From Styling Design to Products Fabricated by Planar Materials

Charlie C.L. Wang, *Member, IEEE,* Yunbo Zhang, and Hoi Sheung

***Abstract*— This article describes a geometric modeling system that generates industry required planar pieces for fabricating user-customized products from styling designs. The processing from style design to industrial patterns is automated. Pre-stored styling designs can be automatically mapped into different reference model shapes and then unfolded into planar pieces. Besides, a map-guided algorithm has been developed to locate unfolded pieces according to industrial requirement.**

***Index Terms*— Geometric modeling, styling design, industrial pieces, unfolding, application.**

I. Introduction

THE computer graphics techniques have been employed to simulate the behavior of sheet materials like cloth, leather and paper in computer animation or virtual reality systems for more than a decade (e.g., [1]). There are also a few commercial *Computer-Aided Design* (CAD) systems for industrial products fabricated from planar sheets. However, industrial designers rarely use the 3D tools provided in these systems as they do not agree with the habit of designers. In this paper, we describe what tools are expected by designers in relevant industries, review existing methods, and analyze why the existing tools cannot satisfy designers.

A. Tools Expected by Designers

The tools expected by designers in cloth industry can be classified into three groups.

1) Freehand styling design: The most natural way for designers to create new fashion is to have a freehand drawing tool like drawing on a piece of paper with a pencil. This is a reason why the sketch-based interface becomes so popular in many 3D computer graphics applications. Moreover, designers also want to have the curve creating and editing tools such as the widely used spline curves in CAD systems since these tools can define the styling shape more precisely. The final shape of a designed product will be specified by the styling curves (see Fig.1(a) for the application in the design of wetsuit). Such tools have been recently provided in some CAD systems (e.g., [2]). However, the freehand styling design tools are seldom used without the support of design transformation and unfolding solutions.

Manuscript prepared in September, 2008; revision prepared in June, 2009.
All the authors are with the Department of Mechanical and Automation Engineering Department, Chinese University of Hong Kong, Shatin, NT, Hong Kong. E-mail: cwang@mae.cuhk.edu.hk; Fax: (852) 2609 8052.

2) Design transformation: After carefully designing the shape of a product around one reference model, it is wished to have an automated tool to transfer this product to various reference model shapes. One styling design can then be repeatedly employed to generate products while the re-generated products fit the various shape of reference models (e.g., human bodies). Figure 1(b) gives an example of such a design transformation for a wetsuit product.

3) Generation of precise planar pieces: The final products in sheet manufacturing industries are generally fabricated from two-dimensional pieces of materials. During fabrication, the 2D pieces are warped and stitched together to build the final product. Ideally, the warping and the stitching should be stretch-free since a stretch will produce an elastic energy in the final product which debases the fitness, creates unwanted wrinkles, and even leads to material fatigue. Therefore, designers need the solution for unfolding given 3D surface patches into 2D pieces with invariant lengths at prescribed feature curves and boundaries. An example is shown in Fig.1.

B. Existing Methods

In order to let designers have better interfaces for styling design, techniques have been developed in the community of computer graphics to provide sketch-based styling design tools for toy industry and cloth industry. Turquin et al. developed a virtual garment design system in [3], where the 3D shape for a garment is constructed from contour lines drawn by designers in a sketch-based interface. The contour lines are classified as boundary if they cross the virtual 3D human body or as silhouette if they do not. Distance field around the mannequin is employed to determine the 3D position of contour curves, and the final 3D mesh surface is reconstructed from these curves while mimicking cloth tension and modelling surface folds. However, they did not address the problem of how to convert 3D virtual clothes into 2D pieces to fabricate the real cloth products. Mori and Igarashi recently introduced a sketch-based design system for plush toys (ref. [4]). The modelling tools are similar to their previous famous Teddy system in [5]. Another sketch-based modelling system for creating 3D-shape from industrial styling design was described in [6].

For the plush toy design system introduced in [4], the final 2D pieces to produce the plush toys are unfolded from its 3D shape by the ABF++ algorithm in [7]. Nevertheless, as we know that the ABF++ algorithm obtains the optimal parameterization by minimizing the angle distortion of polygons, the length of curves on surface pieces is *not* preserved. It is very problematic when using 2D patterns determined in this

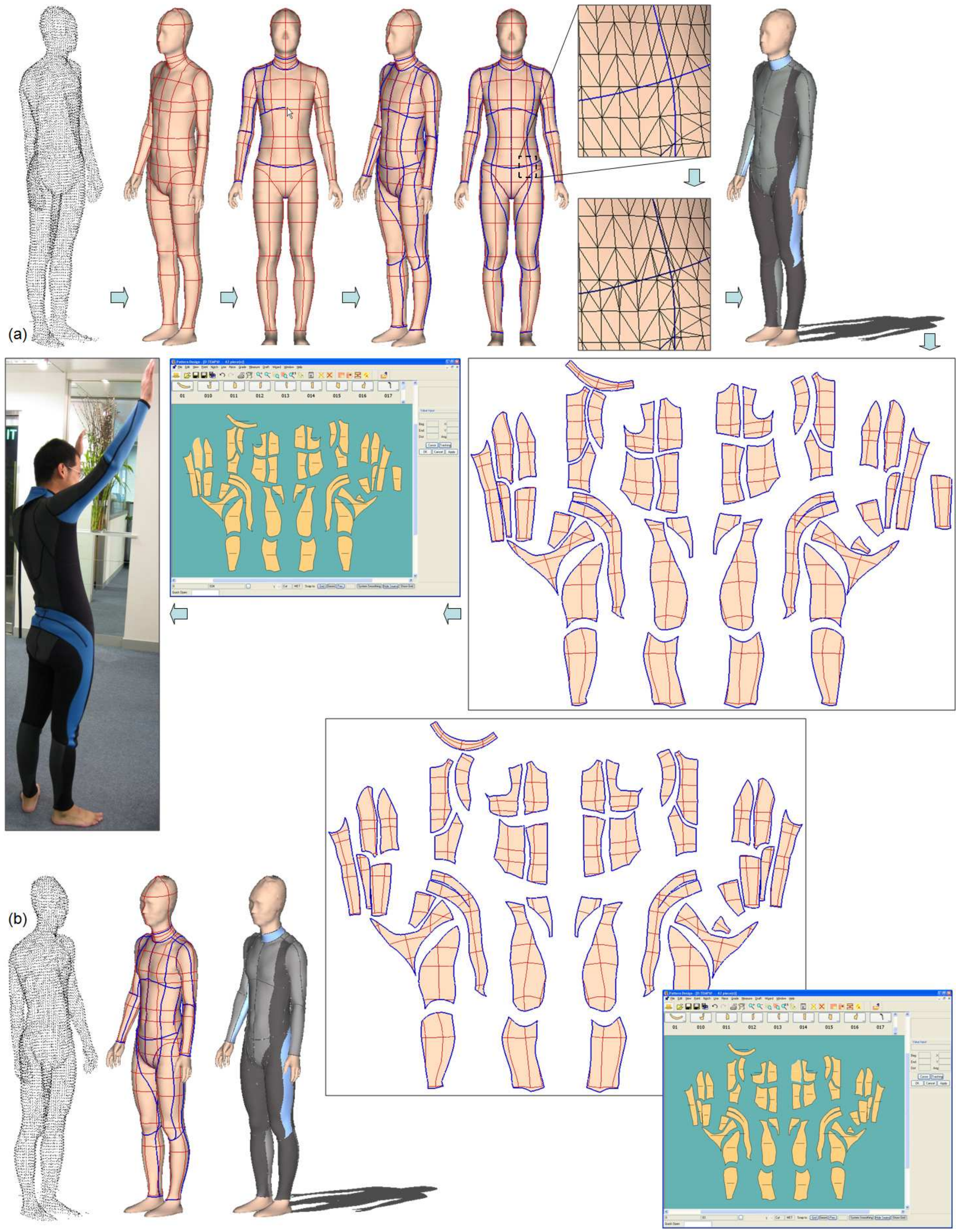


Fig. 1. Overview of application in wetsuit design and fabrication. (a) Steps from styling design to fabricated wetsuit on human model H_A: after using freehand drawing tools to get the styling design on the mesh surface generated from scanned human body point-cloud, the surface of a wetsuit is trimmed into 3D pieces, unfolded into 2D pieces and then output to commercial 2D Garment CAD software in DXF file standard; finally, the real wetsuit customized for H_A is fabricated from these 2D patterns. (b) Design transformation – the styling design of wetsuit on H_A can be easily transferred to the human model H_B to generate planar pieces for industrial fabrication.

way to fabricate high-accuracy products like wetsuit. Firstly, the dimension of 2D pieces and their corresponding 3D shape could be quite different. This is acceptable for applications like plush toys where elastic stretch is allowed. However, such a stretch on wetsuit makes the person wearing it feel very uncomfortable. Therefore, a new method needs to be developed to preserve the dimension of pieces in unfolding. More seriously, the length variation on the boundaries of pieces makes neighboring pieces hard to be stitched together. Sewing boundaries with different lengths together produces unwanted wrinkles, which is considered in the cloth industry to distinguish the quality of a fabric product – thus must be prevented. In order to solve a similar problem, Decaudin et al. in [8] employed a deformation process to process a given mesh surface so that it locally approximates a conical surface. It is a sufficient (but not necessary) condition for a conical mesh surface to be flattenable (or discrete developable). Moreover, for products like wetsuit and shoes, customers prefer their shape to be as-close-as-possible to the human body. Directly applying the techniques in [8] to every piece will make the products unfit. New techniques are requested to preserve the dimension of unfolded pieces at certain places or to process the surface to be more easily flattened while minimizing deformation.

According to the aforementioned transformation tool, it can be implemented based on the template warping based compatible mesh generation (ref. [9]) and the freeform deformation technique (ref. [9], [10]). Once all reference models H_j $(j = 1, \cdots, n)$ are with the same mesh connectivity, we can easily apply the freeform deformation to warp the product designed around H_i to the shape around H_j $(i \neq j)$.

C. Conventional Fabrication of User Customized Products

At present, without the help of CAD tools, user customized products are fabricated in the conventional tailor making manner. As shown in the top-left of Fig.2, a loosely fit wetsuit is firstly produced for a customer, and then revision on the wetsuit is tailored through trial and error to remove the unwanted fabric. However, wetsuits fabricated in this way is inaccurate and time-consuming because it is not easy to convert a freehand styling design into a final product, and whether the revised wetsuit fits the customer's body does very much rely on the skills of tailors. Besides, design transformation is unavailable in the conventional method. To overcome these difficulties, we have developed a system to generate 2D pieces for fabricating user customized products from 3D styling deign.

II. Overview

The diagram in Fig.1 shows how our system generates the 2D pieces from styling design for the fabrication of products. To generate the 2D patterns from a styling design, our system provides geometric modelling tools in five aspects below.

- *Styling Design:* The freehand drawing tools are provided to enable designers to specify styling curves on the mesh surface of a reference model. Users are also able to snap points on a styling curve to drag it into a desired shape. Of course, similar to other drawing based design systems, tools like mirror copying, curves trimming and erasing are provided too. The designed styling curves are stored as an attachment of the reference model's polygonal mesh surface so that it can be further adjusted and used later.
- *Trimming:* After confirming the styling design, the mesh surface of the reference model is duplicated and trimmed into separate 3D mesh pieces by the styling curves. The feature curve information (i.e., the red curves on mannequin in Fig.1) must be transferred to the newly generated 3D mesh patches as it plays an important role in the following unfolding procedure.
- *Unfolding and Geometry Processing:* The 3D pieces are unfolded into planar pieces while the lengths of feature curves and boundary curves remain invariant. Therefore, the dimension in 3D is preserved and there will be no sewing problem between neighboring patches. However, for some patches, it is not possible to retain the length of all feature curves in flattening as they are highly curved and unflattenable. A geometry processing tool has been developed to deform such patches into a similar shape but being more flattenable.
- *Layout Arrangement:* It is unacceptable to industrial users if the unfolded 2D pieces are randomly oriented and placed. To solve this problem, we develop a map-guided layout arrangement method so that the position and orientation of flattened patterns can be determined according to the guiding map. The resultant layout is output into a DXF file according to the standard of Garment CAD systems to generate further industrial processing information which is used to fabricate the final product.
- *Design Transformation:* Once the point-to-point correspondence between reference models is found, we can very easily transform the styling design from one model into another. Then, the industry required 2D patterns for

Fig. 2. Wetsuit produced by conventional tailor-making (top-left and middle column) vs. wetsuit fabricated from 2D pieces generated by our system (bottom-left and right column) – it is easy to find that the conventional wetsuit is a planar design but the wetsuit made by our system is a real spatial one.

another user can be generated (see Fig.1(b)).

III. Implementation

A. Styling Design and Its Transformation

As analyzed above, the most popular tool for designers is the freehand drawing tool. The curve creating tool of our system allows users to specify some points of the curve (i.e., control points). Then, a subdivision curve passing through these control points is generated by refining the line segments linking control points three times – using the *Modified Butterfly Mask* in [11]. Here the curve is treated as a 2D curve on the viewing plane. The control points are allowed to be moved to change the curve's shape. After construction, the refined data points on the curve are projected onto the triangular mesh surface of the reference model and linked by discrete geodesic curves. The projected styling curve is stored as a sequence of line segments in triangles together with the control points for its construction.

As the viewing direction for further editing a curve may not be the same as the one creating it, the curve editing tool is developed in a different way – we process the styling curve in 3D now. First of all, the portion to be edited on a curve is specified. The original control points of this partial curve are employed as control points for further editing. If no control point is found (e.g., the part to be edited is just between two control points during construction), we uniformly sample the curve into four to six points by its arc length and employ these sample points as control points. Then, users can move the control points to adjust the curve. During movement, the control points and the refined data points are snapped onto the surface of the reference model. This is implemented by tracking these points to their closest point on the mesh surface in real-time with the help of space partition technique. The shape of the edited curve is again computed by the the *Modified Butterfly Mask* but in 3D – so there may be interference between the curve and the mannequin, which can be eliminated by linking data points on the surface of the reference model with discrete geodesic curves. Figure 3 gives an illustration of this 3D curve editing tool.

The styling design stored on a reference model H_A can be easily transferred onto another reference model (e.g., H_B) as long as the bijective point-to-point mapping between H_A and H_B has been established. More specifically, we determine the corresponding position on H_B for every data point of styling curves on H_A using this mapping, and link the transferred data points by discrete geodesic curves. One example of such a transformation can be found in Fig.1(b). In our implementation, we employed a template fitting based method as [9] to generate fully compatible meshes among scanned human bodies. Not only the styling curves but also the feature curves (e.g., the red ones in Fig.1) can be transferred in this way.

B. Trimming

The 3D pieces of products are generated by trimming duplicated mesh surface using the styling curves. For the mesh surface M of a reference model, the triangles with no styling curve are duplicated firstly into a new triangular mesh model

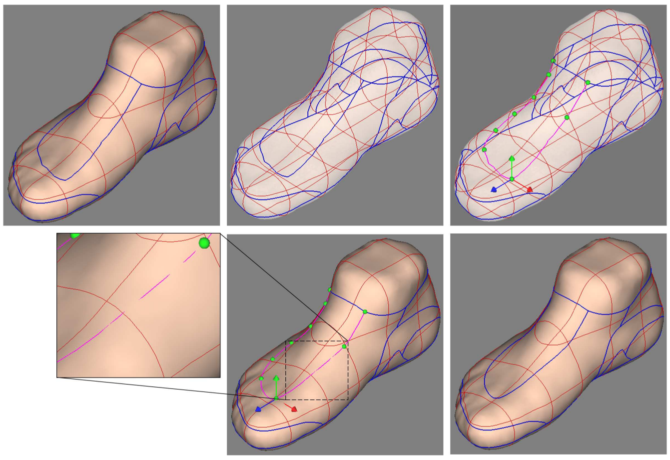


Fig. 3. The editing of styling curves is conducted in 3D: (top-left) the curve to be edited on the foot, (top-middle) the curve cannot be edited in 2D manner as part of it is at the back of the foot surface under the current view, (top-right) the curve has been edited by dragging control points, (bottom-left) interference between the 3D curve and the mannequin occurs as only data points are snapped onto the human's foot, and (bottom-right) the interference is eliminated by linking snapped data points with discrete geodesic curves.

M'. Then, the triangles with styling curves are re-triangulated one by one using the line segments of styling curves. The *Constrained Delaunay Triangulation* (CDT) is employed to carry out triangulation on the plane of these triangles. To make CDT robust to numerical errors, we process the line segments inside a triangle by

1) finding the intersections between line segments;
2) inserting intersections between line segments and the triangle edges;
3) eliminating nearly overlapped (or partially overlapped) line segments;
4) merging nearly coincident points.

This is similar to the method presented in [12]. The resultant triangles of CDT are attached onto the duplicated mesh model M', and the information of whether a triangle edge is derived from a styling curve (or a feature curve) on M can be easily retrieved through local search using the correspondences of triangles between M' and M. Note that if the feature curves on M are not presented by triangle edges, they will also be introduced into CDT to become triangle edges on M. Lastly, the triangle edges generated from styling curves are duplicated so that the regions on M' surrounded by these edges are separated into pieces of mesh patches.

C. Unfolding with Length-Preserved Feature Curves

The 3D pieces of a wetsuit must be unfolded into planar patterns to be used in industrial fabrication. In order to generate precise planar patterns that keep the dimension at critical places, we developed an unfolding algorithm named as *WireWarping* that preserves the length of feature curves.

The basic idea of *WireWarping* is to simulate the physical phenomenon of warping tendon wires from a 3D wireframe onto plane, wherein the wires can only be warped but not stretched so that their lengths can be retained. Here, the feature curves and the boundaries of 3D pieces serve as the tendon

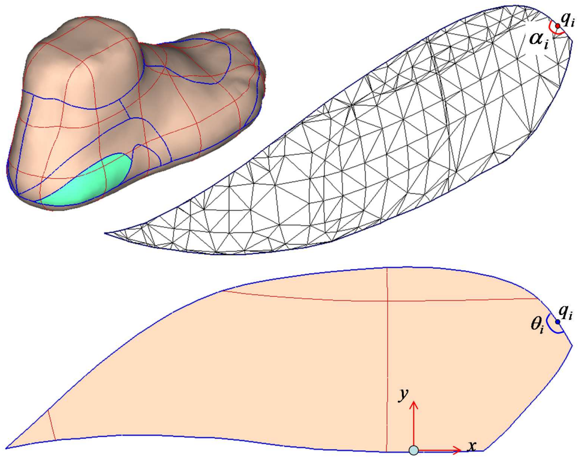


Fig. 4. An illustration of warping the boundary of a given mesh patch into an optimal planar shape while preserving the length of edges on it.

wires. Let us first consider about how to warp the simplest wireframe – the boundary curve of a mesh surface patch M_P (as shown in the top of Fig.4), which consists of n triangular edges. There are many possible ways to warp such a simple wire into different 2D polygons without varying its length. Among all these possible 2D polygons, we seek one which is most similar to the 3D shape of M_P's boundary. Therefore, the one which minimizes the sum of angle variation at all boundary nodes is what we want, i.e.,

$$\min_{q_i} \sum_{i=1}^{n} (\theta_i - \alpha_i)^2 \tag{1}$$

where θ_i is the 2D angle associated with the boundary vertex q_i, α_i represents its 3D surface angle on M_P that is computed by adding all triangles' angle at q_i, and there are n boundary vertices on M_P. Since we need to preserve the length of boundary edges during the warping, three constraints are added to ensure this.

1) From the *closed-path theorem*, we know that: for a simple non-self-intersecting planar closed path, if its path is anti-clockwise, the total turning must be 2π.
2) When keeping the length of boundary edges unchanged, the 2D curve should still be a closed one – i.e., with the determined angle θ_i, the first point on the curve must have the same x- and y- coordinates as that of the last point on the curve (see the right of Fig.4).

Hard constraints derived from these are added into the optimization framework to compute the optimal values of θ_i. Note that, to efficiently solve the constrained optimization problem in Eq.(1), we compute optimal θ_is instead of q_is.

A 3D piece to be unfolded is usually subdivided into several such simple wireframes by feature curves and boundary curves. If we use the above method to unfold them one by one, the resultant 2D wires in general cannot be stitched together without distortion. To solve this incompatible problem, for each vertex $\mathbf{v}$ on feature curve which is shared by m simple wireframes M_{P_j} $(j = 1, \cdots, m)$, the hard constraint $\sum_{j=1}^{m} \theta_{M_{P_j}}(\mathbf{v}) \equiv 2\pi$ is added if v is not on the boundary of the 3D piece. $\theta_{M_{P_j}}$ denotes the 2D angle at $\mathbf{v}$ in the warped simple wireframe M_{P_j}. Integrating all wireframes on the given 3D piece into one optimization framework by adding these constraints gives the unfolded results with the lengths of feature curves and boundaries preserved. More details can be found in [13].

The planar positions of vertices *neither* on the feature curves *nor* on the boundary curves can be determined by the shape-preserved mesh parameterization method in [14] after fixing the vertices on wires.

D. Discrete Developable Mesh Processing

Nearly developable polygonal mesh surfaces can be successfully unfolded into planar pieces using the *WireWarping* methods introduced in previous subsection. Here, the *discrete developable* is derived from the definition of developable surface in differential geometry as: a polygonal mesh surface is said to be discrete developable if the sum of polygonal angles at each non-boundary vertex is 2π. A nearly developable mesh surface means the angle sum at non-boundary vertices are close to 2π. As those mesh surfaces far from discrete developable will create great area errors on resultant 2D pieces generated by *WireWarping*, we developed a tool to process a given undevelopable mesh surface M_P into a nearly developable one, M_P^F. The processed mesh surface M_P^F must approximate the shape of M_P. This is again formulated as a constrained optimization problem,

$$\min_{p \in V_{int}} J_d \quad s.t. \; \alpha(\mathbf{v}_p) \equiv 2\pi \tag{2}$$

where J_d is the objective function that measures the deformation between M_P and M_P^F, V_{int} is the set of non-boundary vertices on M_P, and $\alpha(\mathbf{v}_p)$ gives the summed polygon angles at the vertex $\mathbf{v}_p$. Boundary vertices are fixed so that the assembly relationship between neighboring pieces will not be broken. The most common method to measure the deformation is to employ $J_d = \sum_p \|\mathbf{v}_p - \mathbf{v}_p^0\|^2$ with $\mathbf{v}_p^0$ being the closest point of $\mathbf{v}_p$ on M_P. However, as analyzed in [15], this objective function prevents the movement of vertices in some sense. Therefore, a least-norm approach was proposed to solve this constrained optimization problem.

Firstly, the nonlinear developable constraints $\alpha(\mathbf{v}_p) \equiv 2\pi$ are linearized by using the Taylor expansion and truncating the nonlinear terms. Then, the unknown variables $\mathbf{v}_p$ are replaced by $\mathbf{d}_p$ with $\mathbf{v}_p^0 = \mathbf{v}_p + \mathbf{d}_p$. There are $3|V_{int}|$ unknown variables but only $|V_{int}|$ linear equations derived from the developable constraints (here, $|\cdots|$ denoting the number of elements in a set). Thus, the least-norm method can be applied to determine the solution of $\mathbf{d} = \{\mathbf{d}_p\}$ with the minimal norm $\|\mathbf{d}\|$. The above steps are then repeated until the deformed mesh surface M_P^F becomes nearly developable. Furthermore, to make the numerical computation more stable, the least-square methods are used to update the position of vertices – details can be found in [15].

In practice, the users may feel that using M_P^F has changed the shape of a 3D piece too much. They would like to use an interpolation between M_P and M_P^F as $M_P^t = (1-t)M_P + tM_P^F$ with $t \in (0, 1)$ as long as the area error of unfolded M_P^t is below an acceptable ratio compared with its 3D surface. As

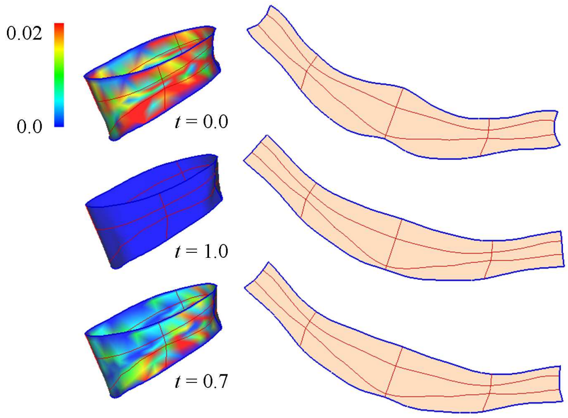


Fig. 5. An example of discrete developable mesh processing: (top) the given collar piece M_P, (middle) the fully processed mesh surface M_P^F, and (bottom) the interpolated mesh M_P^t with $t = 0.7$. Colors denote how the value of summed angle at a non-boundary vertex is different from 2π.

shown in Fig.5, the collar patch gives a large area error before discrete developable mesh processing but a small error on M_P^t with $t = 0.7$. Note that the processed mesh surfaces are also unfolded by the *WireWarping* method.

E. Map-guided Layout Arrangement

The randomly placed 2D pieces are in a tangle. This can hardly be used in the downstream applications. Lévy et al. in [16] employed a "Tetris" game like method to lay out the 2D pieces. However, the industrial users still find it difficult to decide where a 2D piece of wetsuit should be located in 3D on the human body. Therefore, a map-guided layout arrangement algorithm is developed to cope with it.

A guiding map is in fact the parameterized planar mesh M_G of a reference model H_G (e.g., the guiding map of human body shown in Fig.6(a)) – here the method in [16] is employed. Therefore, for any other reference model H_A, the corresponding point on M_G of a point on H_A can be easily found once the cross-parameterization between H_A and H_G has been established. Such mapping is denoted by $\Omega_{H_A \Leftrightarrow M_G}$.

For a flattened mesh piece M_P for the wetsuit of mannequin H_A, the bijective mapping from any point on M_P to a point on H_A is denoted by $\Omega_{M_P \Leftrightarrow H_A}$ and can be retained in the above modelling steps of trimming and unfolding. By $\Omega_{H_A \Leftrightarrow M_G}$ and $\Omega_{M_P \Leftrightarrow H_A}$, the bijective mapping $\Omega_{M_P \Leftrightarrow M_G}$ from M_P to the guiding map can be obtained. Using the method in [17], the rotation matrix $\hat{\mathbf{R}}$ and the translation vector $\hat{\mathbf{T}}$ from $\{p_k | \forall p_k \in M_P\}$ to $\{p'_k | p'_k = \Omega_{M_P \Leftrightarrow M_G}(p_k)\}$ can be computed by least-square fitting. Therefore, the locations and orientations of all 2D pieces according to the guiding map are determined. However, as the guiding map consists of a few separate regions, the points on flattened pieces may be mapped to different regions, which will lead to an unsatisfactory result. For a piece M_P whose points are mapped to different regions, the points are clustered into different sets according to their mapped region. Only the points in the set with maximal number of points are used to determine $\hat{\mathbf{R}}$ and $\hat{\mathbf{T}}$.

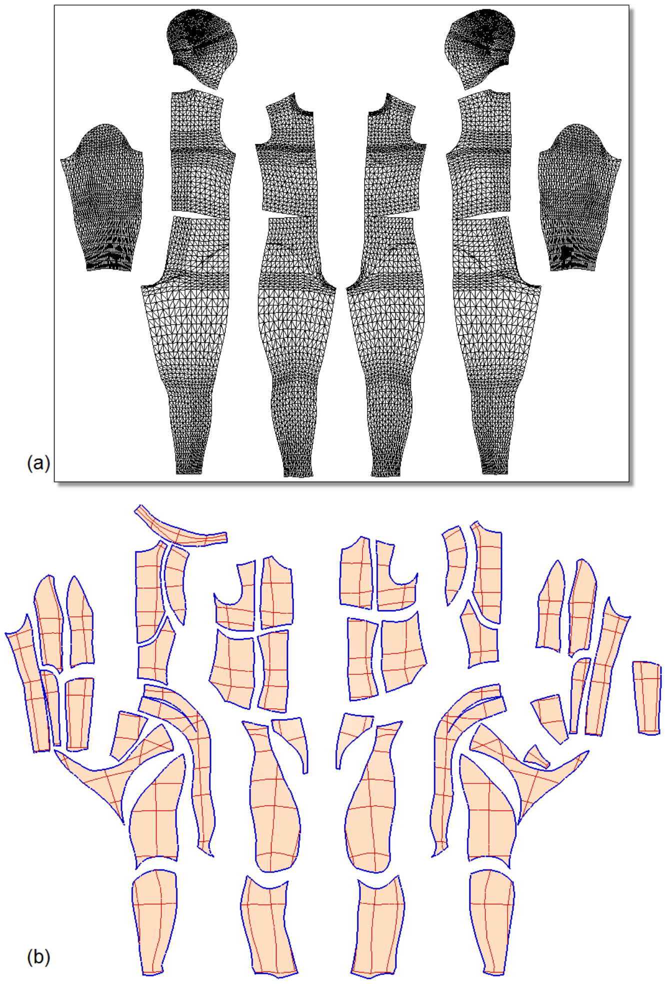


Fig. 6. Map-guided layout arrangement: (a) the guiding map and (b) the final layout with overlaps eliminated.

Overlap happens when using the above determined $\hat{\mathbf{R}}$ and $\hat{\mathbf{T}}$ to place planar patterns. The initially placed patterns are then moved iteratively to eliminate overlaps. To speed up the overlapping detection, we rasterize the planar patterns into 2D boolean arrays using the off-screen rendering. By our experiments, rasterization at the resolution of three pixels for one centimeter balances the computing speed and the accuracy. Furthermore, to retain the symmetrical layout given by the guiding map, the initially placed patterns are classified into left and right groups. The patterns in the left group are moved leftwards or upwards while the ones in right are moved rightwards or upwards to eliminate overlaps. Before that, all patterns in both groups are sorted by the vertical coordinate of their bounding boxes' bottom edges ascendingly. For two patterns having the same vertical coordinate, their horizontal coordinates (of left or right edges) are used. At the beginning, all pieces are considered as "movable" except the lowest one which has been fixed by its current position. The first piece in the sorted movable piece-list is then removed from the list and moved to a position where no overlap is found with other fixed ones and this process repeats iteratively. When moving a piece to eliminate overlap, it is either moved upwards or horizontally away from the medial-axis. The direction with a

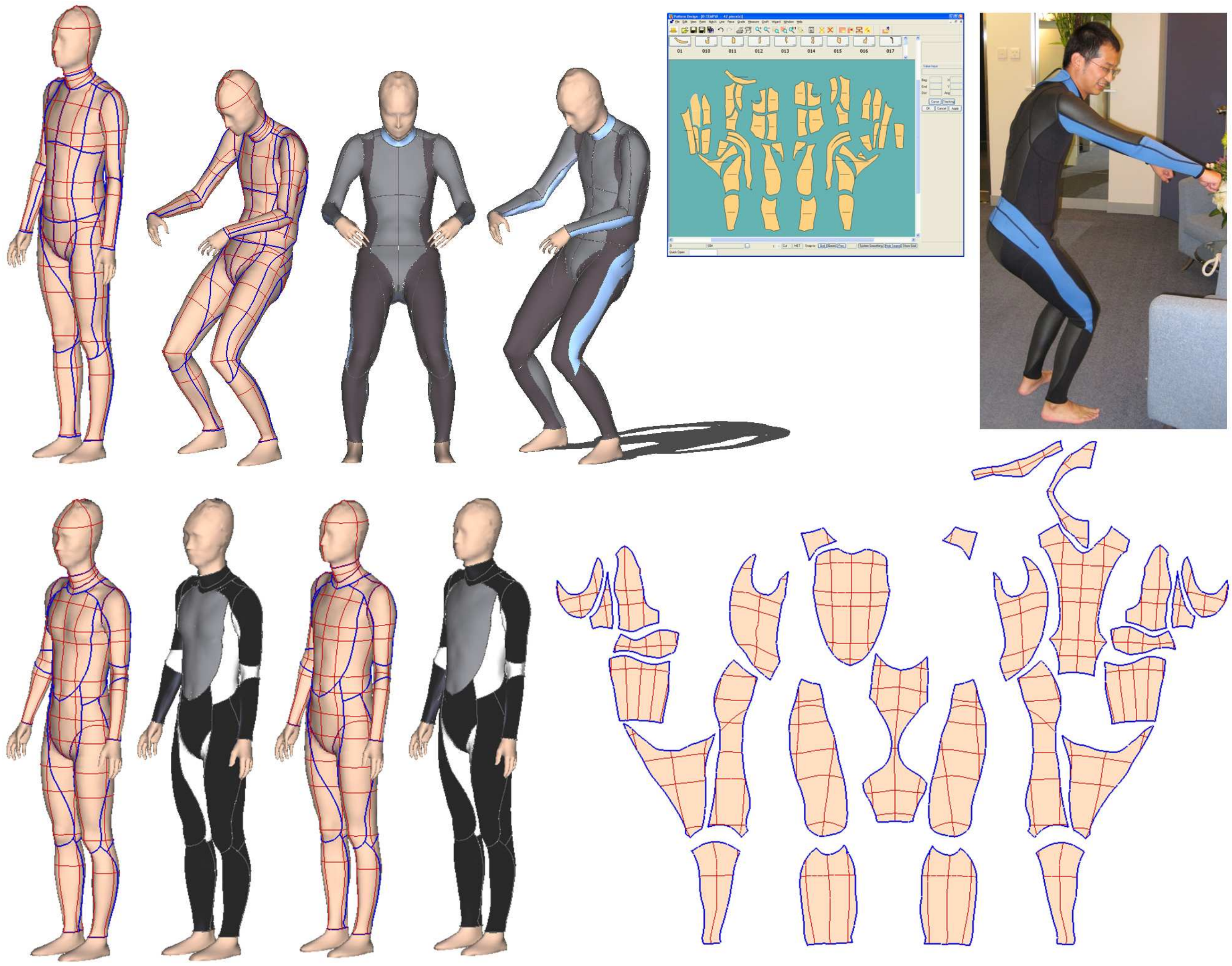


Fig. 7. More examples: (top) the patterns and wetsuit for styling design I with different postures are generated and (bottom) styling design II.

smaller movement is chosen. Note that all movable patterns are moved in the same direction and with the same extent as that of the piece. The iteration stops when the list becomes empty. An example of a final layout has been shown in Fig.6(b).

IV. Applications

We have implemented the proposed algorithms in a prototype system by Visual C++ and OpenGL. SuperLU wrapped by OpenNL is conducted as the numerical computation kernel. The computation from a styling design to the patterns to be fabricated can be completed in less than one minute on a desktop PC with the standard configuration. We have tested the prototype system in several applications.

Our modeling system is motivated by an industrial project whose purpose is to automatically generate planar patterns for user customized wetsuit from different styling designs. Previous examples shown in Fig.1 demonstrate the functions of our system. In this application, the reference models are human bodies, and the styling design is conducted on the human models that are presented by two-manifold triangular mesh surfaces. More examples of the wetsuit application are shown in Fig.7, where the top row shows the patterns and wetsuit for the styling design I that has been shown in Fig.1 but with different postures. As the posture of the human body is changed, the mesh surfaces of wetsuit at some regions like the elbow, the knee and the hip become far from nearly developable. Therefore, the discrete developable mesh processing is applied to the 3D pieces at these regions. The final fabricated wetsuit is shown at the rightmost end of the top row in Fig.7. The bottom row of Fig.7 shows the wetsuits with different styling designs and on different human bodies.

We also apply the prototype system to shoe design. Different from the application of wetsuit design and fabrication, the styling design of shoes is conducted on the shoe last instead of the surface of scanned foot. The surface of shoe last for a foot is a smooth offset surface of the foot model (see Fig.8 for the example). After designing the styling curves on the shoe last's surface M_A for the reference model, the foot F_A, we can easily warp the surface M_A to the shape fitting a new foot model F_B as long as we have the bijective mapping between F_A and F_B established. Then, the planar patterns for the fabrication of user customized shoes can be generated by our system. Also, the guiding map of layout arrangement can be produced by the mesh parameterization technique, and the

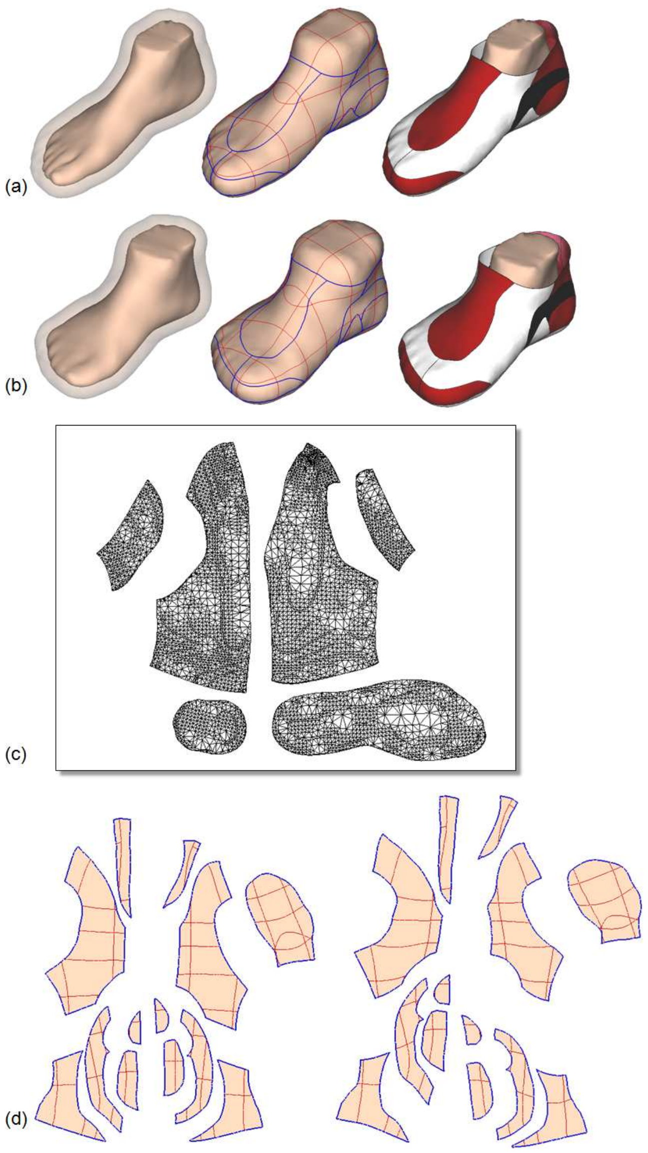


Fig. 8. Using our system in the design automation of shoes. (a) The foot with skin F_A (left), the styling design curves (middle), and the shaded view of designed shoe (right). (b) The automatically generated design on the foot F_B. (c) The guiding map. (d) The 2D layouts of shoes for F_A (left) and F_B (right).

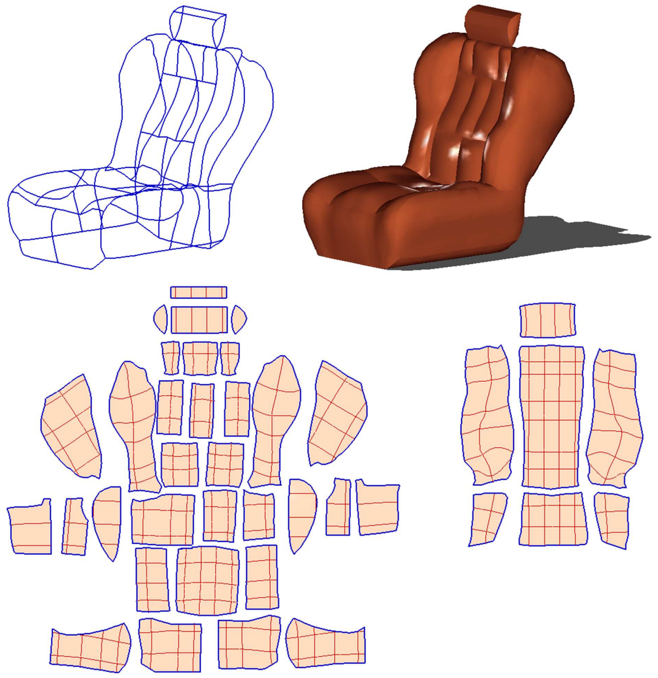


Fig. 9. An example application of designing a furniture: (top-left) the wireframe of the designed sofa-chair, (top-right) the 3D surface of sofa-chair constructed by the approach of [6], and (bottom) the layout of 2D pieces generated by our system for the fabrication.

layout of 2D pieces is automatically arranged according to the guiding map. Figure 8 gives an example of the use of our system in shoe design.

The third application tested on our system is the generation of planar pieces for furniture. Here we adopt the sketch-based modeling method to design a sofa-chair by the approach presented in [6]. Several feature curves are then defined using the tools provided in our system at the places where the dimensions (i.e., lengths) must be controlled. The *WireWarping* algorithm adopted in our system can well preserve their lengths on the resultant patterns – see Fig.9. The final product can be fabricated from the leather sheets in the determined planar shape.

When applying our system to the design of clothes – i.e., general ones unlike the tight wetsuit, it is more or less similar to the application in shoe design. Instead of shoe last, the basic blocks of clothes are employed in this application. In the apparel industry, the general basic blocks include blazers, shirts, jackets, trousers, Jeans-pants, polo-shirts, dresses and skirts, which define the structure and spatial relationship with human bodies. The techniques for building such basic blocks have been studied in our previous work (ref. [18]). After obtaining the surface M_A of a basic block for a human body H_A, we can specify the styling design curves on it and transform them to the shape around another human body H_B by the spatial constraints defined between basic blocks and human bodies. For example, in Fig.10, a new dress is automatically generated around the human body H_B. The planar patterns can then be determined by preserving the lengths of feature curves defined on the basic blocks – therefore, the spaces between the clothes and human bodies are guaranteed to fit.

The last application tested in our study is the generation of 2D pieces for the fabrication of fabric toys. In this application, the designers wish to have a tool to help them analyze the stretch distribution after designing a 3D model and specifying the cutting lines. Figure 11 provides such an example. As the computation of our unfolding results is very fast, after specifying the cutting curves, users can get the error in terms of area difference on 2D versus 3D interactively. The color-map in Fig.11 is an example. Therefore, it is easier for users to add some darts to reduce the stretch in fabrication. Note that a large area distortion between 2D and 3D patterns will make the fabrication of final products difficult, especially when almost inextensible sheet materials are used in manufacturing. At the bottom of Fig.11, the fabric squirrel toy can be fabricated from the pattern generated by our approach. As fewer pieces are

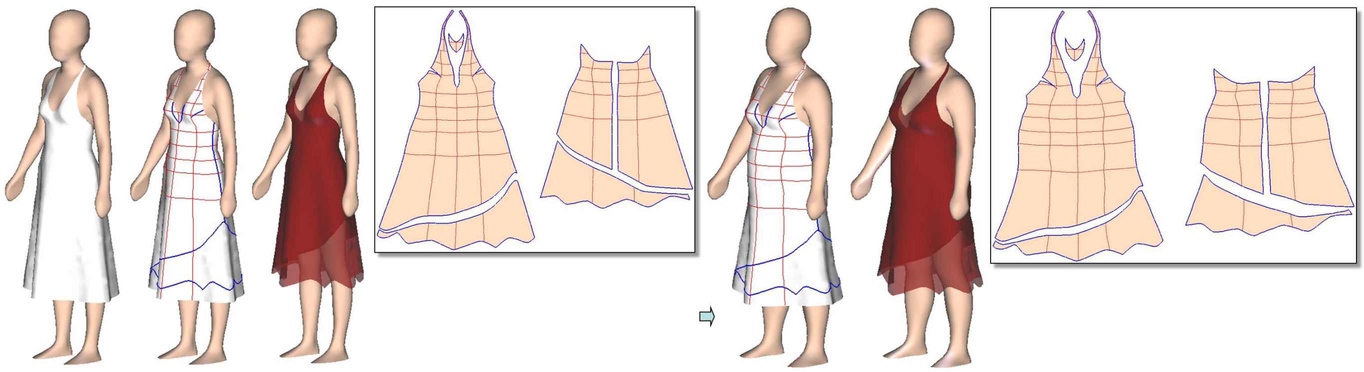


Fig. 10. A new styling design is conducted on the basic block of evening dress around the human body H_A (left), and the design is transferred to a new dress for another body H_B (right). Their corresponding planar patterns are also given.

generated by our approach comparing to [5], the fabrication time could be greatly reduced.

V. User experiences

We have licensed the developed prototype software to an industrial company whose major business is user-customized wetsuit. Feedback from the industrial usage of our system is very positive. The users are very satisfied with the ease use of our freehand styling design tools, the design transformation tools and the function of generating precise planar patterns. By using the newly developed system, the design/manufacturing cycle of the user-customized wetsuit has been reduced from around one week to one day. A well-trained designer can produce a new styling design and the corresponding 2D patterns in around 15 to 30 minutes on average. Even for novices, after taking a half-day training course, they can generate a new styling design and the 2D patterns in less than one hour. The planar patterns generated by our system can be directly imported into the downstream CAD systems to generate the seam allowance and printed out for cutting of fabrics.

In addition, the users from industries also point out some drawbacks of our current implementation. After transforming the styling design between human bodies with greatly varied shapes, the smoothness of styling curves cannot be preserved. More specifically, the S-shape distortion may be generated on originally smooth curves (e.g., the one shown in Fig.12(top)). Currently, users need to select the unsatisfactory curves and adjust their shape by moving the control points interactively. A curve fairing function needs to be developed in the near future to solve this problem – note that, different from general 3D spatial curve fairing, the difficulty here is to conduct curve fairing on a 3D freeform surface.

The second expected function reported by industrial users is the symmetric unfolding. After user specifying the styling curves on a reference model, the 3D patterns generated by trimming and re-triangulation may become asymmetric in topology (i.e., mesh connectivity) even if the styling curves are specified symmetrically and the 3D shape is also symmetric. Therefore, the unfolded 2D patterns cannot be exactly symmetric. Figure 12(bottom) gives an illustration of such a case. Our current implementation solves such a problem by a post-processing in 2D. For such a 2D piece P, we first get its mirrored 2D pattern $\tilde{P}$, and then the average 2D piece of P and $\tilde{P}$ is used as the result. Here, the average is computed by the morphing technique using length-parameterization on 2D polygons. An example of the resultant symmetric pattern can also be found in Fig.12(bottom).

Although the *WireWarping* algorithm can prevent local self-intersection of an unfolded patch, it cannot prevent the global self-intersection. Therefore, the global self-intersection is detected by a post-process in the image space and reported. However, we leave the problem of how to generate a new cut to eliminate the self-intersection to users. Users need to add the new darts by the styling design tools. The method of generating such darts automatically is considered as a future work of our research.

VI. Conclusion

In this article, we introduce an application of using the state-of-the-art geometric modelling techniques in the business of user-customized products fabricated from sheet materials. Based on the analyzed requests from industrial designers, several tools have been developed. The implementation methods have been detailed. The fabricated products have been shown to verify the functionality of these tools. Moreover, the experiences of real industrial users have been reported.

Acknowledgment

This work was partially supported by the Hong Kong Innovation and Technology Fund (ITF) ITS/026/07 and Shun Hing Institute of Advanced Engineering (SHIAE) Research Grant (CUHK/8115022), and the authors would like to thank TPC (HK) Ltd. for providing some testing data and Pu Huang for implementing the rendering program of our system.

References

[1] Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K., “Elastically deformable models,” *In Proceedings of ACM SIGGRAPH 87*, pp.205-214, 1987.

[2] Design Concept Furniture, Lectra, http://www.lectra.com/en/index.html.

[3] Turquin, E., Wither, J., Boissieux, L., and Cani, M.-P., “A sketch-based interface for clothing virtual characters,” *IEEE Computer Graphics and Application*, 27(1), pp.72-81, 2007.

[4] Mori, Y., and Igarashi, T., “Plushie: an interactive design system for plush toys,” *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, 23(3), Article No.45, 2007.

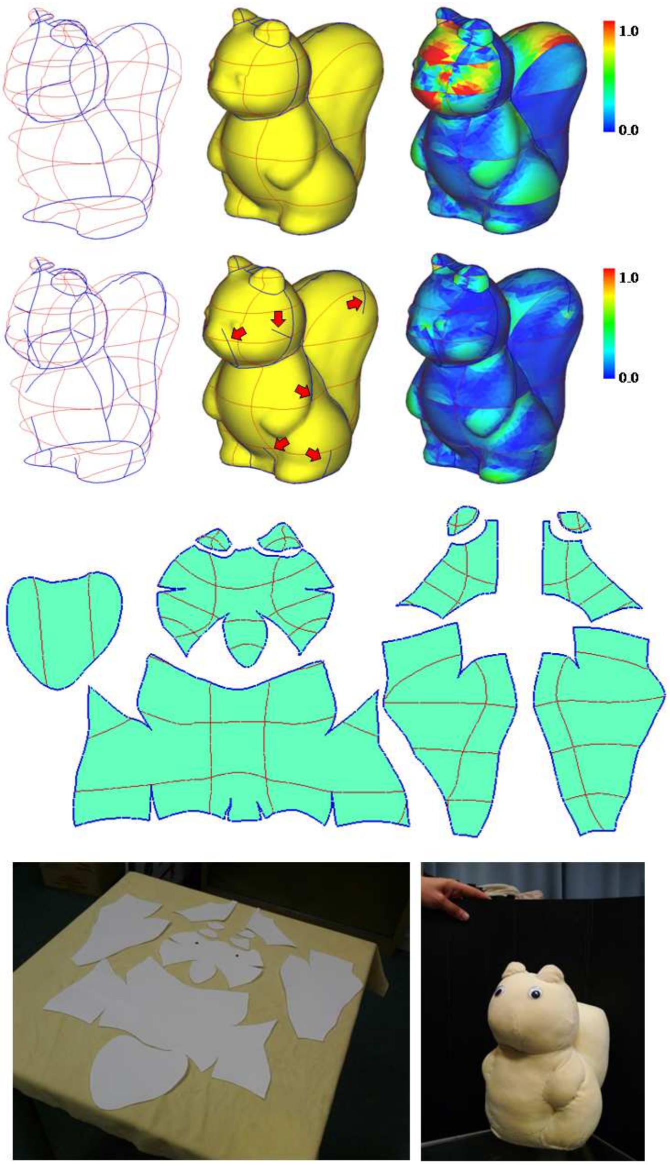


Fig. 11. Example of generating 2D patterns for a 3D toy squirrel. (Top row) Several cutting lines have been specified by users, and the stretches of area between 2D and 3D are displayed by color-map where red denotes 100% or even more area stretch while blue represents no area change on triangles. (Second row) After adding some darts (pointed at red arrows), the stretches have been reduced. (Third row) The flattened planar pieces for the toy squirrel and the final squirrel fabricated from these pieces. (Bottom row) The squirrel toy fabricated from the 2D pieces computed by our approach.

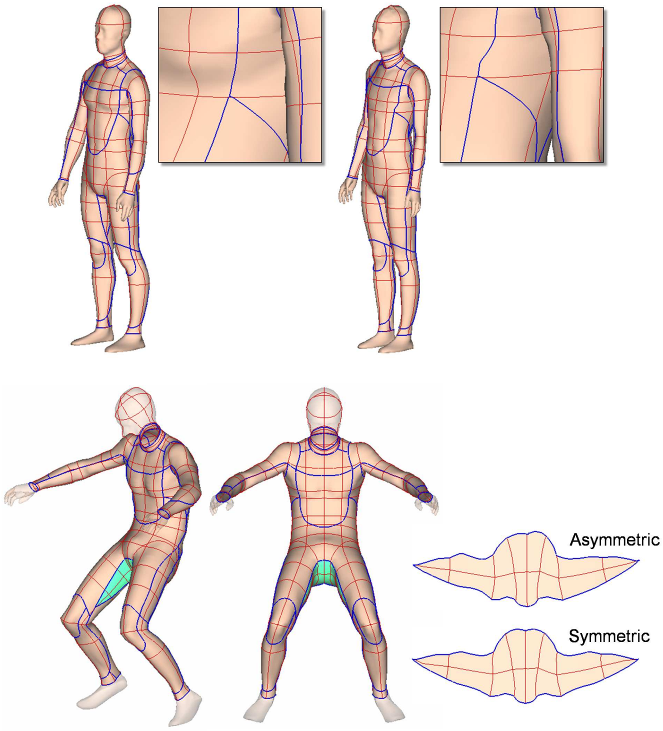


Fig. 12. After design transformation, some styling curves which are originally smooth (top-left) have been deformed into S-shape (top-right). Regarding the symmetry of a 3D piece at crotch, our *WireWarping* does not give a symmetric unfolding result (bottom), which however can be fixed by using a morphing technique to generate symmetric 2D patterns.

[5] Igarashi, T., Matsuoka, S., and Tanaka, H., "Teddy: a sketching interface for 3D freefrom design," *Proceedings of ACM SIGGRAPH 99*, pp.409-416, 1999.

[6] Kara, L.B., and Shimada, K., "Sketch-based 3D-shape creation for industrial styling design," *IEEE Computer Graphics and Application*, 27(1), pp.60-71, 2007.

[7] Sheffer, A., Lévy, B., Mogilnitsky, M., and Bogomyakov, A., "ABF++: fast and robust angle based flattening," *ACM Transactions on Graphics*, 24(2), pp.311-330, 2005.

[8] Decaudin, P., Julius, D., Wither, J., Boissieux, L., Sheffer, A., and Cani, M.-P., "Virtual garments: a fully geometric approach for clothing design," *Computer Graphics Forum (Eurographics'06 proc.)*, 25(3), 2006.

[9] Wang, C.C.L., Hui, K.C., and Tong, K.M., "Volume parameterization for design automation of customized free-form products," *IEEE Transactions on Automation Science and Engineering*, 4(1), pp.11-21, 2007.

[10] Kobayashi, K.G., and Ootsubo, K., "t-FFD: free-form deformation by using triangular mesh," *Proceedings of the eighth ACM symposium on Solid modeling and applications*, pp.226-234, 2003.

[11] 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.

[12] Mitani, J., "A simple-to-implement method for cutting a mesh model by a hand-drawn stroke," *In Proceedings of the 2nd EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, 2005.

[13] Wang, C.C.L., "WireWarping: A fast surface flattening approach with length-preserved feature curves," *Computer-Aided Design*, vol.40, no.3, pp.381-395, 2008.

[14] Floater, M.S., "Parametrization and smooth approximation of surface triangulations," *Computer Aided Geometric Design*, 14(3), pp.231-250, 1997.

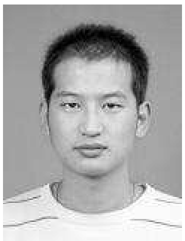
[15] Wang, C.C.L., "A least-norm approach to flattenable mesh surface processing," *Proceedings of IEEE International Conference on Shape Modeling and Applications 2008*, pp.131-138, 2008.

[16] Lévy, B., Petitjean, S., Ray, N., and Maillot, J., "Least squares conformal maps for automatic texture atlas generation," *ACM Transactions on Graphics (Proc. SIGGRAPH 2002)*, 21(3), pp.362-371, 2002.

[17] Shoemake, K., "Animating rotation with quaternion curves," *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, pp.245-254, 1985.

[18] Wang, C.C.L., Wang, Y., and Yuen, M.M.F., "Design automation for customized apparel products," *Computer-Aided Design*, 37(7), pp.675-691, 2005.

Charlie C.L. Wang is currently an Associate Professor at the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, where he began his academic career in 2003. He gained his B.Eng. (1998) in Mechatronics Engineering from Huazhong University of Science and Technology, M.Phil. (2000) and Ph.D. (2002) in Mechanical Engineering from The Hong Kong University of Science and Technology. He is a member of IEEE and ASME, and an executive committee member of Technical Committee on Computer-Aided Product and Process Development (CAPPD) of ASME. Dr. Wang has received a few awards including the ASME CIE Young Engineer Award (2009), the CUHK Vice-Chancellor's Exemplary Teaching Award (2008), and the Best Paper Awards of ASME CIE Conferences (in 2008 and 2001). His current research interests include geometric modeling in computer-aided design and manufacturing, biomedical engineering and computer graphics, as well as computational physics in virtual reality.



Yunbo Zhang is a Ph.D Student of Department of Mechanical and Automation Engineering at The Chinese University of Hong Kong. His research interests include CAD&CAM, physical based geometric modeling, and computer graphics. Zhang received his Mphil in digital design and manufacture from National Engineering Research Center for CAD, Huazhong University of Science and Technology.

Hoi Sheung received the MPhil degree from the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, in 2009. He is now a PhD student at The Chinese University of Hong Kong. His current research interests are mesh reconstruction, denoising, 3d scanning and parallel programming.