CAD Tools in Fashion/Garment Design

Charlie C. L. Wang

[1]Chinese University of Hong Kong, cwang@acae.cuhk.edu.hk

ABSTRACT

This paper present the most resent research that helps the development of CAD tools in fashion/Garment design. First of all, the human model reconstruction method is introduced to rebuild a 3D human body in the computer system. After that, the necessary techniques supporting design and modification of apparel products are described. Finally, the surface flattening algorithm is developed to convert 3D designed cloth pieces into 2D related patterns for manufacturing.

Keywords: CAD tools, fitting, apparel products, and three-dimensional solution.

1. INTRODUCTION

The purpose of this research is trying to provide a solution for the design automation of customized apparel products. This can greatly improve the efficiency of pattern design in apparel industry. At present, 2D CAD systems are widely used in the cloth industry. However, the pattern generation process is still a bottleneck of garment manufacturing, especially when the patterns for the same style but different bodies are required. Current commercial garment CAD systems [1-2] provide 2D grading tools to generate patterns of different sizes from the basic pattern set. Their methods are based on the 2D grading rules – usually offset curves generated in plane. Since the grading rules are applied in plane, it is not intuitive to preserve the fit of final dressing in spatial space. 2D approaches can hardly generate the fitted clothes for different human bodies. The only way to fundamentally solve the fitting problem of clothes is to design products in 3D. In the DresssingSim [3] solutions, some 3D design tools are provides. However, they seem relatively primary.

Cloth simulation techniques [4-9] provide a way of testing patterns by assembling 2D patterns in a computer system and draping them on a virtual human body. However, the functions of modifying the shape of patterns according to different human bodies are not provided. Recently, an online made-to-measure system was presented by Cordier et al. [10]. Their system allows interactive adjustment of the 3D mannequin according to the shopper's body measurements, online resizing of the garment to fit the mannequin, and real-time simulation of the garment corresponding to the body motion. However, their approach is also based on 2D pattern design and draping simulation; and their major purpose is for visualization and animation purpose, not for design. Our approach provides tools to construct patterns directly on virtual human bodies in 3D space, which solves the fitting problem from fundamental.

2. HUMAN BODY RECONSTRUCTION

The human models are reconstructed by the scanned point cloud. Usually, the input point cloud has some noisy. We develop an approach [] to remove both the points with a "far" distance from the surface of the object and the "prods" on the surface of the object (see Fig.1). The filtering algorithm is based on two basic morphology operators of processing binary images: erosion and dilation. Our algorithm consists of 4 steps: 1) project the scanned data with noisy points in different directions to obtain binary images; 2) apply morphology operators to remove or separate noisy points from the scanned object in each binary image; 3) cluster point sets by morphological dilation; 4) maintain the point in the set with the maximum point number and remove points in other sets.

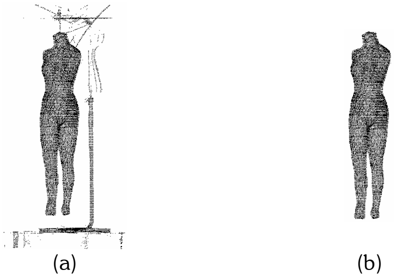


Fig. 1. Input and output of the filtering algorithm: (a) scanned data with noisy points; (b) proposed result.

After the could points are processed to have no noisy and with orientation fixed – facing the x direction, we apply semantic feature extraction technique on the unorganized cloud points from 3D laser scanners to construct the feature wireframe of a human body in three steps: 1) extracting the key feature points on the cloud points; 2) using anthropometry rules to determine the semantic feature points; 3) linking all the feature points by feature curves interpolating the cloud points. The linked feature points and the linking feature curves form the feature wireframe, the topology of which keeps consistent to all processed human models.

The key feature points on the surface of a human body, including the underarm points, the crotch point, the belly-button point, the front neck point, the back neck point, and the busty points (illustrated in Fig.2), must be extracted first. The fuzzy logic based approach [12] is adopted here. The basic idea is that: using some planes to intersect the 3D unorganized points of a human body's scan or projecting points onto some planes to obtain 2D contours (depicted in Fig.3). We can determine turning points on the 2D contours by the "sharp" angles along the contour, where the definition of "sharpness" follows the fuzzy logic concept. With the help of turning points, we can detect the key feature points.

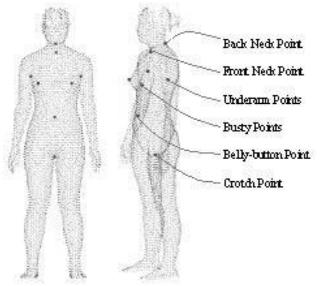


Fig. 2. Key feature points on a human body.

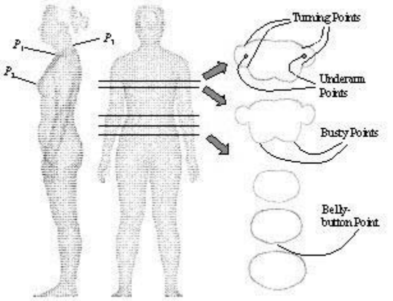


Fig. 3. Determine the key feature points.

The location of the semantic feature points on the surface of a scanned human body can be roughly determined according to the anthropometry rules by the key feature points. To accurate locate the semantic feature points, we should also adopt the feature extraction algorithm with fuzzy logic concept (ref. [12]). The basic idea is similar to the approach of detecting key feature points – using cutting planes and projection planes. The unnecessary detail of the procedure is omitted here. An example is shown in Fig.4a

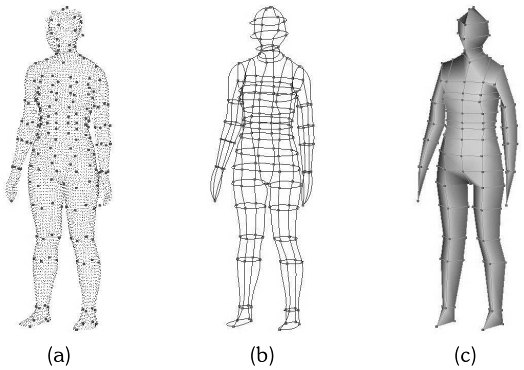


Fig. 4. Feature wireframe and it related topology graph: (a) with all feature points determined; (b) the final feature wireframe; (c) topology graph.

Now, to construct the feature wireframe, we need to link feature points with parametric curve, which are called feature curves as each curve has its semantic meaning according to sizing dimensions. The curve should pass through the feature points and approximate the shape of the scanned human body. The parametric curves utilized in our implementation are 4th-order Bézier curves. Each curve has four control points; the first and the last control points are coincident with the feature points, so only the middle two control points can be adjusted to approximate the shape of the scanned human body. Every feature curve lies on a plane. When computing the control points of a curve, we first intersect the cloud points by the plane containing this curve to obtain a contour of points. Then, using the semantic feature information and the starting and ending points of the curve to select points for approximating the feature curve. Finally, a least-square fitting [13] is adopted to determine the positions of the control points.

Since the feature wireframes are consistent to all human models, topology graphs by connecting feature curves with face entities are also consistent to all human models. This topology graph is a starting point of generating feature patches interpolating the feature wireframe. Fig.4c shows an example topology graph according to the feature wireframe shown in Fig.4b. It is easy to find that not only 4-sided patches but also 5-sided patches are included. The feature patches interpolating the feature curves are first generated using Gregory patch (Fig.5); in the following, the feature patches are updated according to the scanned points by a voxel-based algorithm (Fig.6); finally, the mesh surface is adjusted to become symmetric (Fig.7). For detail algorithms, see [14].

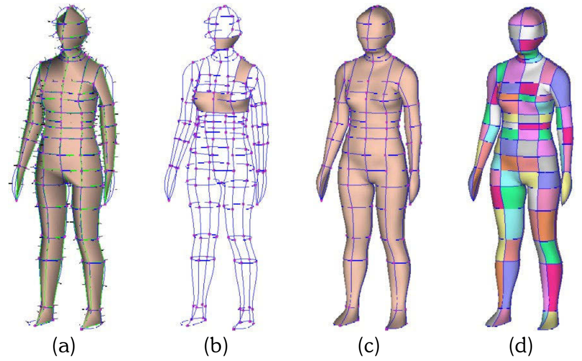

(a) (b) (c) (d)

Fig. 6. Gregory interpolation of feature patches: (a) topology graph with cross normals; (b) curves and patches on a human body; (c) all feature patches generated; (d) checkerboard pattern to verify the feature patches.

In the surface update step, we introduce an algorithm [14] to iteratively improve the fitting accuracy by minimizing the shape difference between the mesh surface M and the scanned data. To have a better performance at the computing speed, a voxel-based technique is adopted. The procedure of surface refinement consists of three basic steps: 1) voxel construction, 2) vertex position update, and 3) mesh relaxation. The second and the third steps are executed iteratively until a satisfied mesh is obtained (see Fig.7).

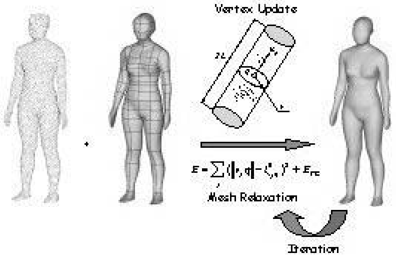


Fig. 7. Voxel-based vertex position update.

The human body generated from the scanned data is usually asymmetric. When the human model is generated for sizing survey or for mannequin manufacturing, the symmetric models are required. If this is the case, the refined mesh surface has to be further modified. Let us name the resultant surface after mesh refinement as M^+, the feature wireframe as F^+, and the scanned cloud points as S^+. First of all, the entire data set of a human body including M^+, F^+, and S^+ are transformed to let the crotch point on the origin. For every feature curve C in F^+, there is a dual curve C^* in F^+, where they should be symmetric on a symmetric wireframe. Also, for each feature node P in F^+, there should exist a symmetric dual node P^* of it in F^+. By this property, we can construct a feature wireframe F^- which is symmetric to F^+. In detail, for each feature node P, update its position by the mirrored coordinate of P^* according to the x-z plane, and vice-versa; for each feature curve C, update its control points by the mirrored copy of the data points on C^*. Then, we mirror the coordinates of all points in S^+ to obtain S^-. After applying the surface interpolation and refinement algorithms on F^- and S^-, we obtain a refined mesh surface M^- of the mirrored human body. The symmetric model surface is obtained by the interpolation of M^+ and M^-: $M^S = \dfrac{M^+ + M^-}{2}$; at the same time, we can have the symmetric feature wireframe F^S by $F^S = \dfrac{F^+ + F^-}{2}$.

Fig. 8 shows a set of S^+, F^+, M^+, S^-, F^-, M^-, M^S and F^S.

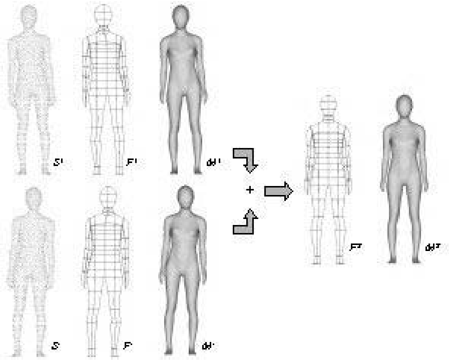


Fig. 8. Making symmetry.

3. NON-MANIFOLD DATA STRUCTURE

The apparel products are represented by freeform surfaces because of their complex geometry models. To simplify the complex surface of apparel products, during the design phase, abstractions are stored along with models, which leads to the non-manifold data structure and operators.

The framework of the data structure is shown in Fig.9. An object stored by this data structure is a collection of MESHSURFACEs, each of which is a complex of triangles. The MESHSURFACEs are jointed by a new entity – MESHJOINT, which is comprised of ordered

MESHEDGEs on the MESHSURFACEs. Locally, this data structure is a little bit similar to the winged-edge data structure. Every TRGLEDGE contains the link to its left face and right face; and every TRGLNODE has its adjacent faces, edges, and nodes stored. Four attributes are also defined in the data structures for the representation of features. They include ATTRIB_EDGE, ATTRIB_EDGENODE, and ATTRIB _FACENODE. The ATTRIB_EDGENODE and ATT- RIB_FACENODE both are attribute nodes; they are derived from ATTRIB_NODE. An ATTRIB_EDGE- NODE is an attribute node on the TRGLEDGE, its coordinate depends on the position of TRGLEDGE's nodes; and an ATTRIB_FACENODE is an attribute node in the TRGLFACE, its coordinate depends on the position of TRGLFACE's nodes. An ATTRIB_EDGE is an ordered collection list of ATTRIB_NODEs, which can be either ATTRIB_EDGENODEs or ATTRIB_ FACENODEs.

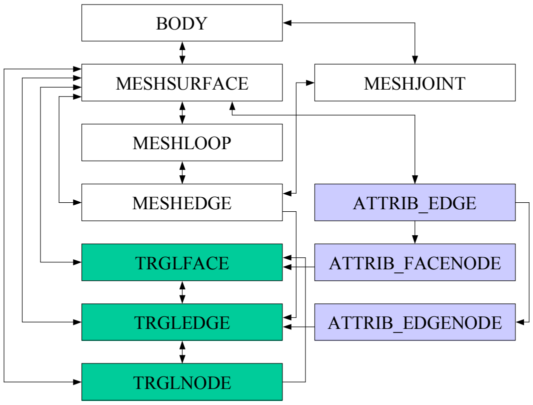


Fig. 9. Data structure framework [15]

4. DESIGN OF 3D APPAREL PRODUCTS

Based on the non-manifold data structure presented in the above section, and the parameterized human bodies generated in section 2, a constructive design technique is developed in this section. By this method, we can construct profile templates, which represent different styles of clothes, corresponding to feature vertices, curves and patches of a parameterized human model. Then, when the body with different shape is applied, by our decoding method, the clothes preserving easing relationships will be generated. Also, we can control the final shape on edges through changing the profile curves on the encoded templates.

When building the profile template T_P for a style of clothes, two steps of interactivities are involved. Every feature node in the profile template should first be encoded on either the parameterized human body or other nodes that have already been encoded in T_P. After all feature nodes are encoded, the topological graph linking the nodes should be interactively input by users. The processes of profile template encoding more or less like using the interactive tools to build a coarse freeform surface.

When encoding a feature node on a parameterized human body H, which is also represented by a polygonal mesh, the feature node can be encoded on a vertex, an edge, or a face on the mesh of human model – they are called reference elements. Every feature node is first created by specifying its (*x*, *y*, *z*) coordinate. You then can choose the encoding manner: 1) by vertex, 2) by edge, or 2) by face. When the manner is chosen, you could pick the reference element, which you want to encode the feature node, on the surface of H. Then, the relationship between the feature node and a human model is encoded. The relationship is actually the relative coordinate of the feature nodes on the selected element. A feature node in T_P can be exactly encoded on a feature node of H in the vertex manner, be encoded on a feature curve of H in the edge manner (since each feature curve is actually a set of linked edges), and be encoded on a feature patch by a face of the patch.

In the vertex-encoding mode, if a vertex V_b is selected as the reference element, it is easy to obtain the unit normal vector n_{V_b} at the vertex; getting the first vertex V_b^1 adjacent to V_b in its adjacent vertices list, we can have

$$X_V = n_{V_b}\ ,\ \ Y_V = \frac{V_b^1 - V_b}{\left\| V_b^1 - V_b \right\|} \times n_{V_b}\ ,\ \ Z_V = X_V \times Y_V\ . \qquad (1)$$

If Y_V is degenerated, by $V_b^1 - V_b = 0$ or $(V_b^1 - V_b) // n_{V_b}$, we take place V_b^1 by V_b^2 in the above equations. Thus, when a feature node V_{T_P} in T_P is encoded on V_b, the encoded information includes the index of V_b on H, and the scalars of $u_V = (V_{T_P} - V_b) \cdot X_V$, $v_V = (V_{T_P} - V_b) \cdot Y_V$, and $w_V = (V_{T_P} - V_b) \cdot Z_V$. In the edge-encoding manner, an edge E_b serves as the reference element; the normal n_{E_b} at E_b is usually computed by averaging the normal vectors of its left and right faces. After determining the nearest point V_{E_b} on E_b to V_{T_P}, a local frame at V_{E_b} can be determined at V_{E_b} by

$$X_E = n_{E_b}\ ,\ \ Y_E = t_{e_b} \big/ \left\| t_{e_b} \right\|\ ,\ \ Z_E = X_V \times Y_V\ , \qquad (2)$$

where t_{e_b} is the direction vector of edge E_b. Since the normals on E_b's left and right faces are perpendicular to v_{e_b}, they form a plane perpendicular to E_b; n_{E_b} is on the plane, so $n_{E_b} \perp v_{e_b}$. The encoded information in the edge manner includes the index of E_b on H, the parameter t of V_{E_b} on E_b ($t \in [0,1]$), the scalars of $u_E = (V_{T_P} - V_{E_b}) \cdot X_E$, $v_E = (V_{T_P} - V_{E_b}) \cdot Y_E$, and $w_E = (V_{T_P} - V_{E_b}) \cdot Z_E$. In the face-encoding way, after a face F_b is chosen to be a reference element, its centroid V_{F_b} and its normal n_{F_b} can be easily determined. Getting the first vertex $V_{F_b}^1$ in the vertices list of F_b, the local frame at V_{F_b} is as

$$X_F = n_{F_b}, \quad Y_F = \frac{V_{F_b}^1 - V_{F_b}}{\left\| V_{F_b}^1 - V_{F_b} \right\|}, \quad Z_F = X_F \times Y_F . \tag{3}$$

Thus, the encoded information includes the index of F_b on H, and the scalars of $u_F = (V_{T_P} - V_{F_b}) \cdot X_F$, $v_F = (V_{T_P} - V_{F_b}) \cdot Y_F$, and $w_F = (V_{T_P} - V_{F_b}) \cdot Z_F$.

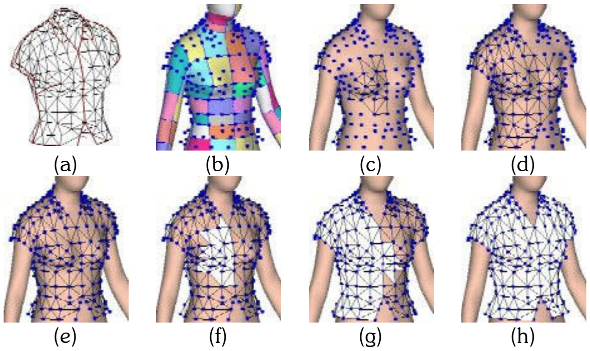


Fig. 10. An example of topological graph construction process.

After all the feature nodes in T_P are encoded, they need to be linked with edges and faces using interactive tools (i.e., specifying the topological graph T_P of interactively). The topological graph is a collection of PMESHs that are connected by PMESHJOINTs. For example, Fig.10a shows the topological graph of a stored in a BODY. Since our data structure is complex-based, the incomplete topology information during the construction process is easy to be stored. Fig.10b-10h shows some fragments of the construction process of the topological graph. In Fig.10b, the feature nodes have been defined around the reference model. An interactive tool is utilized to connect the feature nodes by edges as shown in Fig.10c-10d; and Fig.10e shows the feature template after creating all edges. Polygonal faces can also be created one by one interactively (Fig.10f and 10g). Fig.10h shows the final result.

In the above encoded T_P, the shape on an edge is not controlled. Here, we conduct the profile curves to control the shape of final surface on edges. A profile curve is a parametric curve $C_p(u)$ attached on a edge E_p in T_P, where $C_p(0)$ and $C_p(1)$ coincident to the two ending vertices on E_p. For example, it is represented as 4th order Bézier curves in our prototype system. E_p can also be represented by a parametric curve with the same type, so we have $E_p(0) = C_p(0)$ and $E_p(1) = C_p(1)$. The curve $C_p(u)$ can be specified by the traditional 3D curve input methods in CAD systems or by the sketched-input as shown in [15]. The following profile template decoding process will persuade the final refined surface interpolating the specified profile curves. Thus, the profile curves are utilized to control local shape on the final surface. A profile curve $C_p(u)$ is encoded on its edge by

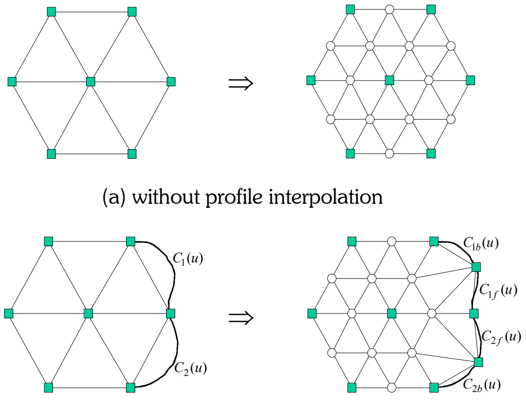
$$ep(u) = C_p(u) - E_p(u) . \tag{4}$$

After encoding the polynomial of $ep(u)$, the relationship between $C_p(u)$ and E_p is stored. When the positions of the endpoints of E_p are adjusted, we can construct a new parametric line segment $E_p^*(u)$ for it. Thus, the new profile curve can be obtained by

$$C_p^*(u) = ep(u) + E_p^*(u) . \tag{5}$$

After a profile template T_P has been encoded, it can be applied to any parameterized human model H* whose shape is different to H. This involves the profile template decoding process. A decoding process includes the steps of relocating the positions of feature nodes, reshaping the profile curves, and surface refinement.

For a feature node V_{T_P} encoded on a vertex, by the encoded vertex index, the vertex V_b^* on H* is determined. A local frame can be established at V_b^* the formula shown in eq.(1). By the stored (u_V, v_V, w_V) and the reconstructed local frame at V_b^* on H*, it is straightforward to determine the new position of V_{T_P}. For a feature node encoded on an edge, by index and t, the edge E_b^* on H* and the vertex $V_{E_b}^*$ on E_b^* are easily to be obtained. After the local frame at $V_{E_b}^*$ is determined by eq.(2), the new position of V_{T_P} is obtained using (u_E, v_E, w_E). Also, for the feature nodes encoded on faces, we relocate their position in the same way: calculate the centroid of the face on H*; computer the local frame at the centroid by eq.(3); finally relocate the feature node by the stored (u_F, v_F, w_F). After all feature nodes have been relocated, the new parametric representation of the profile curves are computed by eq.(5). They and the feature nodes are interpolated during the following surface refinement.



(a) without profile interpolation

(b) with profile interpolation

Fig. 11. Topological splitting operator.

The surface refinement step is to provide detail and smooth freeform surfaces for representing the shape of an apparel product in computer system. The method applied here is from [15] – the modified variational subdivision scheme, which iteratively applies a topological splitting operator to introduce new vertices to increase the degrees of freedom, followed by a discrete fairing operator to increase the overall smoothness. The constructed mesh surfaces interpolate not only the initial vertices but also the specified profiles. The topological splitting operator 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. 11a). If non- triangular faces are involved, we just simply triangulate them before refinement. When inserting a new vertex on a edge, its position is located at the middle of the edge if there is no profile curve attached; if a profile curve $C_p(u)$ is attached on this edge, the inserted vertex is located at $C_p(0.5)$. The curve $C_p(u)$ is divided into two curves and attached on the split two edges (see Fig. 11b). The smoothing operator moves the control vertices according to the 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 [16]. In order to guarantee that the resultant fine mesh interpolates the originally given vertices, the umbrella operator must not be applied to those vertices belonging to the initial mesh. Also in order to guarantee that the resultant fine mesh interpolates the 3D profiles, the umbrella operator must not update the positions of the vertices lying on the profiles. As mentioned in [15], collision detection should also be incorporated to prevent vertices moving inside human bodies.

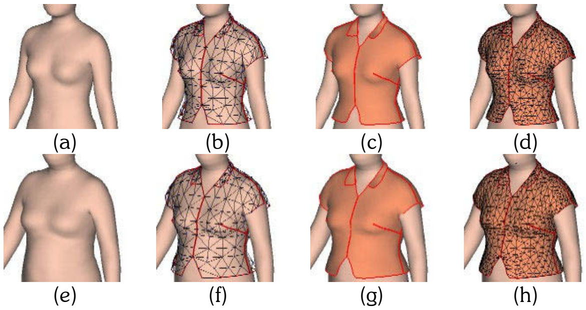


Fig. 12. Clothes regenerated on different human models.

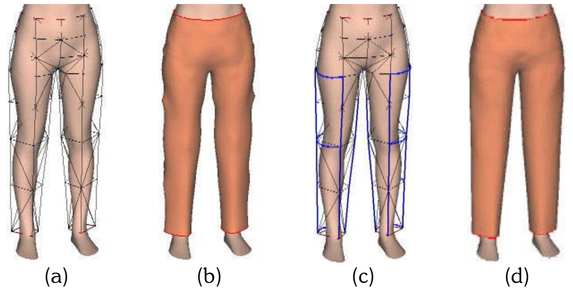


Fig. 13. Using profile curves to control final shape: (a) template without profile curves; (b) resultant shape without profile control; (c) template with profile curves; (d) resultant shape with profile control.

Fig.12 shows the profile template decoding results on different human models. Fig.12a and 12e are the new human bodies H1 and H2 applied; Fig.12b and 12f show the decoding processing after all feature nodes relocated; Fig.12c and 12g give the final surface on human bodies generated by the modified variational subdivision scheme, and Fig. 12d and 12h are related mesh representations. Fig. 13 explains how the profile curves (blue curves in the figure) control the final shape of refined surfaces. Fig. 13a is a template of pants without profile curves, and Fig. 13b gives its resultant shape. Fig. 13c and 13d shows the template and final shape with profile curve control, which is much better (with no shrinkage happens).

Fig.14 gives another example of the design automation of customized apparel products. After designing and encoding a piece of waistcoat and a piece of pants on a parameterized human body H which guarantees fit, the encoded relationship between the clothes and the mannequin can be applied to different human models to generate the customized 3D clothes fitting individual bodies – so the design automation of customized apparel products is implemented.

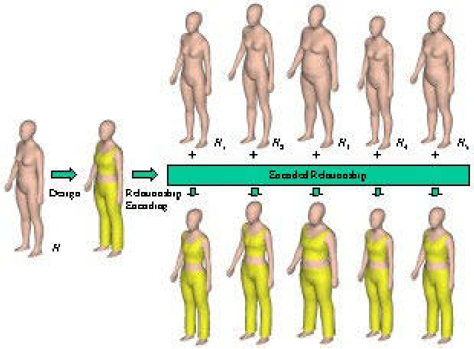


Fig. 14. Example of cloth design automation.

5. FREEFORM MODIFICATION

In this section, four most useful freeform surface modification tools for apparel products are introduced.

Mesh painting

Users can conduct this tool to specify curves on the surface of products by 2D strokes. Our algorithm creates 3D line segments by projecting each line segment of the input 2D stroke onto the surface meshes of the model along the view direction. The overall procedure is: for each line segment of the 2D stroke, first determine a bounding plane containing the projection of the line segment from the viewing position; then the system finds all intersections between the plane and each polygon of the object, and splices the resulting 3D line segments together (see Fig. 15). The actual implementation searches for the intersections efficiently using polygon connectivity information. If a ray from the viewing position crosses multiple polygons, only the polygon nearest to the viewing position is used for the surface painting. If the resulting 3D segments cannot be spliced together (e.g., if the stroke crosses a "fold" of the object, as shown in Fig. 16), the algorithm fails. The painted curves are stored by the ATTRIB_EDGEs in the data structure. There is another kind of painting, called penetrated painting, in which, all the polygons crossed by the rays are used to compute intersections. A "fold" does not influence the penetrated painting. The painting result is also stored by the ATTRIB_EDGEs. Both these paintings are implemented, and examples are shown in Fig. 17.

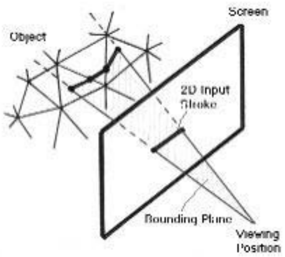


Fig. 15. Painting illustration.

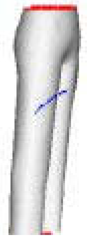
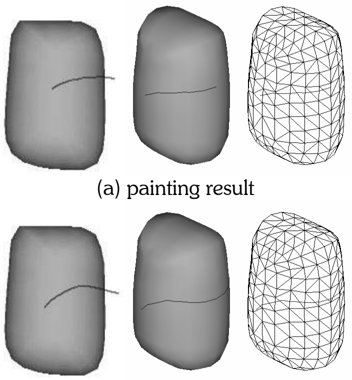


Fig. 16. Stroke across a "fold" leads algorithm fail.



(a) painting result

(b) penetrated painting result

Fig. 17. Examples of painting.

Mesh cutting

The mesh cutting is to remove some parts of the given mesh surface by input 2D strokes. Similar to the Teddy system [17], the cutting tool is based on the painting algorithm. After painting a curve on the surface of a model, the constrained Delaunay triangulation algorithm [18] makes the painted curves to form the triangle edges of the model. Removing the triangles on the user-selected side of the painted curve (specified by another stroke) from the model, the cutting result is obtained (illustrated in Fig. 18). Fig. 19 shows an example for using the mesh cutting tool to modify an evening dress.

Mesh extrusion

The extrusion operation is applied in our approach to create new polygonal meshes based on base surface line segments (called the base curve) and extruding strokes. The mesh extrusion method implemented here is from our previous development in [19]. The mesh extrusion method is best illustrated by examples such as that shown in Fig. 20. In these examples, the given initial model is the freeform surface of a shirt; the mesh extrusion tool allows the user to sketch 2D input strokes (one stroke on the surface of given mesh, and other strokes depicting the profile curves of the extruded surface) to extrude a surface from the given mesh. Firstly the user draws a stroke on the object surface; then the user rotates the model to bring the stroke sideways and draw silhouette lines to extrude the surface. A sweep operation is applied to construct the 3D shape by moving the surface base curve, which is obtained by projecting the first stroke onto the surface of the given mesh, along the skeleton of the profile curves.

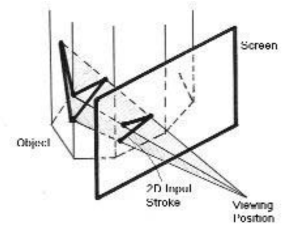


Fig. 18. Mesh cutting illustration.

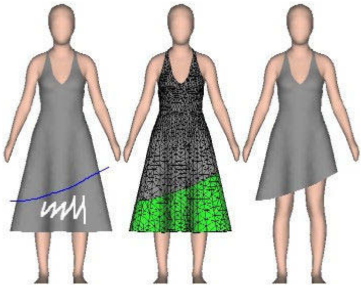


Fig. 19. Example of mesh cutting.

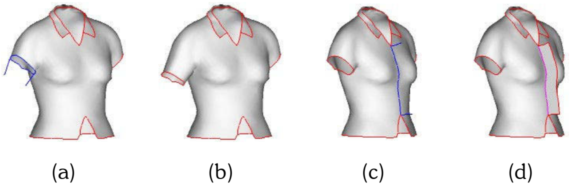


Fig. 20. Mesh extrusion: (a) extrude a ring out; (b) the band sleeve elongated; (c) extrude a curve out; (d) a hanging patch formed.

Mesh partitioning

The same as the cutting tool, the partitioning operation is also based on the painting algorithm. After painting the separating curves on the surface of a model (Fig. 21a), we apply the constrained Delaunay triangulation algorithm [18] to convert the painted curves into triangle edges of the model. After re-triangulation, the whole model can be divided to several sets of triangles; each set of triangles is a component of the product model (see Fig. 21b).

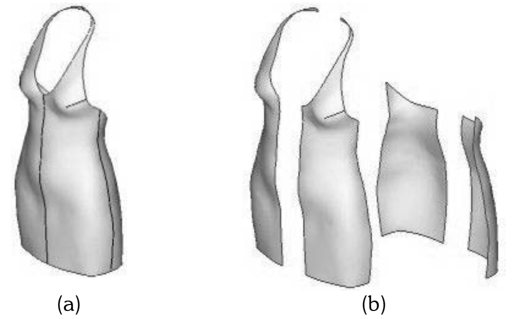


Fig. 21. Mesh partitioning: (a) parting curves are defined on the dress; (b) partitioned 3D pieces.

6. SURFACE FLATTENING

Using the operations introduced in above section, the user can create and modify 3D apparel models intuitively and efficiently. However, the garment manufacturing industry needs 2D patterns to be used in the manufacturing processes. Thus, we implement the energy-based surface flattening algorithm [20] to generate the corresponding 2D patterns of a 3D apparel model by using a spring-mass model.

This procedure consists of triangles flattening and planar mesh deformation. During the triangles flattening phase, triangles are flattened one by one; and a partial spring-mass system containing flattened triangles is deformed to release the strain energy during the flattening. After all the triangles are flattened, the spring-mass system will have all the triangles of the given surface. The planar triangular mesh deformation process is directed by the energy function of the spring-mass system. One example of a spring-mass system is shown in Fig. 22. In this example, nodes P_i are masses that correspond to the vertex $v_i \in \mathrm{M}$; 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 between them on the original spatial surface, we apply an attraction force between them (e.g., the force between P_0 and P_1); otherwise there will be a repellent force between them (e.g., P_0 and P_3). The energy function on one single mass P_i to be minimized is

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

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, and d_j is the geodesic distance between v_i and v_j on the given mesh surface M. The energy function for the whole surface patch is

$$E(\mathrm{M}) = \sum_{i=1}^{N} E(P_i)\,.$$

By releasing the energy function, we can obtain the 2D pattern related to the given 3D mesh surface. Fig. 23 shows the flattening result of the underwear in Fig. 21.

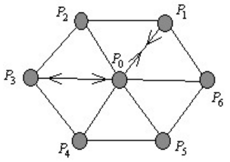


Fig. 22. Example of a node in a spring-mass system.

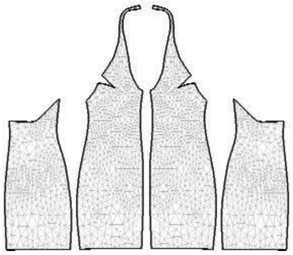


Fig. 22. 2D patterns for the 3D dress shown in Fig. 19.

Also, the method presented in [21] can be applied to find the cutting paths on a 3D triangular mesh surface to reduce the stretch in the flattened surface. The shortest cutting paths connecting all nodes where Gaussian curvature is larger than a threshold are generated according to our newly defined boundary geodesic distance map. The map encapsulates the undirected geodesic distance from every triangular node to the surface boundary approximately, and can be computed in linear time. The cutting paths on surfaces with widely distributed curvatures can also be found by generating cuts during the flattening process.

Our recent work related to surface development is about converting a non-developable surface into developable one during free-form deformation (FFD). In [22], the developability-preserved FFD problem is formulated as a constrained optimization problem. Different from other contained FFD approaches, the positions of lattice control points are not modified in our algorithm – as their control is insufficient in regards to the developability of all the nodes in the mesh. Moreover, the optimization is performed on the parameters of the mesh nodes rather than directly modifying their 3D coordinates, which avoids the time-consuming inverse calculation of the parameters of every node in a non-parallelepiped control lattice when further deformations are required. Fig. 23 gives an experimental result of our approach; where the color bar represents the degree of non-developability.

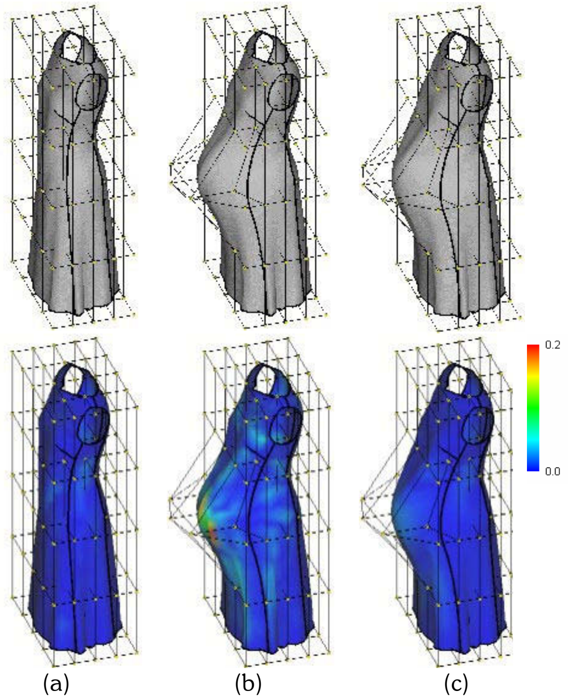


Fig.23. The style modification of lady dress: (a) before FFD; (b) after traditional FFD; (c) after developability preserved FFD.

7. CONCLUSION

This paper present the most resent research that helps the development of CAD tools in fashion/Garment design. First of all, the human model reconstruction method is introduced to rebuild a 3D human body in the computer system. After that, the necessary techniques supporting design and modification of apparel products are described. Finally, the surface flattening algorithm is developed to convert 3D designed cloth pieces into 2D related patterns for manufacturing. Compared to other approaches, our solution provides tools to construct patterns directly on virtual human bodies in 3D space, which solves the fitting problem from fundamental.

8. REFERENCES

[1] Gerber. http://www.gerbertechnology.com.

[2] Lectra. http://www.lectra.com.

[3] DressingSim. http://www.dressingsim.com.

[4] 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.

[5] 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.

[6] Baraff D., and Witkin A., Large steps in cloth simulation, *Proceedings of SIGGRAPH 98*, pp.43–54, 1998.

[7] Choi K.J., and Ko H.S., Stable but responsive cloth, *Proceedings of SIGGRAPH 2002*, pp.604–611, 2002.

[8] Bridson R., Fedkiw R.P., and Anderson J., Robust treatment of collisions, contact, and friction for cloth animation, *Proceedings of SIGGRAPH 2002*, pp.594–603, 2002.

[9] Bridson R., Marino S., and Fedkiw R., Simulation of clothing with folds and wrinkles, *Eurographics/SIGGRAPH Symposium on Computer Animation 2003*, pp.28-36, 2003.

[10] Cordier F., Seo H., and Thalmann N.M., Made-to-measure technologies for an online clothing store. *IEEE Computer Graphics and Applications*, vol.23, no.1, pp.38–48, January 2003.

[11] Wang C.C.L., and Yuen M.M.F., A binary morphology based filtering algorithm for reverse engineering, *International Journal of Advanced Manufacturing Technology*, vol.21, no.4, pp.257-262, 2003.

[12] Wang C.C.L., Cheng 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*, vol.35, no.3, pp.241-253, 2003.

[13] Mortenson M.E., *Geometric Modeling (2nd Edition)*. Wiley: New York, 1997.

[14] Wang C.C.L., Parameterization and parametric design of mannequins, *Computer-Aided Design*, to appear.

[15] Wang C.C.L., Wang Y., and Yuen M.M.F., Feature based 3D garment design through 2D sketches, *Computer-Aided Design*, vol.35, no.7, pp.659-672, 2003.

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

[17] Igarashi T., Tanaka H., and Matsuoka S., Teddy: a sketching interface for 3D freeform design, *SIGGRAPH 1999 Conference Proceedings*, 1999.

[18] de Floriani L., and Puppo E., An online algorithm for constrained delaunay triangulation, *CVGIP-Graphical Models & Image Processing*, vol.54, no.4, pp.290-300, USA.

[19] Wang C.C.L., and Yuen M.M.F., Freeform extrusion by sketched input, *Computers & Graphics*, vol.27, no.2, pp.255-263, 2003.

[20] Wang C.C.L., Smith S.S.F., and Yuen M.M.F., Surface flattening based on energy model, *Computer-Aided Design*, vol.34, no.11, pp.823-833, 2002.

[21] Wang C.C.L., Wang Y., Tang K., and Yuen M.M.F., Reduce the stretch in surface flattening by finding cutting paths to the surface boundary, *Computer-Aided Design*, accepted.

[22] Wang C.C.L., and Tang K., Developability-Preserved Free-Form Deformation of Assembled Patches, *ACM Symposium on Solid Modeling and Applications*, Genova, Italy, June 7-9, 2004.