_{penalty} \tag{15}$$

Scheme of Energy Release: The following iterative algorithm, *EnergyRelease(Φ)*, finds a set of vertex positions P defining a planar mesh $\Phi = (K, P)$ that minimizes the energy function $E(\Phi)$.

Algorithm EnergyRelease(Φ)

Input: A set of nodes, P, in the initial position, $P = \{P_0, P_1, \cdots, P_{n-1}\}$; and n is the number of nodes.
Output: The final positions of P minimize $E(\Phi)$.

```
1.  for ( i = 0 ; i < n ; i = i +1 )
2.     Use Eq. (10) to compute the mass of node $P_i$;
3.  while( ( the relative area difference $E_S$ > the permissible accuracy OR
             the relative edge length difference $E_C$ > the permissible accuracy ) AND
             the variation of $E(\Phi)$ > the permissible percentage $\varepsilon$ AND
```

```
        the number of iterations < the permissible number $N$ )
4.  {
5.          for ( i = 0 ; i < n ; i = i + 1 )
6.          {
7.              Use Eq. (2) to compute the tensile force on node $P_i$ ;
8.              Use Eq. (11, 12 and 13) to compute the new position of node $P_i$ ;
9.              Use Eq. (14 and 15) to compute the penalty function and apply it to node $P_i$ ;
10.             Move $P_i$ to the new position;
11.         }
12.     Use Eq. (4) to compute the new $E_S$ ;
13.     Use Eq. (6) to compute the new $E_C$ ;
14.     Use Eq. (1 and 3) to compute the new $E(\Phi)$ ;
15. }
16. return $\Phi$ ;
```

The variation of $E(\Phi)$ can be computed by $\frac{|E_j(\Phi) - E_{j-1}(\Phi)|}{E_{j-1}(\Phi)}$, where $E_j(\Phi)$ represents the energy of Φ in the *j*th step of iteration. In ***Algorithm*** *EnergyRelease(Φ)*, ε is the permissible percentage to the variation of $E(\Phi)$, usually we use 0.05%. The determination of N will be introduced in the following section.

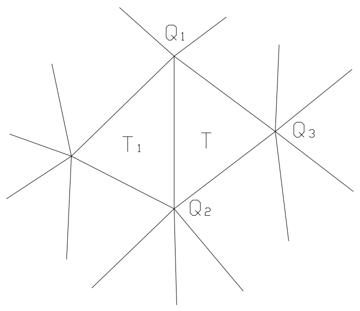
6. Surface Flattening

6.1 Phase One: Triangle Flattening

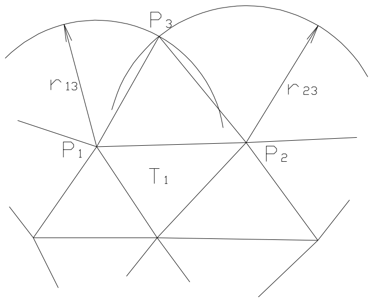
In this phase of surface flattening, triangles of the 3D given mesh will be flattened one by one to obtain the initial 2D pattern.

6.1.1 Triangle Flatten without Energy Release

Unconstrained Triangle Flattening: We assumed that this will occur when one edge (Q_1Q_2) of the triangle (T) has already been flattened while the third node (Q_3) is going to be located on the flattened plan. The position of Q_3 on the 2D flattening, P_3, is found by finding the intersection of two circles. These circles are centered at P_1 and P_2 with radii r_{13} and r_{23} respectively which are the lengths of the appropriate edges on the original 3D surface (the approximate length of r_{13} and r_{23} is the Euclidean distance between two nodes), as shown in Figure 4. To a developable surface, the 2D position of Q_3 determined by triangle T is same with the 2D position of Q_3 determined by other triangles containing Q_3; but to a non-developable surface, they will be different. Thus, constrained triangle flattening can be used.



(a) 3D Triangulated Surface

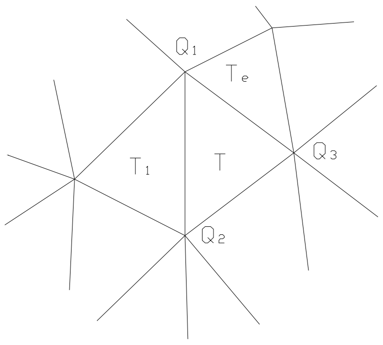


(b) 2D Partial Flattening

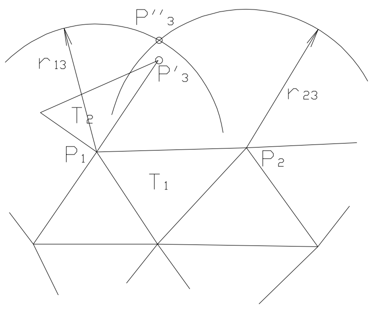
Figure 4 Unconstrained Triangle Flattening

Constrained Triangle Flattening: When a triangle T shares an edge with the previously flattened triangle T_1, a situation can arise whereby another edge of triangle T is shared with another previously flattened triangle T_2. By flattening triangle T_2, the 2D location of Q_3 will be P_3' (as shown in Figure 5b). Then if we flatten the triangle T_1 by the unconstrained triangle flattening method, this will probably provide another location (on the 2D flattening), P_3'', for the node Q_3 (Figure 5). To initially remedy this situation, a mean position is used to resolve this conflict so that a unique location is produced.

The constrained triangle flattening process generates strain energy in the edges, which will lead to overlap error (as shown in Figure 6b).

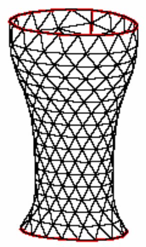


(a) 3D Triangulated Surface

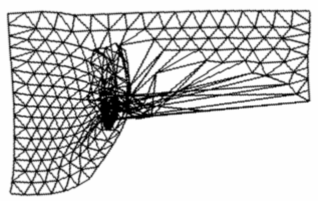


(b) 2D Partial Flattening

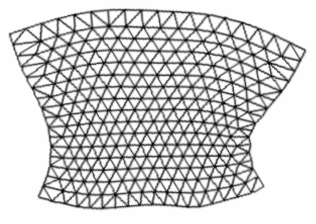
Figure 5 Constrained Triangle Flattening



(a) 3D Triangulated Surface



(b) Without Energy Release



(c) With Energy Release

Figure 6 Initial Triangle Flattening without and with Energy Release

6.1.2 Triangle Flatten with Energy Release

To release the strain energy generated by the averaging process, we build a various spring-mass system Ψ that contains the triangles flattened. After the flattening of each triangle, the various spring-mass system, Ψ, will be altered and will be activated to release the strain energy by the function *EnergyRelease*, which is described in section 5. The algorithm, *InitFlatten*, described below can be used to obtain the initial 2D pattern.

Algorithm InitFlatten

Input: A given 3D triangular mesh surface with triangles $T = \{T_0, \ldots, T_{n-1}\}$; n is the number of triangles.
Output: The initial 2D pattern.

```
1. Build a null spring-mass system $\Psi$, $\Psi = \phi$;
2. for ( i = 0 ; i < n ; i = i +1 ) {
3.    Flatten triangle $T_i$;
4.    Add triangle $T_i$ into $\Psi$;
5.    Call EnergyRelease( $\Psi$ ) with the permissible iteration steps number $N = 50$;
6. }
7. Initial 2D pattern $\Leftarrow$ Position of the various spring-mass system;
8. return the initial 2D pattern;
```

Figure 6a shows a non-developable surface to be flattened. Figure 6b shows the initial 2D pattern obtained by triangle flattening without energy release. It is obvious that overlapping error has occurred. And Figure 6c shows the initial 2D pattern obtained by triangle flattening with energy release. The initial flattening result is acceptable.

6.2 Phase Two: Planar Mesh Deformation

Using the simple iterative algorithm, *EnergyRelease(* Φ *)*, described in section 5 to release energy in subsequent steps, the final 2D pattern can be obtained. Some examples of 3D surface flattening using our model are shown in Figure 7 and 8. The resulting accuracy of each development is shown in Table 1. For the results in Table 1, permissible E_S and E_L accuracy are both selected as 1%, the permissible percentage of energy variation is 0.05%, and the permissible iteration steps is 1000. In phase two, Example I reaches the iteration terminal condition after iterating 11 steps; and Example I reaches the iteration terminal condition after iterating 17 steps

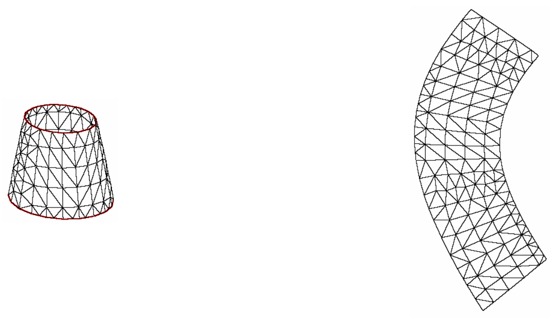

(a) A ruled surface (b) After development

Figure 7 Example I - Conical surface and its 2D pattern

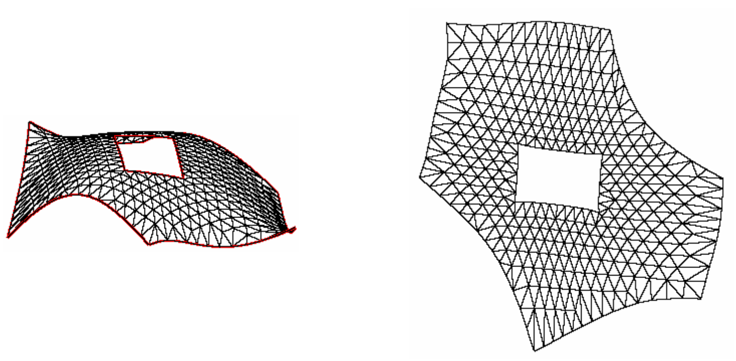

(a) A trimmed surface (b) After development

Figure 8 Example II - Trimmed surface and its 2D pattern

Table 1 Calculation statistics

Example	Number of triangles	Time of phase one	Time of phase two	Area error	Shape error
I	149	10s	6s	0.092%	0.174%
II	381	37s	19s	0.784%	1.13%

*Using PIII 667 MHz PC.

6.3 Additional Phase: Surface Cutting

Since some complex surfaces are difficult to develop, at the end of the deformation, we frequently experienced errors in the resulting 2D patterns. Cutting operations can be applied to reduce development errors. Here, an interpolation function is used to compute the energy on the developed surface and then a reference cutting line is determined using an elastic deformation energy distribution map represented by a color graph.

Interpolation Function: When energy values for the three nodes of a triangle are known, an interpolation function can be built to get the energy value for any point in the triangle. After assuming the energy value of any point $p(x, y)$ in the triangle is a linear function of its coordinate, the energy value for any point can be computed by

$$\left.\begin{aligned} e &= \sum N_k e_k \\ N_k &= \frac{1}{2A}(a_k + b_k x + c_k y) \end{aligned}\right\} k = i, j, m \tag{16}$$

where

$$\begin{array}{lll} a_i = x_j y_m - x_m y_j & b_i = y_j - y_m & c_i = x_m - x_j \\ a_j = x_m y_i - x_i y_m & b_j = y_m - y_i & c_j = x_i - x_m \\ a_m = x_i y_j - x_j y_i & b_m = y_i - y_j & c_m = x_j - x_i \end{array}$$

e_i, e_j, and e_m are the given energy values for the three nodes of $\Delta P_i P_j P_m$, N_k is the shape-function of a triangle element, (x, y) is the coordinate of any point in the triangle, and A is the area of the triangle. Thus the energy value for any point in a triangle can be calculated.

Using equation (16), an elastic deformation energy distribution map can be obtained, as shown in Figure 9. Red represents regions with the highest energy level, and blue represents regions with the lowest energy level.

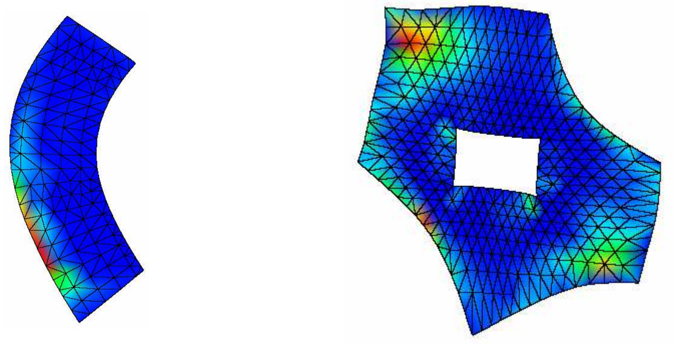
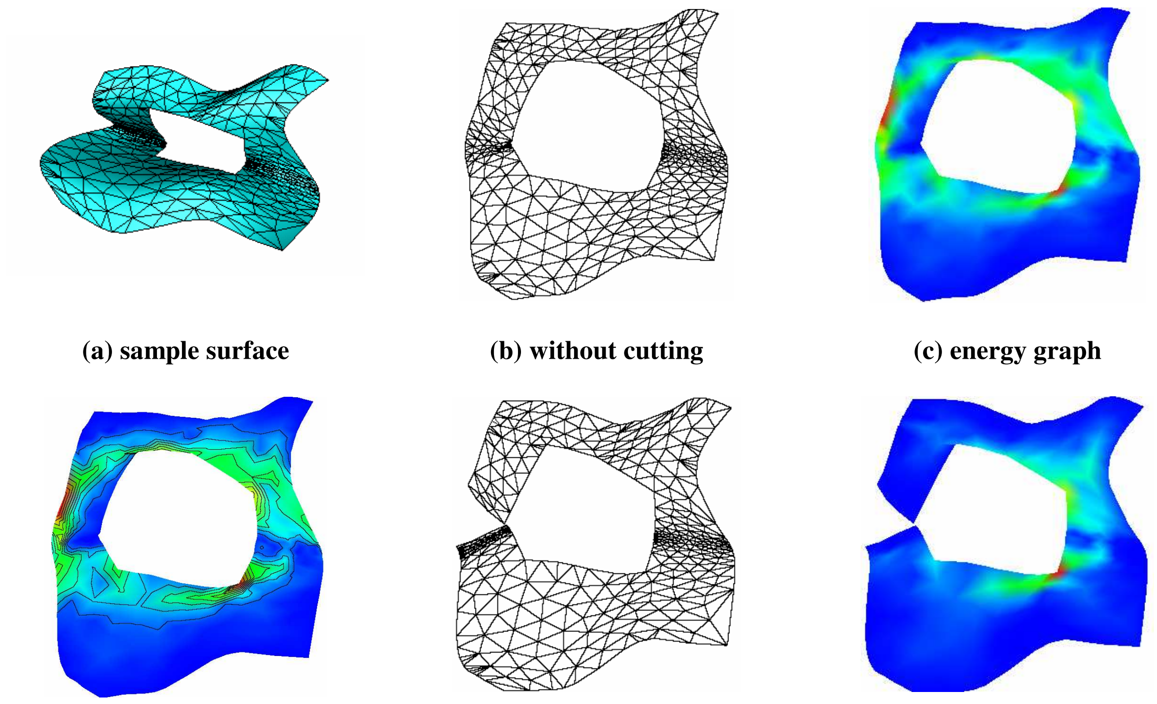

(a) Ruled Surface (b) Trimmed Surface

Figure 9 Energy color graphs

Surface Cutting: From the elastic deformation energy distribution map, cutting lines can be defined. Many methods can be used to search for an acceptable cutting direction. Here, a gradient method is used. First, using equation (16), isolines of the energy distribution are obtained. Second, we find the maximum energy point on the surface. Then, a crest line can be found (a crest line is a curve which passes through the maximum energy point along which the descent of energy is slowest). In our algorithm, the crest line is taken as a reference cutting line. After cutting, some additional deformation can be applied. Deformation and cutting can be done repeatedly.

Figure 10a shows a surface to be developed and cut. Figure 10b, 10c, 10d, 10e, and 10f show resulting developed surfaces with and without cutting. Table 2 shows the accuracy of the results for the two cases. For the results in Table 2, permissible E_S and E_L accuracy are both selected to be 1%, and the permissible percentage of energy variation is 0.05%, and the permissible iteration steps is 1000. From the statistics in Table 2, we found that both the area error and the shape error are larger than 1% even after cutting (the terminal condition of iteration is the permissible iteration steps in this example). Thus, the surface needs more cutting to reduce the elastic deformation energy.



(a) sample surface **(b) without cutting** **(c) energy graph**

(d) isolines representation **(e) with cutting** **(f) energy graph**

Figure 10 Surface development without cutting and with cutting

Table 2 Development statistics

Status	Area error	Shape error
Without Cutting	8.060%	9.786%
With Cutting	1.257%	2.386%

7. Conclusion and Further Research

This paper presents an energy model used to flatten a given 3D mesh surface into a 2D pattern. The accuracy of local areas of the surface can be easily controlled. An interpolation function is used to show the energy distribution on the developed surface. Cutting lines can be obtained from the energy distribution map and cutting can be used to reduce the error of the flattened surface.

Our surface flattening algorithm has the following advantages:

- Complex trimmed surfaces can be developed.
- Accuracy of local surface regions can be controlled by changing the constant "C" (Equations (1) and (2)) during the procedure.

- Distribution of surface elastic deformation energy can be used to find a reference cutting line for the surface.
- Since triangular mesh can represent all manifold surfaces, our approach provides a generic solution for the surface flattening problem.

For energy release, most of our physical parameters such as system forces, masses and elastic deformation energies are defined by their corresponding geometric parameters. Our model is a spring-mass model in terms of geometry. Future research can be done to incorporate material properties into our model. Different materials will lead to different results. This method can be used in other applications as well; e.g., in sheet-metal manufacture, our algorithm can be used to solve the problem of developing patterns for ductile material.

Acknowledgement

The authors would like to acknowledge the comments made by Dr. Fan Jin, Mr. Greg Smith, Dr. Wu Zhang, Dr. Wang Qifu, Dr. Deng Jianshong and the support of the CAD/CAM Facility at the Hong Kong University of Science and Technology.

References

[1] Fan J., Wang Q.F., Chen S.F., Yuen M.M.F. and Chan C.C., A spring-mass model-based approach for warping cloth patterns on 3D objects, The Journal of Visualization and Computer Animation, Vol. 9 No. 4, October/December 1998, pp. 215-227.

[2] Hsu W.M., Hughes J.F. and Kaufman H., Direct manipulation of free-form deformations, Computer Graphics (ACM), vol.26, no.2, July 1992, pp.177-84.

[3] Carmo M.P.D., Differential Geometry of Curves and Surfaces, Prentice Hall, Englewood Cliffs, NJ, U.S.A, 1976.

[4] Shimada T. and Tada Y., Approximate transformation of an arbitrary curved surface into a plane using dynamic programming, Computer Aided Design, 1991, 23(2):155-159.

[5] Parida, L., Mudur, S. P., Constraint-satisfying planar development of complex surfaces, Computer-Aided Design, 1993, 25(4):225-232.

[6] Ma S.D. and Lin H., Optimal texture mapping, Proceeding of Eurographics'88, 1988, 421-428.

[7] Maillot J., Yahia H. and Verroust A., Interactive texture mapping, SIGGRAPH 93, Computer Graphics proceedings, 1993, 27-34.

[8] Sun M. and Fiume E., A technique for constructing developable surfaces, Graphics Interface, May 1996, pp. 176-185.

[9] McCartney J., Hinds B.K. and Seow B.L., The flattening of triangulated surfaces incorporating darts and gussets, Computer-Aided Design, vol.31, no.4, April 1999, pp.249-60. Publisher: Elsevier, UK.

[10] Metaxas D.N., Physics-based Deformable Models, Kluwer Academic Publishers, 1997.

[11] Provot X., Deformation constraints in a mass-spring model to describe rigid cloth behavior, Proceedings Graphics Interface '95, Canadian Inf. Process. Soc. 1995, pp.147-54, Toronto, Ont., Canada.

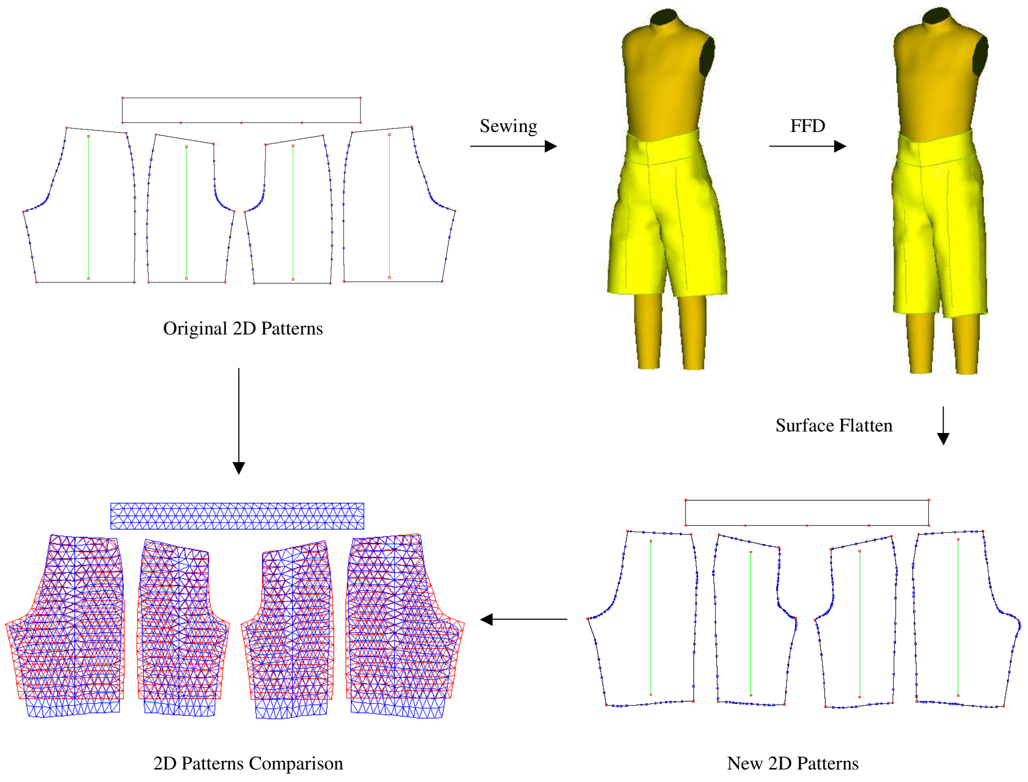


Figure 11 Application of Surface Flattening Technology in Three-dimensional Computer Aided Garment Design