Feature based 3D garment design through 2D sketches

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

Department of Mechanical Engineering, Hong Kong University of Science and Technology,

Clear Water Bay, Kowloon, Hong Kong

Abstract

This paper presents a new approach for intuitively modeling a 3D garment around a 3D human model by 2D sketches input. Our approach is feature based – every human model has pre-defined features, and the constructed garments are related to the features on human models. Firstly, a feature template for creating a customized 3D garment is defined according to the features on a human model; secondly, the profiles of the 3D garment are specified through 2D sketches; finally, a smooth mesh surface interpolating the specified profiles is constructed by a modified variational subdivision scheme. The result mesh surface can be cut and flattened into 2D patterns to be manufactured. Our approach provides a 3D design tool to create garment patterns directly in the 3D space through 2D strokes, which is a characteristic not available in other computer aided garment design systems. The constructed garment patterns are related to the features on a human model, so the patterns can be regenerated automatically when creating the same style of garment for other human models. Our technique can greatly improve the efficiency and the quality of pattern making in the garment industry.

Keywords: sketched input, 2D strokes, 3D design, computer-aided design, garment industry.

1. Overview

The work presented in this paper is part of the project of building a 3D sketch-based apparel product development platform, whose purpose is to help apparel enterprises to stay ahead of current technology innovations promoted by major retailers and manufacturers. At present, the garment industry uses two-dimensional CAD tools. It is expected that three-dimensional tools might be used for garment design to improve the efficiency of pattern generation and for more attractive design presentation. Studies have shown that pattern designers wish to design patterns directly on a 3D human model. Currently, alternative techniques [1, 2] focus on the simulation of a 3D dressing result from 2D patterns input, but not on 3D design. Thus, a design platform that allows the pattern designers to design and modify cloth pattern prototypes in the 3D space is imperative. It has been estimated that most designers still prefer to express their creative design ideas through 2D sketches

* Corresponding Author: Charlie C. L. Wang; E-mail: wangcl@ust.hk or mewangcl@hotmail.com; Tel: (852) 2358-8095

during the conceptual design procedure. Thus, it is important for a computer-aided design system to allow sketched input. In this paper, we present a new approach for modeling a 3D garment around a 3D feature human model through 2D sketches. Our method is a feature-based approach. Therefore, the 3D garment can be automatically regenerated on another feature human model.

The flow chart of our approach is shown in Fig. 1. Three kinds of input can be used to construct a feature human model: 1) a laser scanner [3]; 2) a size chart [4]; and 3) photographs [5]. Garments of different styles are stored as different garment feature templates. Each garment feature template is represented by a coarse triangular mesh M^f, in which every node is related to features on a human model. The 3D profiles of a garment are specified through 2D sketches. The profiles are attached to the edges of M^f; and they should be interpolated in the following mesh refinement process using the modified variational subdivision scheme. The constructed surface is a 3D garment prototype, which can be cut and flattened [6] into 2D patterns to be manufactured or to be modified in a further precise design. Thus, the whole approach consists of three parts: 1) construction of garment feature templates; 2) 3D profile specification using 2D strokes; and 3) construction of garment surfaces interpolating the specified 3D profiles. They are respectively described in sections 3, 4 and 5 in detail.

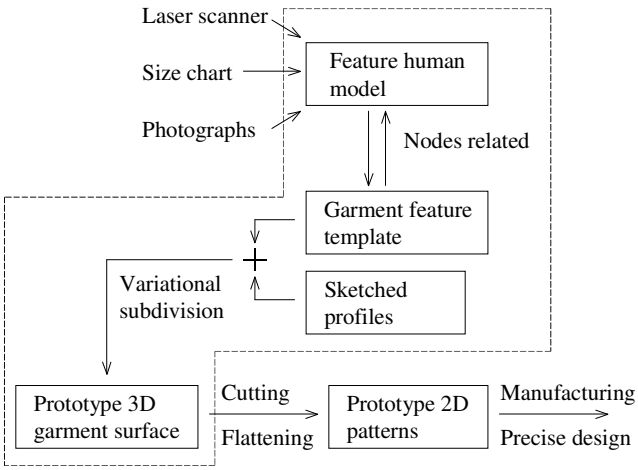


Fig. 1 Flow chart

2. Related Work

In the past ten years, mesh-processing techniques, such as mesh simplification and mesh fairing, have been improved significantly. Several algorithms have been formulated to simplify triangular mesh surfaces [7, 8, 9]. One approach [7] is to simplify a mesh surface by iteratively selecting a vertex for removal, removing all adjacent faces and re-triangulating the resulting hole. Another approach [8] iteratively contracts vertex pairs to

simplify models and maintain surface error approximations using quadric matrices. Statistical measurement is used in [9] to guide the mesh simplification through a faces constriction process. We implement the algorithm [8] to obtain a topological graph from a refined mesh surface with feature nodes defined. Mesh fairing is used to enhance the smoothness of a triangulated surface by removing high frequency noises [10, 11, 12, 13, 14]. A signal processing approach [10] is proposed for solving the problem of fairing arbitrary topology surface triangulations. The signal processing approach is based on defining a suitable generalization of frequency to the case of meshes with arbitrary connectivity. Related to the signal processing approach, Kobbelt et al. (1998) [11] proposed a similar discretization of Laplacian, the umbrella-algorithm. It is a linear scheme; and the approaches [12, 13, 14] lead to nonlinear faring. Since our approach expects to obtain the result in "real time", we choose the umbrella-algorithm to perform discrete fairing in our modified variational subdivision scheme.

Apart from fundamental mesh processing algorithms, many new freeform modeling approaches have also been developed. The SKETCH system [15] rapidly constructs an approximate shape via direct mark based interaction. The Teddy system [16] constructed a rounded freeform mesh model by finding the chordal-axis of user input 2D closed stroke to build a smooth surface around the axis. Other approaches construct mesh surfaces by the use of implicit surfaces [17, 18]. Suzuki et al. [19] presented a 3D mesh-dragging method for intuitive, efficient geometric modeling of free-form polygonal models; this method is based on an adaptive remeshing procedure. With their method, the user can drag a part of a triangular mesh and change its position and orientation. Other interactive modeling researches were proposed for the multi-resolution presentation of models: Zorin et al. (1997) [20] built a scalable interactive multi-resolution editing system based on mesh refinement and coarsification algorithms; Khodakovsky and Schroder (1999) [21] developed an algorithm based on Zorin's approach that can modify the fine level shape of a surface. In this paper, we develop a feature-based approach to construct 3D objects through 2D sketches; our method is also based on mesh refinement and coarsification. However, the semantic features and user specified profiles are preserved to influence the mesh construction process.

There are two approaches to construct smooth mesh surfaces interpolating given curves: the combined subdivision scheme [22, 23] and the variational subdivision scheme [24]. In our approach, not all the edges in the initial coarse mesh have interpolating curve; this does not fit the initial condition of the combined subdivision scheme. Thus, we choose the variational subdivision scheme. The surface constructed in our approach is around a 3D human model, so the collision between the constructed surface and the human model should be prevented during the surface construction. Here, we modify the variational subdivision scheme by

integrating the collision avoidance. In order to achieve an efficient scheme, we use the image-based approach [25] to detect the collision. To prevent shrinking during the refinement, we applied the modified Butterfly subdivision scheme [26] to the free boundaries in our modified variational subdivision scheme.

3. Garment Feature Template

Garments of different styles are stored as different garment feature templates in our system. Each garment feature template is represented by a triangular mesh M^f. We follow the notation in [27] to define our feature template mesh: a mesh M^f is a pair (K^f, V^f), where K^f is a simplicial complex specifying the connectivity of the mesh simplices (the adjacency of the vertices, edges, and faces; in other words, the topological graph of M^f), and $V^f = \{\vec{v}_1, \cdots, \vec{v}_m\}$ is the set of vertices defining the shape of the mesh in $\Re^3$ (each $\vec{v}_i \in V^f$ is called a feature node on the garment feature template). In our approach, the position of $\vec{v}_i$ is related to the features on a human model. Thus, when we apply the same template on different human models, we obtain triangular meshes of different shapes (see Fig. 2).

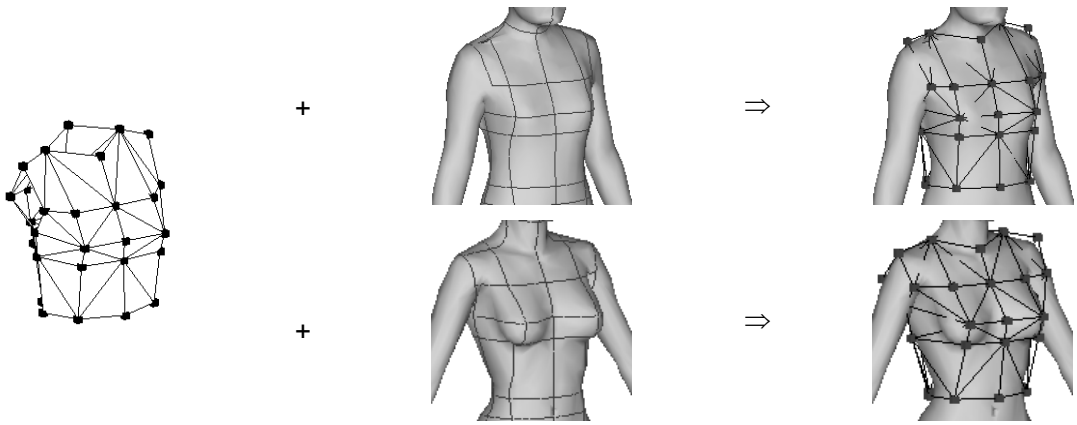


Fig. 2 Different shapes of the same template on different human models

Since our approach is feature based, we firstly give human models with feature lines and points defined – called feature human models. In the following feature node encoding process, the relationship between the position of each feature node $\vec{v}_i \in V^f$ and the features on a human model are defined. The topological graph K^f of M^f can be manually defined, or be automatically determined by a mesh simplification process. The mesh simplification process preserves the encoded vertices.

Feature human model

Before discussing the technique of constructing garment feature templates, we should prepare feature human models. A feature human model can be built from a laser scanner, a size chart, or photographs (Fig. 3). Scanned data from a laser provides the explicit 3D positions from which a 3D model can be reconstructed [28].

In order to create a feature model of an object from a point cloud, the embedded features are recognized; and the features also benefit the surface construction process [3]. Free-form deformation (FFD) technique supports our system to generate a feature human model from a size chart [4]. Here, the feature-based NURBS FFD is used [29]. Since the new human model is deformed from a template human model, the pre-defined semantic feature lines and points on the template human model are maintained after the deformation. The third method involves the construction of a feature human model from two photographs of a human body in two orthogonal views [5]. The view-dependent deformation can also maintain the pre-defined semantic feature lines and points on the template human model.

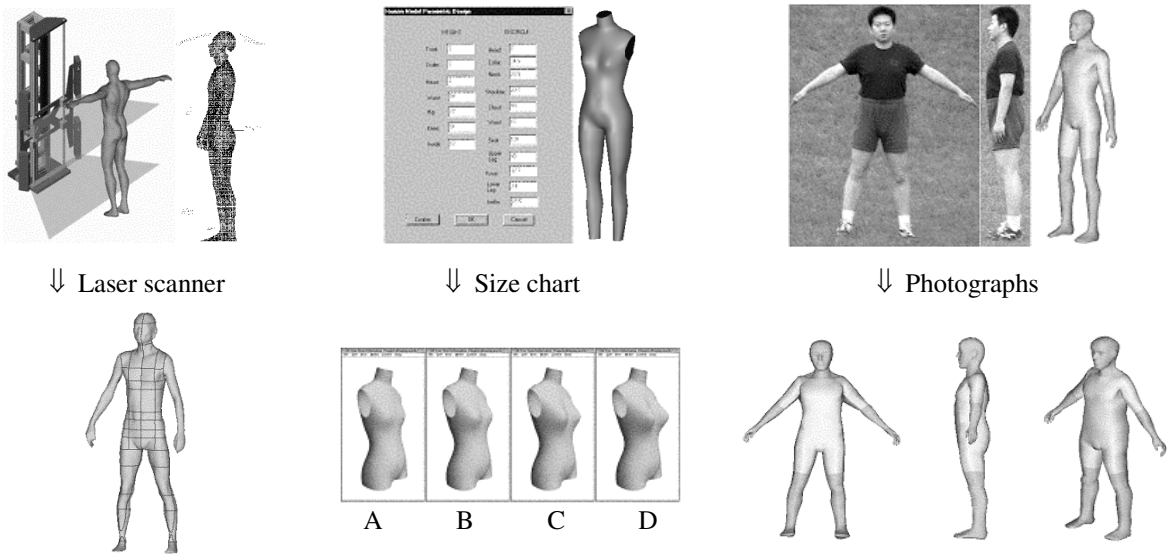
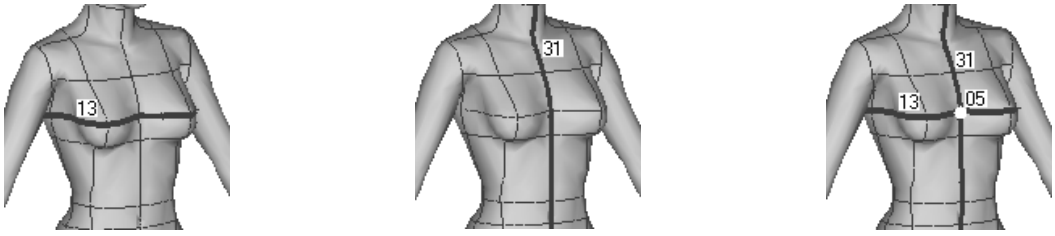


Fig. 3 Feature human model construction

In our semantic feature representation, each semantic feature curve consists of a sorted set of line segments lying on the mesh surface of a human model. The line segments that belong to one semantic feature curve have the same feature ID number. Each semantic feature point is an intersection point of two semantic feature curves, and every semantic feature point has its own ID number. For example, in Fig. 4a, the bold line segments with ID number 13 represents the chest feature curve; in Fig. 4b, the bold line segments with ID number 31 represents the center-front feature curve; and in Fig. 4c, the white point is the feature point determined by these two feature curves, the ID number of the feature point is 05.



(a) **(b)** **(c)**

Fig. 4 Feature lines and points on a human model

Feature nodes encoding

As we mentioned at the beginning of section 3, the position of $\vec{v}_i \in V^f$ is related to the features on a human model H. We build the relationship between a feature node $\vec{v}_i$ in the garment template and a feature point q_j on H, where j is the ID number of point q_j. The position $\vec{p}(q_j)$ of q_j is determined by the intersection of two feature curves f_1 and f_2. Since the feature point q_j lies on f_1 and f_2, the unit tangent vectors of feature lines f_1 and f_2 at q_j – $\vec{t}(f_1)$ and $\vec{t}(f_2)$ is easy to compute; and the unit normal vector $\vec{n}(q_j)$ of the surface of H at q_j can also be calculated. The vectors $\vec{t}(f_1)$, $\vec{t}(f_2)$, and $\vec{n}(q_j)$ form a local coordinate frame as shown in Fig. 5. The three vectors may not be perpendicular to each other, but at least $\left(\vec{t}(f_1)\,\vec{t}(f_2)\,\vec{n}(q_j)\right) \neq 0$. Thus, the position $\vec{p}(\vec{v}_i)$ of any vertex $\vec{v}_i \in V^f$ can be represented by

$$\vec{p}(\vec{v}_i) = \vec{p}(q_j) + \alpha_i \vec{n}(q_j) + \beta_i \vec{t}(f_1) + \gamma_i \vec{t}(f_2) \tag{1}$$

Each vertex $\vec{v}_i$ consists of four elements: $(j, \alpha_i, \beta_i, \gamma_i)$. The encoding process which relates one feature node $\vec{v}_i \in V^f$ to a feature point q_j on H is actually a process to determine the four elements of vertex $\vec{v}_i$.

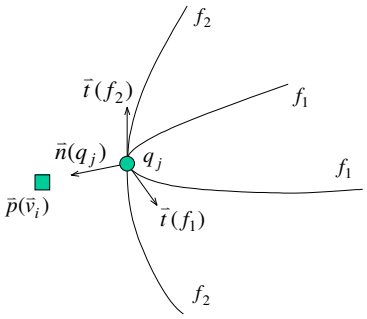


Fig. 5 Local frame on feature point q_j

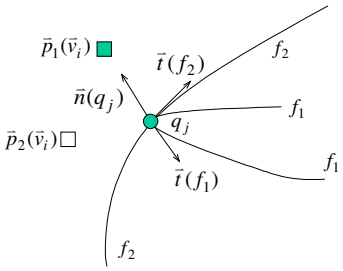


Fig. 6 New position of $\vec{v}_i$ determined

Equation (1) defines the relationship between the positions of $\vec{v}_i$ and q_j. Another popular form used to define the relationship between positions of two points is

$$\vec{p}(\vec{v}_i) = \vec{p}(q_j) + \vec{d} \tag{2}$$

by which the position of vertex $\vec{v}_i$ is also determined by four elements: (j, d_x, d_y, d_z). When a new human model H′ is used, the new position of each vertex $\vec{v}_i$ in Fig. 5 can be easily calculated by (1) or (2). As shown in Fig. 6, $\vec{p}_1(\vec{v}_i)$ is the new position determined by equation (1), and $\vec{p}_2(\vec{v}_i)$ is the new position determined by equation (2). It was found that the orientation of the garment feature node related to a human model is not

guaranteed when using equation (2), but equation (1) strongly preserves the orientation. This is an important factor in the garment industry, so we choose $(j, \alpha_i, \beta_i, \gamma_i)$ as the four elements of $\vec{v}_i$.

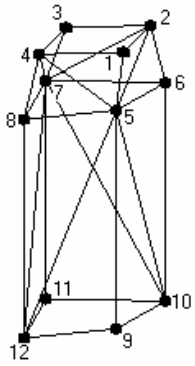
Determine topological graph

The garment feature template is represented by $M^f = (K^f, V^f)$, so after the positions of feature nodes in the set $V^f = \{\vec{v}_1, \cdots, \vec{v}_m\}$ are defined, we determine the topological graph K^f of M^f. The topological graph K^f consists of an edge table and a face table, whose structures are shown in Table 1. In the edge table, the attached points list is the set of 3D points defining the shape of an edge profile in $\Re^3$ (described in detail in section 4); and in the face table, the direction value is +1 when an edge is in the anti-clockwise direction in a triangular face, and is -1 when an edge is in the clockwise direction (the direction of one edge is pointing from its start node to its end node). One example of the edge table and the face table in a topological graph is shown in Fig. 7.

Table 1 Structures of edge table and face table

Edge Table				
ID	Start node ID	End node ID	Number of attached points	List of attached points

Face Table						
ID	1st edge ID	2nd edge ID	3rd edge ID	1st edge direction	2nd edge direction	3rd edge direction



Edge Table				
1	1	2	0	NULL
2	2	3	0	NULL
3	3	4	0	NULL
4	4	1	0	NULL
⋮				
25	10	5	0	NULL
26	10	7	0	NULL
27	12	7	0	NULL
28	12	5	0	NULL

Face Table						
1	1	13	21	-1	+1	+1
2	5	14	21	+1	-1	-1
3	6	22	14	+1	+1	+1
4	15	2	22	-1	-1	-1
⋮						
13	27	19	11	+1	+1	+1
14	20	7	27	-1	-1	-1
15	20	28	8	+1	+1	-1
16	12	17	28	+1	-1	-1

Fig. 7 Example of edge table and face table in a topological graph

To a topological graph with few edges and faces, the edge table and the face table can be filled manually. However, when K^f has many edges and faces (e.g., $K^f \in M^f$ shown in Fig. 2), it is a boring job to fill them by hand. An approach that automates this process is to construct K^f from a similar mesh surface M' by simplifying M' with the encoded vertices preserved. The similar mesh surface M' to M^f is cut from a human model Φ represented by a triangular mesh (Fig. 8a and Fig. 8b). The feature nodes on M' are then specified interactively; the result is shown in Fig. 8c. In the following, vertex pairs are contracted iteratively

(Fig. 8d) while maintaining surface error approximations using quadric matrices [8]. When one vertex is an encoded feature node in a contracting pair, the position of the encoded node is preserved; and when two vertices are both encoded feature nodes, the contraction is prevented. The iteration stops when all pairs are with two encoded feature nodes. M'' is the result of the iteration procedure. The topological graph $K'' \in M''$ is the determined K^f from M' (as shown in Fig. 8e).

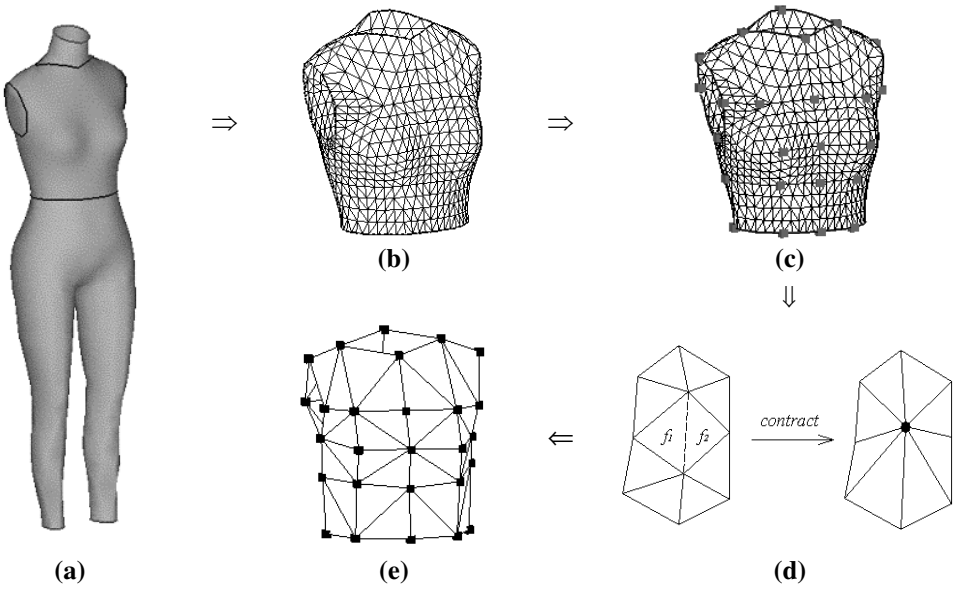


Fig. 8 Semi-automatic approach of topological graph construction

4. Profile Specification through 2D Sketches

3D profiles of a piece of garment are specified through 2D sketches. The profiles are attached to the edges of M^f (e.g., Fig. 9a and 9b); and they are interpolated in the following mesh refinement process. While sketching profiles, the easing space between a garment and a human model is given (Fig. 9c); and the positions of feature nodes are also adjusted. After introducing how to store the profile, we present the profile sketching method in detail.

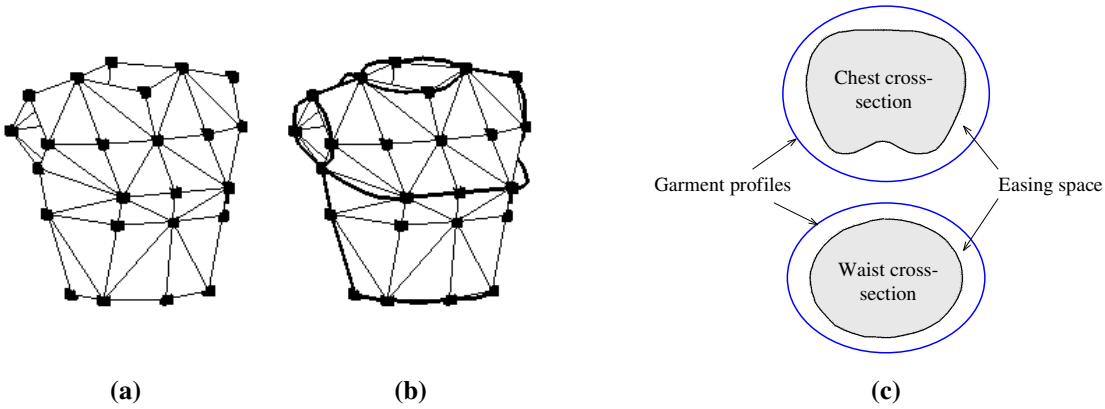


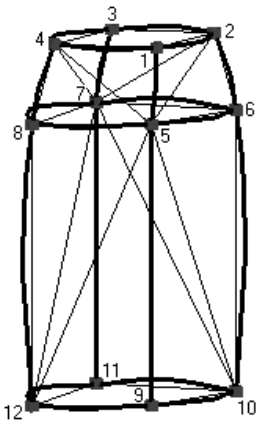
Fig. 9 Profiles attached on the garment template

Store profile

As we discussed in section 3.3, the shape of the edge profile in $\Re^3$ is stored as an attached points list $Q_{i,\,i=0\ldots n-1}$ in the edge table, where n is the number of attached points. The first attached point Q_0 has the same position as the start node of the edge, and the last attached point Q_{n-1} has the same position as the end node of the edge. We can compute a parametric curve $\vec{c}(u)$ passing all these points to represent the shape of the edge profile, where every attached point has its parameter $u_i \in [0, 1]$, $\vec{c}(0)$ passes point Q_0, and $\vec{c}(1)$ passes Q_{n-1}. The parameter u_i is determined by the lengths of line segments [30],

$$u_i = \begin{cases} 0 & (i = 0) \\ u_{i-1} + \dfrac{|Q_i - Q_{i-1}|}{\sum_{j=1}^{n-1} |Q_j - Q_{j-1}|} & (i \neq 0) \end{cases}. \tag{3}$$

An example edge table with profiles defined and its related object are shown in Fig. 10. In order to reduce the computing time of display, we store the absolute coordinates of the attached points.



Edge Table				
1	1	2	17	(13.7, 0.0, 0.1); (13.7, 0.6, 0.1); (13.7, 1.2, 0.1); …
2	2	3	21	(5.0, 13.2, 0.1); (-1.9, 10.6, 0.1); (-2.6, 10.0, 0.1); …
3	3	4	23	(-3.8, 0.0, 0.1); (-3.7, -1.2, 0.1); (-3.7, -1.7, 0.1); …
4	4	1	18	(5.0, -13.1, 0.1); (7.4, -11.0, 0.1); (8.8, -10.4, 0.1); …
⋮				
18	6	10	4	(4.9, 16.8, -12.1); (4.9, 17.9, -20.2); (4.9, 18.4, -3.8); (4.9, 16.8, -55.6)
19	7	11	2	(-9.8, 0.0, -12.1); (-9.8, 0.0, -55.6)
20	8	12	4	(4.9, -16.8, -12.1); (4.9, -17.9, -20.2); (4.9, -18.5, -3.8); (4.9, -16.8, -55.6)
⋮				
27	12	7	0	NULL
28	12	5	0	NULL

Fig. 10 An example edge table with profiles defined

Move node

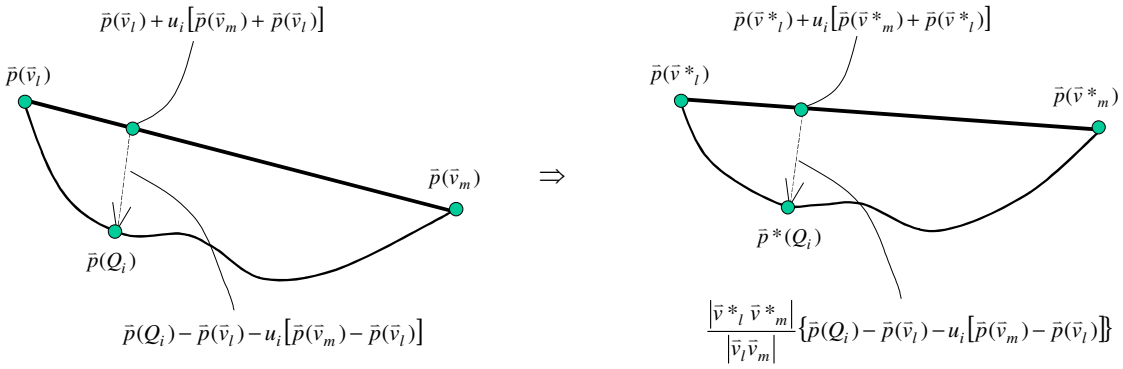
A 2D sketched circle is used to select the feature nodes to move (Fig. 11). After the user strokes the "circle", we use an ellipse Γ to approximate the user input stroke that consists of a set of points $\Phi = \{\varphi_1, \ldots, \varphi_n\}, \varphi_i \in \Re^2 \quad (n > 2)$ by the *Least Square* method [31]. A visible feature node $\vec{v}^*_i$ on M^f inside Γ is selected to move.

After that, another stroke $\Phi^* = \{\varphi^*_1, \ldots, \varphi^*_n\}, \varphi^*_i \in \Re^2$ is input to specify the moving direction and the moving distance. The angles $\theta_{\tau,\,\tau=x,y,z}$ between $\overrightarrow{\varphi^*_n \varphi^*_1}$ and the *x*-, *y*-, *z*- axes are used to select the direction

to move. The probability of the three axes – P_τ is $P_\tau \propto \frac{1}{|\theta_\tau|}$, $\tau = x, y, z$, so we move the selected feature node along the axis with the maximum probability. The new position of $\vec{v}^*_i$ is

$$\vec{p}^*(\vec{v}^*_i) = \vec{p}(\vec{v}^*_i) + \lambda \left| (\overrightarrow{\varphi^*_n \varphi^*_1})_\tau \right| \vec{n}_\tau \tag{4}$$

where $(\overrightarrow{\varphi^*_n \varphi^*_1})_\tau$ is the projection of $\overrightarrow{\varphi^*_n \varphi^*_1}$ on the axis τ in the screen plane, λ is the scale factor between the screen space and the spatial space, and $\vec{n}_\tau \in \Re^3$ is the unit vector of axis τ.



(a) before moving **(b) after moving**

Fig. 11 Shift the attached points

When the position of feature node $\vec{v}^*_i$ is moved, we should also adjust the profile defined on its related edges. If an edge is moved from $\vec{v}_l\vec{v}_m$ to $\vec{v}^*_l\,\vec{v}^*_m$, the new positions of points $Q_{i,\,i=0\ldots n-1}$ attached to it are shifted by scaling the vector between them and $\vec{v}_l\vec{v}_m$. The idea is shown in Fig. 11, where the vector between Q_i and $\vec{v}_l\vec{v}_m$ is scaled by $\frac{|\vec{v}^*_l\,\vec{v}^*_m|}{|\vec{v}_l\vec{v}_m|}$. The formulation to compute the new positions of Q_i is shown below.

$$\vec{p}^*(Q_i) = \frac{|\vec{v}^*_l\,\vec{v}^*_m|}{|\vec{v}_l\vec{v}_m|}\{\vec{p}(Q_i) - \vec{p}(\vec{v}_l) - u_i[\vec{p}(\vec{v}_m) - \vec{p}(\vec{v}_l)]\} + \vec{p}(\vec{v}^*_l) + u_i[(\vec{p}(\vec{v}^*_m) - \vec{p}(\vec{v}^*_l)] \tag{5}$$

where u_i is the parameter defined in equation (3). After all the new positions of $Q_{i,\,i=0\ldots n-1}$ are obtained. The new parametric curve $\vec{c}^*(u)$ passing all these new points needs to be recomputed. An example of moving a feature node is shown in Fig. 12. The original M^f and its related profile is shown in Fig. 12a and 12b; we select one node to move via a stroked "circle" in Fig. 12c, and then, we specify the moving direction and distance by a short stroke near the selected node in Fig. 12d. Finally, the modified M^f and its profile are shown in Fig. 12e and 12f.

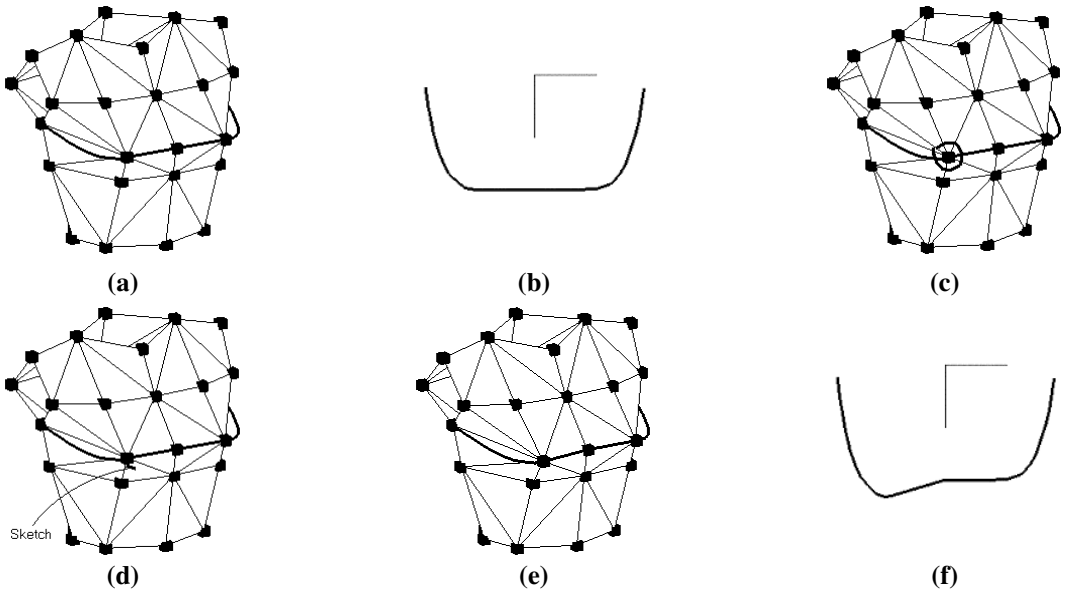


Fig. 12 Example of moving a feature node

Specifying a profile through a 2D stroke

After determining the positions of feature nodes, we will solve the problem of how to specify a 3D edge profile though a 2D stroke in the section. The profile specification through a 2D stroke Ψ has two major issues: 1) whose profiles are specified by Ψ, and 2) how to determine the positions of the 3D attached points by Ψ. After solving these two problems, the 2D stroke Ψ is converted to the profiles specified on some edges.

For a given edge $\vec{v}_l\vec{v}_m$ with endpoints $\vec{v}_l$ and $\vec{v}_m$, it is selected by the stroke Ψ if $\vec{v}_l\vec{v}_m$ is visible and the distances between $\zeta(\vec{v}_l)$, $\zeta(\vec{v}_m)$ and Ψ is less than ε, where $\zeta:\Re^3 \to \Re^2$ is the map that sends spatial point $\vec{v}_i \in \Re^3$ to screen point $\psi_i \in \Re^2$, and ε is a small tolerance value (e.g., $\varepsilon = 4$ pixels). The bold black edges in Fig. 13a and 13d are the edges selected by the bold gray stroke.

In the following, we determine a plane to project the points $\psi_i \in \Psi$ to convert them into $\psi^*_i \in \Re^3$. The plane approximately passes through the selected edges (Fig. 13b). Since we know the positions of the endpoints of the selected edges, we use the *Least Square* method [31] to obtain the approximate plane of these points. Two cases can cause the *Least Square* method to fail to determine the projection plane. One case is when all the selected edges are coplanar, so we get the accurate plane passing through them; another case is when all the selected edges are collinear or only one edge is selected (Fig. 13e), then we use the plane that bisects the dihedral angle along the chosen edges to project the points $\psi_i \in \Psi$. In this way, the sketched profile faces towards the camera as much as possible.

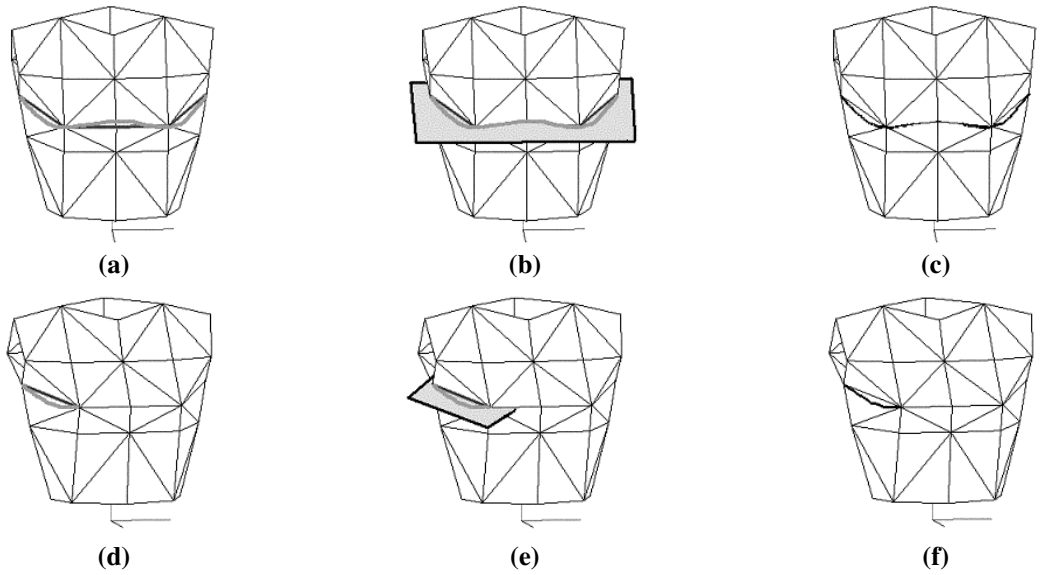


Fig. 13 Specify profile through a 2D stroke

After all $\psi_i \in \Psi$ are converted to $\psi^*_i \in \Psi^*$ in $\Re^3$, we separate them into intervals and store the points in one interval in its related edge. To a selected edge $\vec{v}_l\vec{v}_m$, we search the closest point ψ^*_l to $\vec{v}_l$ in Ψ^*, and the closest point ψ^*_m to $\vec{v}_m$ in Ψ^*. Then, the points $\psi^*_{j,\ j=l,\ldots,m}$ are stored as the attached points list in $\vec{v}_l\vec{v}_m$. We define $K(\vec{v}_i)$ as the number of selected edges adjacent to $\vec{v}_i$, if $K(\vec{v}_l)=1$ and $K(\vec{v}_m)=1$, fix the positions of $\vec{v}_l$ and $\vec{v}_m$, and shift the position of point $\psi^*_{j,\ j=l,\ldots,m}$ by the following equation to let $\vec{p}^*(\psi^*_1)=\vec{v}_l$ and $\vec{p}^*(\psi^*_m)=\vec{v}_m$. The idea is the same as equation (5).

$$\vec{p}^*(\psi^*_j)=\frac{|\vec{v}_l\vec{v}_m|}{|\psi^*_l\,\psi^*_m|}\left\{\vec{p}(\psi^*_j)-\vec{p}(\psi^*_l)-u_j\left[\vec{p}(\psi^*_m)-\vec{p}(\psi^*_l)\right]\right\}+\vec{p}(\vec{v}_l)+u_j[(\vec{p}(\vec{v}_m)-\vec{p}(\vec{v}_l)] \qquad (6)$$

where u_j is defined in the same way as (3) as

$$u_j=\begin{cases} 0 & (j=l) \\ u_{j-1}+\dfrac{\left|\psi^*_j-\psi^*_{j-1}\right|}{\sum_{j=1+1}^{m}\left|\psi^*_j-\psi^*_{j-1}\right|} & (j\neq l) \end{cases}; \qquad (7)$$

if $K(\vec{v}_l)>1$ and $K(\vec{v}_m)>1$, change the positions of $\vec{v}_l$ and $\vec{v}_m$ to $\vec{p}(\psi^*_l)$ and $\vec{p}(\psi^*_m)$, and fix the positions of $\psi^*_{j,\ j=l,\ldots,m}$; and if $K(\vec{v}_l)=1$ and $K(\vec{v}_m)>1$, move $\vec{v}_m$ to $\vec{v}^*_m$ whose position $\vec{p}(\vec{v}^*_m)=\vec{p}(\psi^*_m)$, and adjust the position of point $\psi^*_{j,\ j=l,\ldots,m}$. In the same way, if $K(\vec{v}_l)>1$ and $K(\vec{v}_m)=1$, we move $\vec{v}_l$ to $\vec{v}^*_l$ whose position $\vec{p}(\vec{v}^*_l)=\vec{p}(\psi^*_l)$, and adjust the position of point $\psi^*_{j,\ j=l,\ldots,m}$. After the position adjustment,

the projected points are shifted to attach to the triangular edges closely. A new parametric curve $c^*(u)$ passing all these points to represent the shape of the edge profile must also be computed. The resultant 3D profiles specified by the 2D strokes in Fig. 13a and 13d are shown in Fig. 13c and 13f.

5. Modified Variational Subdivision

In this section, we describe the technique to construct a triangular mesh surface interpolating the specified 3D profiles and the features nodes on a given garment template M^f. We achieve it by applying a modified variational subdivision scheme on the given garment template, which is defined as a coarse triangular mesh with some interpolating curves (profiles). The basic idea of a variational subdivision scheme is to iteratively applying a topological splitting operator to introduce new vertices to increase the number of degree of freedom, followed by a discrete fairing operator to increase the overall smoothness [24]. In our approach, we apply the scheme to construct a mesh surface that interpolates not only the initial vertices but also the specified profiles. The scheme is modified to achieve a better result by using the modified Butterfly scheme on the free boundaries of M_j^f (j represents the j th subdivision result of M^f, so $M_0^f \equiv M^f$), and the image-based collision detection is integrated to prevent collision between the constructed mesh surface and the human model. The collision detection is achieved by detecting whether the *z*-coordinate of a free vertex is between the front depth value and the back depth value of a human model [25]. The front depth value and the back depth value of a human model are computed by the graphics hardware. Thus, it is a very fast scheme.

Discrete fairing

Most surface fairing techniques use constrained energy minimization. One prominent energy function used in the theory and practice of surface design is the thin plate energy of a mesh M [32],

$$E_{TP}(M) = \int M_{uu}^2 + 2M_{uv}^2 + M_{vv}^2 . \tag{8}$$

Two kinds of schemes can minimize the energy function: linear and nonlinear. Since our surface design approach expects to obtain the result in "real time", we choose the umbrella-algorithm [11], which is a linear scheme. The respective variational derivative of $E_{TP}(M)$ corresponds to the second Laplacian,

$$L^2(M) = M_{uuuu} + 2M_{uuvv} + M_{vvvv} \tag{9}$$

After choosing a symmetric parameterization [33], the discrete representation of the Laplacian $L(M) = M_{uu} + M_{vv}$ turns out to be the umbrella operator

$$\vec{u}(\vec{v}_i) = \frac{1}{n}\sum_{k=0}^{n-1}(\vec{v}_k - \vec{v}_i) \tag{10}$$

with $\vec{v}_k$ being the direct neighbors of $\vec{v}_i$. The umbrella-operator can be applied recursively leading to

$$\vec{u}^2(\vec{v}_i) = \frac{1}{n}\sum_{i=0}^{n-1}(\vec{u}(\vec{v}_k) - \vec{u}(\vec{v}_i)) \tag{11}$$

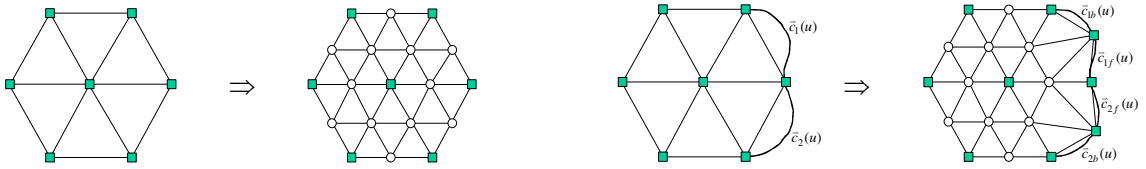
as a discretization of $L^2(M)$, which is called the 2nd order umbrella-operator. The characteristic system for the corresponding unconstrained minimization problem has rows $\vec{u}^2(\vec{v}_i) = 0$ for the free vertices $\vec{v}_i$. An iterative solving scheme approaches the optimal solution by solving each row of the system separately and cycling through the list of free vertices until a stable solution is reached. For the thin plate energy E_{TP}, this leads to

$$\vec{p}^{\,*}(\vec{v}_i) \leftarrow \vec{p}(\vec{v}_i) - \frac{1}{v}\vec{u}^2(\vec{v}_i) \tag{12}$$

with $v = 1 + \frac{1}{n_i}\sum_k \frac{1}{n_{i,k}}$, where n_i and $n_{i,k}$ are the valences of the center vertex $\vec{v}_i$ and its k th neighbor respectively [11].

Subdivision operators

Subdivision schemes are the most effective technique in geometric modeling with polygonal meshes [34]. A subdivision scheme here is given by operators to refine a triangular coarse mesh. By iteratively applying these operators, we generate a sequence of meshes $M^f_{j,\ j=1,\ldots,n}$ that eventually converge to a smooth limit surface M^f_∞ (Fig. 18).



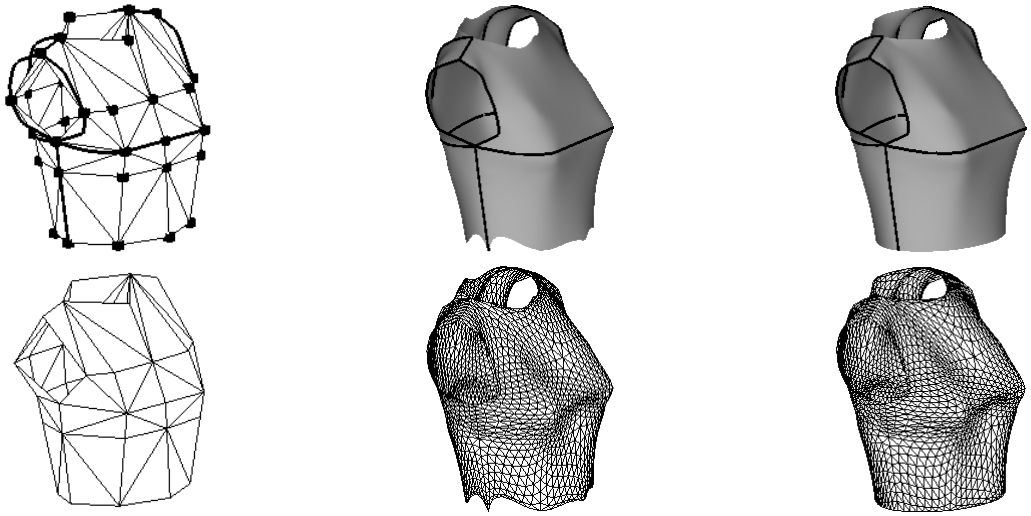
(a) without profile interpolation **(b) with profile interpolation**

Fig. 14 Topological split operation

There are always two operators used in the subdivision scheme. The first operator is a topological splitting operator that inserts new control vertices into the mesh. The split operation is chosen to be uniform so that all the new vertices are regular (valance is equal to 6, as shown in Fig. 14a). The position of the inserted new vertex $\vec{v}^{\,*}$, which lies on the edge $\vec{v}_s\vec{v}_e$, is determined by $\vec{p}(\vec{v}^*) = \frac{1}{2}(\vec{p}(\vec{v}_s) + \vec{p}(\vec{v}_e))$ if there is no profile specified on

$\vec{v}_s\vec{v}_e$; or by $\vec{p}(\vec{v}^*) = \vec{c}(\frac{1}{2})$ if there is a profile specified on $\vec{v}_s\vec{v}_e$, where $\vec{c}(u)$ is the parametric curve that represents the profile shape. We also divide $\vec{c}(u)$ into two parts $\vec{c}_f(u)$ and $\vec{c}_b(u)$ at $\vec{c}(\frac{1}{2})$, and attach $\vec{c}_f(u)$ and $\vec{c}_b(u)$ on the newly created edges from splitting (in Fig. 14b, $\vec{c}_{1f}(u)$ and $\vec{c}_{1b}(u)$ are from $\vec{c}_1(u)$, and $\vec{c}_{2f}(u)$ and $\vec{c}_{2b}(u)$ are from $\vec{c}_2(u)$).

The second operator is a smoothing operator that moves the control vertices according to weighted averages of neighboring vertices. The positions of vertices in the refined mesh are changed to achieve a global energy functional minimization. Here, we implement the 2nd order umbrella operator as an iterative solver of the problem. As mention by Kobblet [24], since each update step only computes a linear combination of nearby vertices, the computational complexity is linear if the number of umbrella iterations is bounded (in fact, a constant number). In order to guarantee that the resultant fine mesh interpolates the originally given vertices, the umbrella operator must not be applied to those vertices that already belong to the initial mesh. Also in order to guarantee that the result fine mesh interpolates the specified profiles, the umbrella operator must not update the positions of the vertices lying on the specified profiles. The vertices whose positions can be updated are called free vertices (rounded white nodes in Fig. 14), and the vertices whose positions cannot be updated are called fixed vertices (rectangular gray nodes in Fig. 14).



(a) **(b)** **(c)**

Fig. 15 Influence of boundary conditions

Boundary conditions

The boundary without additional constraints defined usually shrinks when using the 2nd order umbrella operator. For example, on the garment template M^f shown in Fig. 15a, there is no profile specified on the

boundary at the waist plane. Thus, after applying the above subdivision operators, the waist boundary shrinks (as shown in Fig. 15b). Keeping several vertices fixed during iterative energy minimization prevents the mesh from shrinking and collapsing. Therefore, for the vertices $\vec{v} \notin M_0^f$ on the boundaries without profiles specified, we calculate their positions by the mask for boundary vertices in the modified Butterfly subdivision scheme [26] (Fig. 16 shows the mask, where the rounded white node is the vertex whose position to be determined; and the rectangular gray nodes are the boundary vertices generated in last subdivision) and fix their position in the later discrete fairing process. After the modification, our subdivision scheme can obtain a mesh surface without boundary shrinking from the garment template M^f in Fig. 15a. The resultant mesh surface is shown in Fig. 15c.

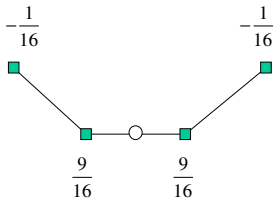
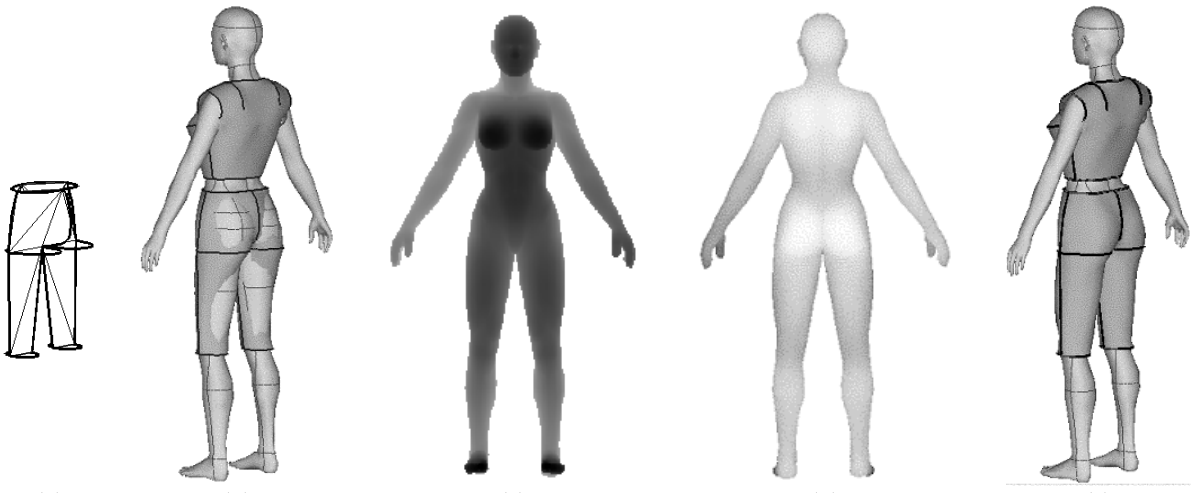


Fig. 16 Modified Butterfly mask for boundary vertices [27]

Collision avoidance

The surface constructed in our approach is around a 3D human model, so the collision between the constructed surface and the human model may happen during the surface construction procedure (i.e., the pants in Fig. 17b is constructed from the template mesh in Fig. 17a). Here, we modified the variational subdivision scheme by integrating the collision avoidance. In order to achieve a fast scheme, we implement the image-based collision detection [25].



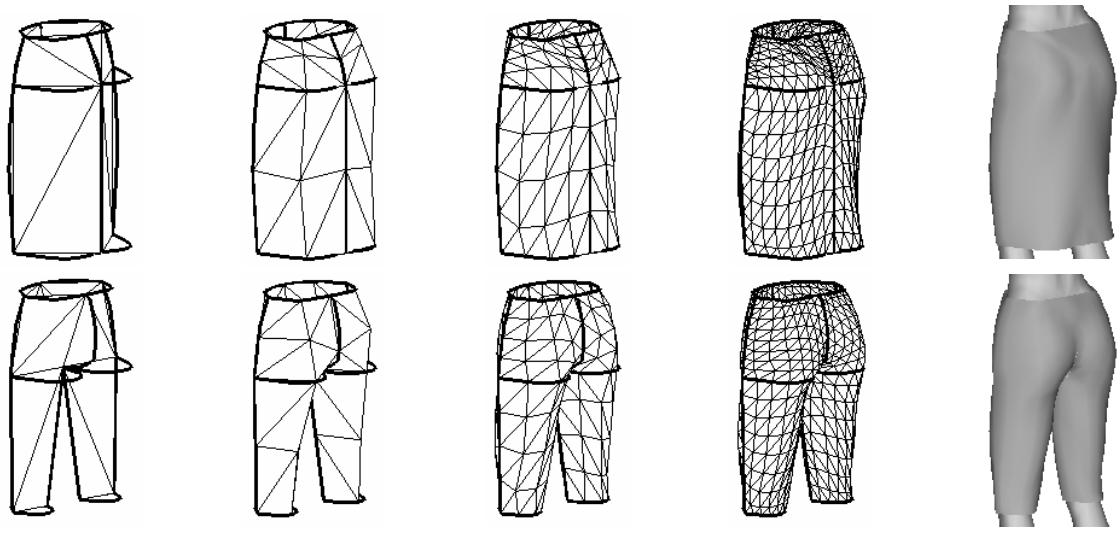
(a) **(b)** **(c)** **(d)** **(e)**

Fig. 17 Collision avoidance

First of all, the front depth buffer and the back depth buffer of a human model can be obtained from the graphics hardware (e.g., by the buffer-read function of the OpenGL Library). The buffers present the z height map of each pixel. In the same way, the coordinate surface normal of each pixel can be obtained from the RGB color map [25]. After the topological split operator inserts new vertices, we use the front depth buffer (Fig. 17c) and the back depth buffer (Fig. 17d) to detect whether a free vertex is inside a human model. We compute the pixel coordinate (X_{front}, Y) and (X_{back}, Y) of the free vertex in the front-view image and the back-view image. If the following conditions are satisfied, the free vertex is inside the human model.

$$\begin{cases} back: & z \geq backbuffermap(X_{back}, Y) \\ front: & z \leq frontbuffermap(X_{front}, Y) \end{cases} \tag{13}$$

If a vertex is inside the human model, pull it to the outside space of the human model along the discrete surface normal direction at the free vertex. In the discrete fairing operation, if a 2nd umbrella-operator pull a vertex from the outside space of human model to the inside space of human model, the update is prevented. To obtain a more accurate result, we can subdivide the update $-\frac{1}{v}\vec{u}^2(\vec{v}_i)$ into n steps, and perform the update step by step while detecting the collision between the updated vertex and the human model. The surface construction result with collision avoidance is shown in Fig. 17e, and the progressive results of constructing a mesh surface is shown in Fig. 18. Results show that this method is sufficient to detect all collisions around the human model as well as under the arms and between the legs. It is because that this approach is a little bit like converting the human model from the mesh-based representation into a voxel-based representation in the pixel level by the graphics hardware. Thus, it is a very fast scheme.



(a) M_0^f **(b)** M_1^f **(c)** M_2^f **(d)** M_3^f **(e)** M_∞^f

Fig. 18 Progressive construction results

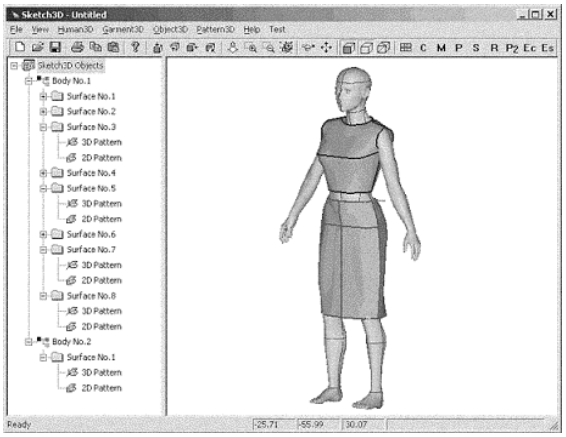


Fig. 19 Interface of the prototype software

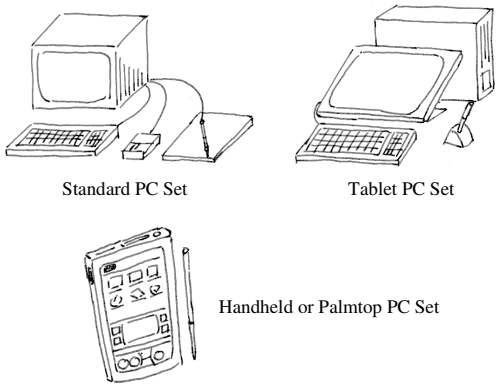
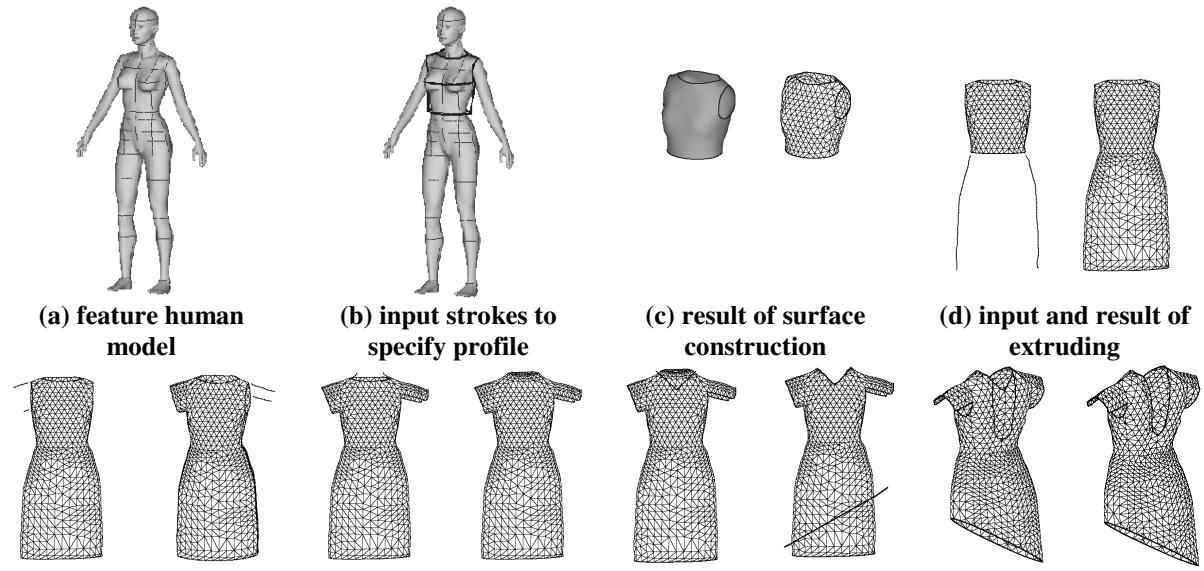


Fig. 20 Hardware sets for implementation



(a) feature human model **(b) input strokes to specify profile** **(c) result of surface construction** **(d) input and result of extruding**

(e) progressive extruding **(f) cutting strokes and results**

(g) painting seam line **(h) results of separation** **(i) related 2D patterns**

Fig. 21 Overview of the system operations

6. System and Application

We implemented a prototype of the sketch-based garment design platform on Windows NT/2000, using Visual C++ and OpenGL Library (Fig. 19). Our platform can be used on a standard PC (preferably equipped with a pen pad) or a tablet PC set at present; and a handheld or palmtop PC set is another piece of equipment that has potential in the future (Fig. 20). User inputs 2D freeform strokes via the pen-based drawing devices, and

then these 2D strokes can be used as an expressive design tool in our system to create and modify 3D models or to separate a model into components that can be flattened into 2D patterns.

In the prototype system, the approach presented in this paper is a key part that constructs a prototype garment mesh surface according to the features on a human model. Once a prototype model is created though 2D strokes (Fig. 21a, 21b, and 21c), it may be modified using various operations. The user can draw a stroke to specify the boundary or area to be extruded; and then rotate the model to draw another stroke or two specifying the silhouette of the extruded surface (Fig. 21d and 21e) [35]. In the cutting operation mode, the user can draw a stroke across the contour of the model to cut some parts of the model off (Fig. 21f). Seam lines on the surface are painted by drawing strokes within the model silhouette (Fig. 21g). Finally, the whole model can be separated into product components with reference to the painted seam lines (Fig. 21h), and flattened into 2D patterns (Fig. 21i) [6].

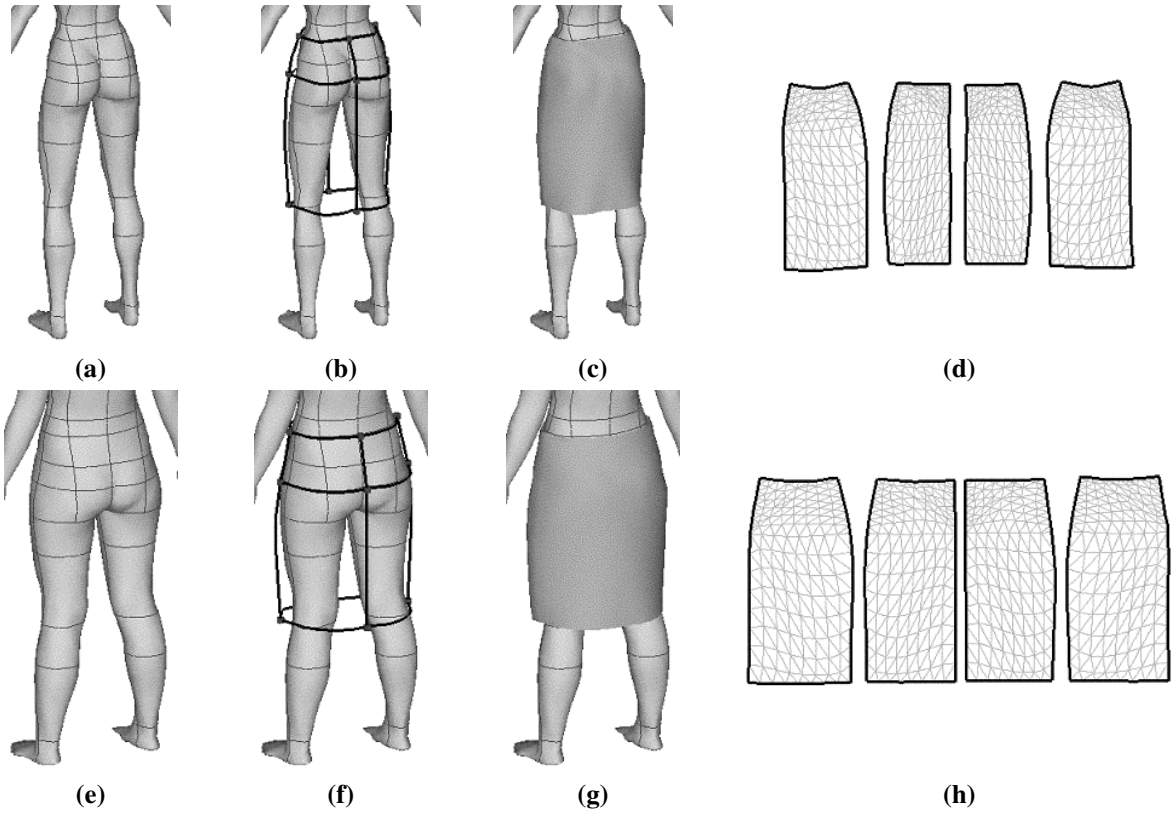


Fig. 22 Customized 3D grading

The technique presented in this paper also serves as the foundation to the customized 3D grading in the garment industry. For example, after we specify profiles on the human model H_1 in Fig. 22a, we obtain the garment template mesh M_0^f in Fig. 22b. The constructed surface M_∞^f and its related 2D patterns for the garment manufacturing are shown in Fig. 22c and 22d. When we construct the same style of skirt for the human

model H_2 in Fig. 22e, we relocate the feature nodes in M_0^f according to the features on H_2 by equation (1) and compute the new positions of the attached points by equation (5), the new garment template mesh M_0^* is automatically determined (Fig. 22f). After surface construction and flattening, its related surface and patterns are obtained as shown in Fig. 22g and 22h. The patterns generated by this 3D customized grading are totally different from the ones obtained from rule-based 2D scaling; the rule-based scaling methods are currently used by the garment industry. These patterns are regenerated by a feature-based approach, so they fit the customized human model as much as possible. This technique can greatly improve the efficiency and the quality of pattern generation in the garment industry.

7. Conclusion and Discussion

This paper presents a new feature-based approach for intuitively modeling a 3D garment around a 3D human model using 2D sketches input. The human models are with features pre-defined, and the constructed garment surfaces are related to the features on human models. Our approach consists of three parts: 1) construction of garment feature templates; 2) 3D profile specification using 2D strokes; and 3) construction of garment surfaces interpolating the specified 3D profiles. The resultant mesh surface can be cut and flattened into 2D patterns to be manufactured. At the end of the paper, in order to demonstrate the functionality of our approach, the prototype system and applications are given. Compared with earlier approach, our method has the following advantages:

- The tool provided by our approach can design garment patterns directly in the 3D space through 2D sketches, which is a characteristic not available in other computer aided garment design systems.
- The constructed 3D garment patterns are related to the features on a human model, so the patterns can be regenerated automatically when creating the same style of garment for other feature human models.
- The easing space is given between the specified profiles and the cross-section of a human model; therefore, not only the tight garments but also the loose garments can be constructed by our approach.
- Since most designers still prefer to express their creative design ideas through 2D sketches, the 2D sketched input method provided by our approach benefits the designers.

In summary, our technique can greatly improve the efficiency and the quality of pattern making in the garment industry.

For profile specification, we project the user input 2D strokes on a plane to determine the 3D positions of the vertices consisting the profiles. However, users may want to specify the profiles whose points do not lie on

the same plane. More complex specification tools need to be further developed to deal with this problem. While constructing the interpolating surface from the garment template with profiles specified, we apply the uniform subdivision scheme, which leads to the non-uniform distribution of triangle density in the mesh surface. Adaptive subdivision scheme can be applied to avoid the non-uniform distribution.

Acknowledgement

The authors would like to acknowledge the comments given by Mr. Terry Chang, Mr. Luo Zegang, and Dr. Au Chi Kit.

References

[1] Volino P., Courchesne M., and Thalmann N.M., Versatile and efficient technique for simulating cloth and other deformation objects, SIGGRAPH 95 Proceeding, ACM., 1995, pp.137-144, New York, USA.

[2] 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.

[3] Wang C.C.L., Chang T.K.K., and Yuen M.M.F., From laser-scanned data to feature human model: a system based on fuzzy logic concept, Computer-Aided Design, to appear, Elsevier Science, UK.

[4] Chang T.K.K., Wang C.C.L., and Yuen M.M.F., Web-based design and manufacturing of custom manniquin model, ICED01 - The 13th International Conference on Engineering Design, SECC, Glasgow, UK, August, 2001.

[5] Wang C.C.L., Wang Y., Chang T.K.K., and Yuen M.M.F., Virtual human modeling from photographs for garment industry, Computer-Aided Design, to appear, Elsevier Science, UK.

[6] Wang C.C.L., Smith S.S.F., and Yuen M.M.F., Surface flattening based on energy model, Computer-Aided Design, to appear, Elsevier Science, UK.

[7] Schroeder W.J., Zarge J.A., and Lorensen W.E., Decimation of triangle meshes, Computer Graphics, v26, n2, pp.65-70, 1992, USA.

[8] Garland M., and Heckbert P.S., Surface simplification using quadric error metrics, SIGGRAPH 97 Conference Proceedings, pp. 209-16, 1997.

[9] Wu J.-H., Hu S.-M., Tai C.-L., and Sun J.-G., An effective feature-preserving mesh simplification scheme based on face constriction, Proceedings Ninth Pacific Conference on Computer Graphics and

Applications, Pacific Graphics 2001, Tokyo, Japan, 16-18 Oct., 2001. IEEE Comput. Soc. 2001, pp.12-21. Los Alamitos, CA, USA.

[10] Taubin G., A signal processing approach to fairing surface design, SIGGRAPH 95 Conference Proceedings, ACM., 1995, pp.351-58, New York, USA.

[11] Kobbelt L., Campagna S., Vorsatz J., and Seidel H.P., Interactive multi-resolution modeling on arbitrary meshes, SIGGRAPH 98 Conference Proceedings, ACM., 1998, pp.105-114, New York, USA.

[12] Desbrun M., Meyer M., Schroder P, and Barr A.H., Implicit fairing of irregular meshes using diffusion and curvature flow, SIGGRAPH 99 Proceeding, ACM., 1999, pp.409-416, New York, USA.

[13] Schneider R., and Kobbelt L., Generating fair meshes with G^1 boundary conditions, Geometric Modeling and Processing Conference Proceedings, 2000.

[14] Schneider R., and Kobbelt L., Geometric fairing of irregular meshes for free-form surface design, Computer Aided Geometric Design, vol. 18, no. 4, 2001, pp. 359-379.

[15] Zeleznik R.C., Herndon K.P., and Hughes J.F., SKETCH: An interface for sketching 3D scenes, SIGGRAPH 96 Proceeding, ACM., 1996, pp.163-170, New York, USA.

[16] Igarashi T., Matsuoka S., and Tanaka H., Teddy: a sketching interface for 3D freeform design, SIGGRAPH 99 Proceeding, ACM., 1999, pp.409-416, New York, USA.

[17] Bloomenthal J., and Wyvill B., Interactive techniques for implicit modeling, 1990 Symposium on Interactive 3D Graphics, pp. 109-116, 1990.

[18] Markosian L., Cohen J. M., Crulli T., and Hughes J., Skin: a constructive approach to modeling free-form shapes, SIGGRAPH 99 Conference Proceedings, ACM., 1999, pp.393-400, New York, USA.

[19] Suzuki H., Sakurai Y., Kanai T., and Kimura F., Interactive mesh dragging with an adaptive remeshing technique, Visual Computer, vol.16, no.3-4, 2000, pp.159-76, Springer-Verlag, Germany.

[20] Zorin D., Schroder P., and Sweldens W., Interactive Multiresolution Mesh Editing, SIGGRAPH 97 Proceeding, ACM., 1997, New York, USA.

[21] 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.

[22] Levin A., Combined subdivision schemes for the design of surfaces satisfying boundary conditions, Computer Aided Geometry Design, vol. 16, no. 5, 1999, pp. 345-354.

[23] Levin A., Interpolating nets of curves by smooth subdivision surfaces, SIGGRAPH 99 Proceeding, ACM., 1999, pp. 57-64, New York, USA.

[24] Kobbelt L., Discrete fairing and variational subdivision for freeform surface design, Visual Computer, vol. 16, no. 3/4, 2000, pp. 142-158.

[25] Vassilev T., Spanlang B., and Chrysanthou Y., Fast cloth animation on walking avatars, Eurographics 2001 Proceeding, September 2001.

[26] Zorin D., Schroder P., and Sweldens W., Interpolating subdivision for meshes with arbitrary topology, SIGGRAPH 96 Proceeding, ACM., 1996, pp. 189-192, New York, USA.

[27] Hoppe H., DeRose T., Duchamp T., McDonald J., and Stuetzle W., Mesh optimization, SIGGRAPH 93 Proceeding, ACM., 1993, pp. 19-26, New York, USA.

[28] Varady T., Martin R. R., and Cox J., Reverse engineering of geometry models – an introduction. Computer Aided Design, 1997, 29(4): 255-268.

[29] Wang C.C.L., Surface free-form deformation and physics-based surface development, MPhil. Thesis, Hong Kong University of Science and Technology, December 1999.

[30] Piegl L., and Tiller W., *The NURBS Book* (2nd ed), Berlin; Hong Kong: Springer, 1997.

[31] Dahlquist G., and Bjorck A., *Numerical Method* (translated by N. Anderson), Englewood Cliffs, N.J.: Prentice-Hall, 1974.

[32] Moreton H., and Sequin C., Functional optimization for fair surface design, SIGGRAPH 92 Proceeding, ACM., 1992, pp. 167-176, New York, USA.

[33] Kobbelt L., Discrete Fairing. Proceeding of the Seventh IMA Conference on the Mathematics of Surfaces' 97, 1997, pp.101-131.

[34] Zorin D., Schroder P., DeRose T., Kobbelt L., Levin A., and Sweldens W., *SIGGRAPH 2000 Course Notes: Subdivision for Modeling and Animation*, ACM., 2000, New York, USA.

[35] Wang C.C.L., and Yuen M.M.F., Sketch based mesh extrusion with remeshing techniques, 21st Computers and Information in Engineering Conference, Paper No. DETC2001/CIE-21305, ASME, Pittsburgh, Pennsylvania, September 2001.