Computer Aided Geometric Design 29 (2012) 565–578

Contents lists available at SciVerse ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

Least squares quasi-developable mesh approximation

Long Zeng[a], Yong-Jin Liu[b,*], Ming Chen[c], Matthew Ming-Fai Yuen[a]

[a] *Department of Mechanical Engineering, Hong Kong University of Science and Technology, Hong Kong, China*
[b] *TNList, Department of Computer Science and Technology, Tsinghua University, Beijing, China*
[c] *Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China*

A R T I C L E   I N F O

Article history:
Available online 29 March 2012

Keywords:
Developable meshes
Least squares solution
Extended Gaussian image
Optimization

A B S T R A C T

Surface developability is a key feature in many industrial product designs. In this paper, based on a measure defined in an extended Gaussian image, an efficient least squares solution is proposed to achieve an optimal quasi-developable mesh patch. For a free-form mesh model assembled by a set of patches, each patch can be deformed using the least squares solution to obtain the best developability within a user-specified tolerance while the patch boundary remains continuous with neighboring patches. The proposed least squares scheme formulates the quasi-developable mesh approximation problem as a large sparse linear system with its coefficient matrix independent of the mesh vertices' new positions. We show that this linear system can be efficiently solved by a least squares direct matrix solver. Experimental results and applications are provided to demonstrate the controllability of shape change as well as the effectiveness of mesh developability improvement provided by the proposed solution.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

A developable surface can be unfolded onto a plane without any distortion. A necessary and sufficient condition that a surface be developable is that its Gaussian curvature vanishes over its entirety (Struik, 1988). In many industrial applications, the developable surface appearance is of great importance since stretch and compression should be minimized to decrease surface internal strain and stress.

In differential geometry (Struik, 1988), the Gauss map $G : S \to S^2$ continuously maps a surface S in R^3 to a Gaussian sphere S^2 (a unit sphere), which translates the tail of each point's normal to the center of the Gaussian sphere and the head onto the Gaussian sphere surface. $G(S)$ is called the Gaussian image of S. The Gaussian image of a developable surface is a spherical curve. In this paper, we propose an efficient least squares solution to improve the developability of a triangular mesh surface M using a *discrete extended Gaussian image* (DEGI for short). DEGI is defined to be a set of sample points on S^2, where each sample point is contributed by the unit normal of a triangle in M, weighted by the triangle's area.

The input to our method is a non-developable mesh surface segmented into a set of patches, and the output is a set of mesh patches with modification as small as possible: each modified patch has the best developability (measured by a new metric defined in DEGI) and still has continuous boundaries with neighboring patches. The resulting patches are called *quasi-developable* in this paper. In order to preserve the shape details or important design intents, our method also allows users to set a tolerance value to control the displacements of mesh vertices.

* Corresponding author.
E-mail address: liuyongjin@tsinghua.edu.cn (Y.-J. Liu).

0167-8396/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cagd.2012.03.009

We formulate the quasi-developable mesh approximation problem as a quadratic optimization problem. Some preliminary results of this work were orally presented at Zeng et al. (2010). In this paper, we extend the work in Zeng et al. (2010) using DEGI and make the following contribution: the optimization problem is proved to have a global minimization solution and this can be efficiently achieved by solving a large sparse linear system with the coefficient matrix independent of the vertices' new positions after deformation.

After the related work is reviewed in Section 2, an overview of our method is presented in Section 3. The mathematical formulation of our optimization problem is introduced in Section 4, based on a new measure for mesh developability using DEGI. In Section 5, the quasi-developable mesh approximation algorithms are detailed and experiments are presented in Section 6 that demonstrate the efficiency of our method. The conclusions of this paper are presented in Section 7.

2. Related work

2.1. Developable surface design

Developable surface design has been studied for decades. Considering surface developability at different design phases embodies that the design purpose is different. Accordingly, the techniques to obtain developability can be classified into two groups. The first group considers surface developability during the shape design process. That means surface developability is an important property that the designer is pursuing. The second group considers improving surface developability when a product is almost finished. In the case that the surface developability is less important than shape, techniques belonging to the second group are often requested to satisfy a certain shape change tolerance.

In the first group, the input is usually a set of boundary curves, and the purpose is to create a parametric or discrete developable surface to interpolate them. For the parametric case, the Bezier and B-spline surfaces are the most commonly used. Given a boundary curve and some end conditions of this boundary curve, researchers studied the conditions to construct a control net to make the corresponding Bezier or B-spline patch developable (Aumann, 2003, 2004; Chu and Sequin, 2002; Fernandez-Jambrina, 2007). These conditions were expressed in a set of equations, mostly non-linear constraints. There are also some techniques based on projective geometry (Pottmann and Wallner, 1999) or Gaussian image (Chen et al., 1999). For the discrete developable surface, Rose et al. (2007) constructed a discrete developable surface for arbitrary sketch-input boundary curves, using the relationship between the developable surface and the convex hulls of their boundaries. Frey (2002) introduced a boundary triangulation method to generate polygonal developable surfaces, where all triangles have their vertices on two 3D polylines. Based on boundary triangulation, Wang and Tang (2005) reviewed the problem as a deterministic search problem and adopted the Dijkstra algorithm to perform the optimization. They found the best developable boundary triangulation by interpolating two arbitrary polylines.

In the second group, usually the input is a final product, represented in a parametric or polygonal surface, and the purpose is to improve its developability while minimizing its shape change at the same time. This problem is often referred as *maximizing quasi-developability by deformation*. If the input surface is represented in NURBS, the deformation usually is applied to the control net (Wang et al., 2004). If the input surface is a polygonal surface, the deformation is applied to its vertices. The quasi-developability-by-deformation problem is often considered as an optimization problem. Wang and Tang (2004) minimized the total discrete Gaussian curvature for polygonal surfaces by relocating each vertex. In addition, the input polygonal mesh may be approximated by some other elements, such as PQ mesh (Liu et al., 2006), developable mesh (Liu et al., 2011b, 2007) or triangle strips (Chu et al., 2008; Liu et al., 2011, 2009). In this paper we present a novel least squares solution to the quasi-developability-by-deformation problem, which can achieve the best developability and simultaneously minimize a shape difference measure.

2.2. Discrete extended Gaussian image

The Gauss map G maps the normal vector of every point on a surface S to a point on the Gaussian sphere S^2. The Gaussian image of S is the union of all the normal vectors of S on S^2 (Struik, 1988). An extended Gaussian image (EGI) associated with S is a Gaussian image together with a weight set. For example, Horn (1984) defined the weight being the inverse of the Gaussian curvature. Since the EGI is determined uniquely for a convex object and is insensitive to object position, EGI has been used in shape reconstruction (Xu and Suk, 1995), surface partitioning (Tang and Liu, 2005), 3D pose determination (Kang and Ikeuchi, 1993) and shape match (Zouaki, 2003). A comprehensive study on EGI with various geometric operations can be found in Horn (1984), Zouaki (2003).

For triangular mesh surfaces, discrete extended Gaussian image (DEGI) can be used. Decaudin et al. (2006) presented a similar idea as ours, which tried to generate an approximated developable surface by finding a best-fitting developable surface for each triangle locally. This local best-fitting developable surface can be expressed with a constant axis vector and an angle. However, this fitting scheme can be optimized globally; i.e., a global curve fitting all points on S^2 can be obtained and can be used to guide the mesh deformation process.

The technique proposed in this paper treats the quasi-developable mesh approximation problem as an optimized deformation process. The major difference compared to previous methods is that our deformation is guided by the target unit normal (to be defined in Section 5.1) of each sample point in DEGI. This difference is the critical reason to make the coefficient matrix of the final large sparse linear system a constant matrix, i.e., independent of the vertices' new position and

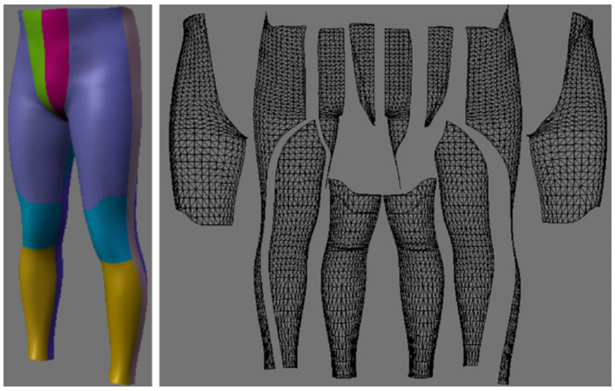
Fig. 1. Assembled mesh patches for a diving suit model.

needs to be computed only once. Thus, the original highly time-consuming optimization problem can be solved efficiently and stably by a least squares method.

3. Overview of the proposed method

The proposed method is based on the following property.

Property 1. If the Gaussian image of a regular surface S is a curve, then S is developable.

To see this property, note that the Gaussian curvature vanishes everywhere is a necessary and sufficient condition of a surface being developable (Struik, 1988). Let Ω be a small region containing a point $p \in S$, G the Gauss map, $A(\Omega)$ the region area on S and $A(G(\Omega))$ the area of Gaussian image of Ω on S^2. Then the Gaussian curvature $K(p)$ of point p can be expressed as

$$K(p) = \lim_{\Omega \to p} \frac{A(G(\Omega))}{A(\Omega)}$$

For any regular point p on S, the area $A(\Omega)$ is not zero, and the area $A(G(\Omega))$ is zero since the Gaussian image of Ω is a 1D curve segment. So the Gaussian curvature vanishes everywhere on S and S is developable.

The quasi-developable mesh approximation algorithm consists of two major stages:

Step 1: *Compute the target unit normals in the DEGI.* The discrete extended Gaussian image (DEGI) of a mesh patch is computed as a set of sample points on the Gaussian sphere S^2, which is mapped to the Gaussian sphere's parametric domain and fitted by a planar B-spline curve, called a *domain-skeleton curve*. This domain-skeleton curve is then mapped back to the Gaussian sphere to obtain the spherical skeleton curve. Finally, for each sample point in DEGI, its target unit normal is computed as the nearest point on the spherical skeleton curve.

Step 2: *Patch deformation guided by target unit normals.* The purpose of mesh deformation guided by target unit normals is to make each sample point in DEGI closer to its target point on the Gaussian sphere S^2. If all sample points in DEGI can coincide with their own target points, the deformed patch P_i' is a fully developable patch. During the deformation procedure, continuity among neighboring patches is guaranteed and the final model can maintain geometric fidelity with the original shape by moving mesh vertices within a user-specified tolerance. The mesh deformation process is treated as an optimization problem as presented below.

4. Problem formulation

The input model in our method is an assembled mesh surface M, which consists of a set of triangular mesh patches P_i (see the diving suit in Fig. 1 for an example). Each patch has C^0 continuity with neighboring patches and is represented by a triangular mesh, defined as a pair (T_i, V_i), where T_i and V_i are the set of triangles and vertices of patch P_i. As a method to improve the developability of a final product (i.e., fall into in the second group of surface design methods as summarized in Section 2.1), our method requires that the input mesh patches are not far from developable. Given a single mesh surface whose normals are diversely scattered over the Gaussian sphere, mesh segmentation into a set of near developable patches may be necessary as a preprocessing step (Julius et al., 2005; Liu et al., 2009; Yamauchi et al., 2005).

4.1. Measure of developability

Traditionally, the developability of a regular surface is measured by the integral of the absolute Gaussian curvature over the entire surface. The smaller the integral, the better the developability. For a discrete polygonal surface, the integral can

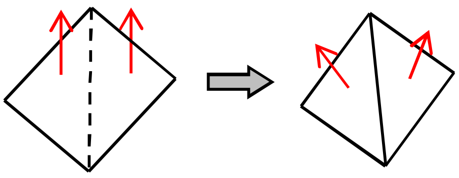

Fig. 2. The deficiency of length-based deformation measure: folding along the dashed line can change two triangles' orientation, but edges' length remains unchanged.

be converted into a summation over vertices' discrete Gaussian curvature. Usually a discrete Gaussian curvature at each vertex is computed based on the internal angles of the one-ring neighboring triangles, we call this discrete developability measure as *angle based developability* (ABD). Wang and Tang (2004) proposed a developability measure based on a discrete Gaussian curvature approximation and an impulse function for the polygonal surface. In Wang and Tang's measure, each mesh vertex only had two statuses: developable or non-developable. Thus, their scheme can only inform users about the number of completely developable vertices, but it is unable to evaluate the developability of each vertex.

In this paper, a new developability measure, which computes the difference between triangles' true orientation and target orientation, is proposed. This new measure is denoted as *orientation based developability* (OBD). The smaller the difference between its true orientation and target orientation is, the better developability (with respect to the whole mesh) the triangle has. Here, we process triangle faces instead of vertices because modifying vertices directly always results in non-linear problems, e.g., by associating a Voronoi area or a barycentric area to a vertex.

Denote a triangle $t = (v_1, v_2, v_3)$ after deformation by $t' = (v'_1, v'_2, v'_3)$: if its true normal is identical with its target normal n_t, then n_t is perpendicular to each edge of this triangle. That is, $d_i = |n_t \cdot (v'_{i+1 \bmod 3} - v'_i)| = 0$, $i = 1, 2, 3$. The more the true normal deviates from the target normal, the larger the summation of $d_1 + d_2 + d_3$ is.

Definition 2. *Orientation based developability* (OBD), denoted as $d(t)$, of a single triangle t is defined as the difference between the triangle's true orientation and the target orientation, computed by

$$d(t) = d_1 + d_2 + d_3 \tag{1}$$

where

$$\begin{aligned} d_1 &= \left|n_t \cdot \left(v'_2 - v'_1\right)\right| \\ d_2 &= \left|n_t \cdot \left(v'_3 - v'_2\right)\right| \\ d_3 &= \left|n_t \cdot \left(v'_1 - v'_3\right)\right| \end{aligned} \tag{2}$$

If $d(t) = 0$, this triangle's normal is identical with the target normal and is considered in a completely developable status. Otherwise, its developability can be evaluated by $d(t)$, which decreases as the developability increases. The measure of mesh developability should be independent of triangle density. For example, a meshed sphere should have similar developability whether it is triangulated with 1000 triangles or 10 000 triangles. It means that a mesh's tessellation scheme, such as simplification and subdivision, should not have serious effect on a mesh surface's developability. It is therefore essential to consider triangle face area in the computation of finding an optimal target normal. We handle this requirement by encoding triangle area in the DEGI (Section 5.1).

Definition 3. The developability of a patch P is defined as the summation of the developability of all its triangles,

$$D(P) = \sum_{i \in |T|} d(t_i) \tag{3}$$

where $|T|$ is the cardinality of the triangle set T of patch P.

4.2. Measure of shape difference

In Wang and Tang (2004), the shape difference of a mesh surface after deformation is expressed as the length change of each mesh edge. This is not an effective criterion. Imagine a paper as shown in Fig. 2, we can fold along the paper's diagonal line and keep all the edges unchanged. In such a case, the previous methods will consider there is no change in shape. This may lead to self-intersection where the triangle's orientation is changed dramatically. Given this observation, in our work a triangle's deformation is measured by the affine transformation of that triangle, which takes changes of both triangle orientation and edge magnitude into account.

Definition 4. The shape difference measure $s(t)$ of a single triangle t is defined by

$$s(t) = s_1(t) + s_2(t) + s_3(t) \tag{4}$$

where

$$s_1(t) = \left\| \left(v'_2 - v'_1\right) - (v_2 - v_1) \right\|_2^2$$
$$s_2(t) = \left\| \left(v'_3 - v'_1\right) - (v_3 - v_1) \right\|_2^2 \tag{5}$$
$$s_3(t) = \frac{\|(v'_3 + v'_2 + v'_1) - (v_3 + v_2 + v_1)\|_2^2}{3} \tag{6}$$

$\| \, \|_2$ is vector 2-norm, (v_1, v_2, v_3) and (v'_1, v'_2, v'_3) are vertices of the original and deformed triangles, respectively.

In the above definition, Eq. (5) describes the affine transformation of a triangle and Eq. (6) describes the translation of the triangle's mass center.

Definition 5. The shape difference $S(P)$ of a patch P is defined as the summation of the shape difference of all its triangles,

$$S(P) = \sum_{i \in |T|} s(t_i) \tag{7}$$

4.3. Optimization problem formulation

Given a triangular mesh patch P with $D(P) > 0$, our problem is to find a new triangular patch P^*, which has the same topology as the original patch P, but with better developability $D(P^*) < D(P)$, and minimizes the shape difference $S(P)$. The optimization problem is formulated by minimizing an objective function F as

$$\arg\min_T F(P) = D(P) + wS(P) \tag{8}$$

This objective function is a combination of mesh developability and shape difference: w is a non-negative penalty value, which allows the user to balance the developability and shape change since they are actually a trade-off between each other.

The objective function given in Eq. (8) applies to a single patch. For an assembled surface with multiple patches, we apply it to each patch separately. The continuity across boundaries of each patch is discussed in detail in Section 5.3.

5. Quasi-developable mesh approximation

In this section, the quasi-developable mesh approximation algorithm is detailed. First, the target unit normal vector for each triangle is computed from a skeleton curve in DEGI, optimized in a triangle-area-weighted sense. Then, these target unit normal vectors are used to guide the mesh deformation process. To cater the algorithm for a complex mesh model, the patch boundary continuity and tolerance of shape change by vertex movement are discussed. All the above considerations can be represented by a single large sparse linear system, which is solved efficiently and stably by a least squares direct matrix solver.

5.1. Computing target normal vectors

Using the front patch in Fig. 1 as an example (redrawn in Fig. 3(a)). The DEGI of that patch is shown in Fig. 3(b). Each sample point in DEGI is mapped to the parameter domain of the Gaussian sphere S^2 by

$$\theta = \begin{cases} \cos^{-1}\left(\frac{x}{\sqrt{x^2+y^2}}\right), & \text{if } y > 0 \\ 2\pi - \cos^{-1}\left(\frac{x}{\sqrt{x^2+y^2}}\right), & \text{otherwise} \end{cases}$$
$$\varphi = \cos^{-1}(\mathbf{R}z) \tag{9}$$

where $\theta \in [0, 2\pi)$, $\varphi \in [0, \pi]$, $\mathbf{R}$ is a rotation matrix used to improve the fitting quality. For this parameterization, the sample points in S^2 should be kept away from the north ($\varphi = 0$) and south points ($\varphi = \pi$), where θ is quite sensitive to the sample point's position. To obtain an optimal $\mathbf{R}$, we find a great circle on S^2 such that there is a minimal summarization of geodesic distances from all sample points to this great circle. Then $\mathbf{R}$ is the matrix that rotates the great circle aligned with the equator. The mapping results for the patch in Fig. 3(a) is shown in Fig. 3(d) (red points).

Given an unordered set of triangle-area-weighted planar points, Goshtasby (1999) proposed a method that converts a point distribution into a digital image and subdivides the plane into grids. Each grid point p holds a radial effect field f with itself as center,

$$f(x, y) = \sum_{i=1}^{N} \sqrt{(x - x_i)^2 + (y - y_i)^2 + r^2} \tag{10}$$

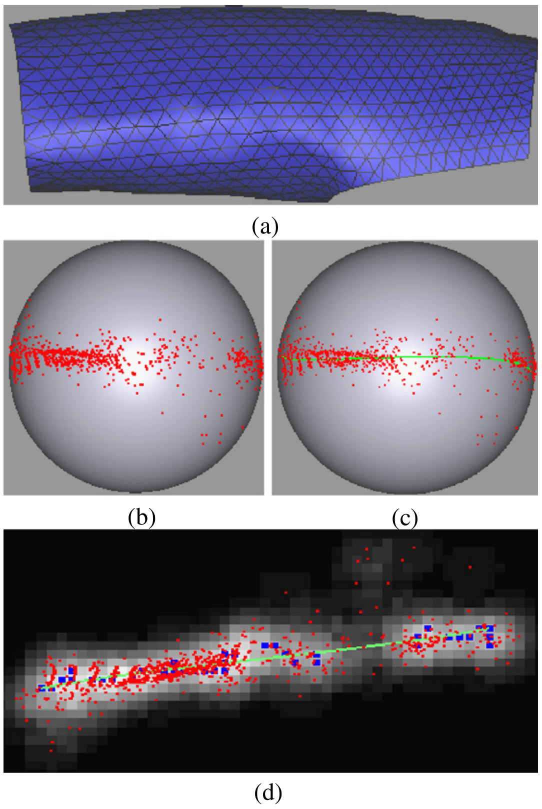

Fig. 3. Spherical curve fitting. (a) Original mesh patch; (b) DEGI of the patch; (c) Spherical curve fitting (shown in green); (d) Digital image of DEGI. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

where N is the number of grid points, (x_i, y_i) is the coordinate of the ith grid point, r is a weight value associated to each grid. One digital image generated from planar point cloud is shown in Fig. 3(d). Based on the local maximum and image gradient direction, it is possible to find the skeleton points (the blue ones) in Fig. 3(d). We use the following modified Goshtasby's method:

- An effect region control is added in Eq. (10),

$$f(x, y) = \sum_{i=1}^{N} sign\big(\|p - p_i\|_2\big)\sqrt{(x - x_i)^2 + (y - y_i)^2 + r^2} \tag{11}$$

  where p is the current reference grid point, p_i is the ith grid point, $sign(x)$ is a control function: if x is greater than a given threshold, then $sign(x) = 1$, otherwise $sign(x) = 0$.
- The value r is no longer a constant. It is proportional to the number of sample points SP mapped to this grid and proportional to the summation of the weights w associated to SP, which is the triangle area according to the DEGI definition. That is,

$$r(p) = k \cdot |SP| \cdot \sum_{p_i \in SP} w(p_i) \tag{12}$$

  where k is a scale factor and $|SP|$ is the cardinality of set SP.
- Fit a B-spline curve c_{2D} to the skeleton grid points, using the least squares method in Piegl and Tiller (1997). One example of plane fitting curve is shown in Fig. 3(d) (green curve). Then the plane fitting curve is mapped back to the Gaussian sphere S^2 as a spherical fitting curve c_{3D}. One example of c_{3D} is shown in Fig. 3(c) (green curve).

For every sample point n in DEGI of a patch P, its nearest point on c_{3D} is found and served as the target normal vector n_t for the triangles whose normal is n.

To find an optimal spherical arc on S^2 that best approximates all sample points, we can also use the spherical spline interpolation/approximation method (e.g., in Buss and Fillmore (2001)) directly on S^2. We have implemented both a spherical spline approximation method (Hoschek and Seemann, 1992) and the above digital-image method. In our implementation, the spherical arc approximation directly on the spherical domain utilizes two spherical angles (i.e., many computations using the values of trigonometric functions like sine and cosine), which makes our implementation less stable. Meanwhile, implementing a weighted point approximation algorithm directly on the spherical domain is also not a trivial task. As a comparison, we use robust image processing codes in MATLAB for the DEGI computation. So in the proposed method, the spherical arc approximation is performed on the DEGI in 2D plane.

5.2. Mesh deformation

The target normal vector n_t is used to guide the mesh deformation complying with the objective function (8). Using Eqs. (2), (5) and (6), the objective function for a single triangle t_i can be written in a matrix form as

$$F_{t_i}(v'_{i1}, v'_{i2}, v'_{i3}) = \|A_i x_i - c_i\|_2^2 \tag{13}$$

where

$$A_i = \frac{1}{3}\begin{bmatrix} -3H_{1\times3} & 3H_{1\times3} & (0)_{1\times3} \\ (0)_{1\times3} & -3H_{1\times3} & 3H_{1\times3} \\ 3H_{1\times3} & 0_{1\times3} & -3H_{1\times3} \\ -3I_{3\times3} & 3I_{3\times3} & (0)_{3\times3} \\ -3I_{3\times3} & (0)_{3\times3} & 3I_{3\times3} \\ 3I_{3\times3} & I_{3\times3} & I_{3\times3} \end{bmatrix}_{12\times9}$$

$$x_i = \begin{bmatrix} (v'_{i1})_{3\times1} \\ (v'_{i2})_{3\times1} \\ (v'_{i3})_{3\times1} \end{bmatrix}_{9\times1}, \qquad c_i = \frac{1}{3}\begin{bmatrix} 0 \\ 0 \\ 0 \\ 3(v_2 - v_1)_{3\times1} \\ 3(v_3 - v_1)_{3\times1} \\ (v_1 + v_2 + v_3)_{3\times1} \end{bmatrix}$$

and $H_{1\times3} = wn_{it}$ is a row vector, since the ith triangle's target normal vector n_{it} has x, y, z three coordinates. $I_{3\times3}$, $(0)_{3\times3}$ and $(0)_{3\times1}$ are 3-by-3 identity matrix, 3-by-3 and 3-by-1 zero matrices respectively.

It is worth noting that A_i, called the *triangle's coefficient matrix* in this paper, only depends on the triangle's target normal and penalty value, and vector c_i only depends on triangle vertex position before deformation.

Now the optimization problem summarized in Eq. (8) can be written as

$$\arg\min_T F(P) = \sum_{i\in|T|} \|A_i x_i - c_i\|_2^2 = \|Ax - c\|_2^2 \tag{14}$$

where T is the triangle set of patch P, $|T|$ is the cardinality and

$$A = \begin{bmatrix} A_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & A_{|T|} \end{bmatrix}_{12|T|\times9|T|}$$

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_{|T|} \end{bmatrix}_{9|T|\times1}, \qquad c = \begin{bmatrix} c_1 \\ \vdots \\ c_{|T|} \end{bmatrix}_{12|T|\times1}$$

The coefficient matrix A is a large sparse matrix, which only depends on the target normal vectors and a constant penalty value. The decomposition of the coefficient matrix needs to be done only once; this property speeds up the algorithm dramatically.

In practical surface design, for an intuitive control by users to preserve shape details, a deformation tolerance variable δ_{err} can be used to restrict the movement of each vertex. After each step during a mesh deformation process, the update of vertex's position $\|\Delta v\|_2$ is checked. If $\|\Delta v\|_2 > \delta_{err}$, then $\|\Delta v\|_2 = \delta_{err}$. The effect of deformation tolerance δ_{err} is discussed in detail in Section 6.

5.3. Continuity preservation

In our method, the measure of mesh developability and shape difference is triangle-based. If two adjacent triangles move in different directions, a gap may be generated in-between. To address this issue, we convert a triangle-based deformation to a vertex-based deformation. If we interpret matrix-vector multiplication Ax in Eq. (14) as a linear combination of column vectors in matrix A and note that a vertex appears several times (i.e., the number of vertex degree in the mesh patch) in the column vector x, the matrix A can be compacted by adding together the column vectors in A related to a common vertex. Given this observation, in Eq. (14) the dimension of the coefficient matrix A, unknown vector x and known vector c can be reduced to $12|T| \times 3|V|$, $3|V| \times 1$ and $12|T| \times 1$, respectively.

For the boundary vertices of a mesh patch in an assembled mesh surface, the C^0 continuity across the patch boundary must be considered. One trivial solution is to set $\delta_{err} = 0$ for boundary vertices. However, in all our experiments, setting an overall small tolerance δ_{err} offers a very hard constraint on mesh deformation and the resulting deformed patches are often very wrinkled. In our method, we propose to use a soft constraint that always leads to a smooth mesh. To do so we adopt the idea of moving least squares (MLS) (Yoshioka et al., 2006): each vertex is associated with a weight value computed from a discrete weight function by

$$w(v) = \frac{1}{\min\{\|v - v_i^c\|_2, v_i^c \in V_{cstr}\} + \varepsilon^2} \tag{15}$$

where V_{cstr} is a set of constrained vertices, including all the boundary vertices. The user can also specify some internal vertices to be constrained. The constant ε is set to be 0.001 to simulate a sufficiently large weight for constrained vertices. Using weight function (15), Eq. (14) can be written as

$$\arg\min_V F(P) = \|AWx - c\|_2^2 \tag{16}$$

where W is a diagonal weight matrix whose diagonal elements are each vertex's weight value.

5.4. Least squares solver

The dimension of coefficient matrix A in Eq. (16) is $12|T| \times 3|V|$ and the large linear system is overdetermined, where $|T|$, $|V|$ are the cardinality of the patch's triangle and vertex set respectively. We solve this linear system in a least squares sense:

$$W^T A^T AWx = W^T A^T c \tag{17}$$

Note that the coefficient matrix $W^T A^T AWx$ is sparse, positive definite, and symmetric. A recent comparative study in Botsch et al. (2005), Davis (2006) has shown that the direct methods are suitable for this type of linear systems. The Cholesky factorization is an efficient direct method that can explore the matrix sparsity well (Davis, 2006). Thus, it is used in our method to decompose the matrix, i.e., $W^T A^T AWx = R^T R$ where R is an upper triangular sparse matrix. After the factorization, x can be solved by back substitution.

In a quasi-developable mesh optimization process, we solve Eq. (17) for several times. Each time the vector c is updated serving as an input in the next iteration. The iteration process is fast, since the coefficient matrix A only depends on the triangle's target normal and the decomposition $W^T A^T AW$ needs to be done only once. The overall algorithm pseudo-code is as follows.

```
**Algorithm 1.** quasi_dev_approx($P$, $V_{cstr}$)
*Input.* A mesh patch $P$ and a set of constrained vertices $V_{cstr}$.
*Output.* A quasi-developable patch $P'$.
**Step 1. Compute the target unit normal vectors**
1.1. Compute the DEGI of $P$.
1.2. Find a skeleton curve in the Gaussian sphere's parameter domain using modified Goshtasby's technique
     (Goshtasby, 1999).
1.3. Map the skeleton curve back to Gaussian sphere and compute the target unit normals.
**Step 2. Quasi-developable approximation**
2.1. Compute matrix $A$ using Eqs. (13), (14).
2.2. Compute matrix $W$ using Eq. (15).
2.3. Cholesky factorization of $W^T A^T AW$.
2.4. While (End condition == false).
2.4.1.  Compute vector $c$ using Eqs. (13), (14).
2.4.2.  Solve the new position of vertices by back substitution in Eq. (17).
2.4.3.  Update position of each vertex with check if the deformation tolerance $\delta_{err}$ is satisfied.
```

The algorithm will stop when one of the following three criteria is satisfied:

1. The algorithm reaches its maximum iterative number *maxIter*.
2. The total Gaussian curvature decrease-ratio is less than a threshold value δ_{ABD}.
3. The total OBD (Definition 1) decrease-ratio is less than a threshold value δ_{OBD}.

For the second criterion, the Gaussian curvature of a vertex is computed by

$$K(v) = \frac{1}{A_v}\left(2\pi - \sum_j \theta_j\right) \tag{18}$$

where A_v is the Voronoi area surrounding v and θ_j is the incident internal angles at v. The total Gaussian curvature of a patch P is given by $G^{ABD} = \sum_{i\in|V|} |K(v)|$. The Gaussian curvature decrease-ratio, between two consecutive iterations j and $j+1$, is computed by

$$ratio_{ABD} = \frac{G_j^{ABD} - G_{j+1}^{ABD}}{G_j^{ABD}} \times 100\% \tag{19}$$

As a comparison, the total OBD (Definition 1) decrease-ratio is computed by

$$ratio_{OBD} = \frac{D_j(P) - D_{j+1}(P)}{D_j(P)} \times 100\% \tag{20}$$

where the developability $D(P)$ of patch P is defined in Definition 2.

6. Experiment results

The proposed developable mesh approximation algorithm has been implemented and applied to the following cases. The first case shows the validity of the developability measure OBD (Definitions 2 and 3). Then the effect of using different deformation tolerance values δ_{err} is explored. Finally, this algorithm is applied to two assembled polygonal models, taken from the garment and the furniture industry, respectively.

For a better understanding and visualizing mesh developability and shape change, we use the following two tools:

- *Gaussian curvature map.* It assigns a color value to each mesh vertex, where the color value represents the magnitude of Gaussian curvature at this vertex.
- *Shape difference map.* It assigns a color value to each mesh vertex, where the color value represents the displacement of this vertex.

In the following discussion, AE denotes the average edge length in a given mesh patch. In the experiment, we scale the mesh model so that the AE is of the order of magnitude 10^{-1}. We empirically determine the weight w in Eq. (8) using the following rule of thumb. We use w to balance the scales of developability measure D (Eqs. (1)–(3)) and shape difference measure S (Eqs. (4)–(7)). Since the target normal is usually not far away from the current normal, the measure d_i in Eq. (2) is of the order of magnitude 10^{-1}, as the AE. For the shape difference s_i in Eqs. (5)–(6), the edge length change is also usually of the order of magnitude as the AE and then the square of L_2 norm has the order of magnitude 10^{-2}. So to balance D and S in the same order of magnitude, we use $w = 10$ in our experiments.

Validity of OBD measure. Fig. 4 shows a quasi-developable mesh optimization of a curved cone-like lampshade. In this example, a large deformation tolerance $\delta_{err} = 10AE$ is used, so that mesh vertices can move freely to achieve the best developability. The algorithm terminates after three iterations. The developability optimization effect during the iteration process is illustrated using both Gaussian curvature maps (Fig. 4(b)) and shape difference maps (Fig. 4(c)). Table 1 presents the statistical data of Gaussian curvature decrease-ratio $ratio_{ABD}$ and OBD decrease-ratio $ratio_{OBD}$. From Table 1, it is readily seen that the mesh developability improves dramatically at the first iteration. Also observed from Table 1, $ratio_{ABD}$ and $ratio_{OBD}$ share the same decreasing trend, validating the performance of OBD measure and its associated optimization.

Effect of deformation tolerance δ_{err}. A large δ_{err} makes vertices move freely to achieve the best developability, and a small δ_{err} can preserve more shape details. To investigate the effect of different deformation tolerance δ_{err}, we redraw Fig. 3(a) in Fig. 5. In this case, three different tolerance values, i.e., $\delta_{err} = AE$, $0.1AE$, $0.02AE$, respectively, are tested. The resulting shape difference maps and Gaussian curvature maps are illustrated in Fig. 5. The statistical data of performance in this case is summarized in Table 2. From the results, it is indicated that the larger δ_{err} is, the more developability we can obtain, and the larger the shape difference is. The trade-off between patch developability and shape difference can be balanced by the value δ_{err}.

Practical industrial models. Two assembled mesh surfaces in industrial design are presented. One is a pair of pants from the garment industry and the other is a seat from the furniture industry. In the upstream model design process, the two mesh models have been segmented into a set of patches joined with C^0 continuity.

Fig. 6 shows an assembled pant mesh surface with seven patches. The performance of our method is illustrated in Fig. 7 (Gaussian curvature maps) and Fig. 8 (shape difference maps). In this assembled pant case, the tolerance value for patches 1, 2, 5 are set to be $0.01AE$, since patch 1 contains some fine detailed wrinkles and patches 2 and 5 contain the crotch point, which is an important reference feature in designing a pair of pants. The tolerance values for other patches are set to be $0.05AE$ to achieve better developability. All the patch boundary vertices are fixed during the optimization process. The second model in industrial design is an assembled seat mesh surface, which is segmented into 34 patches joined with C^0

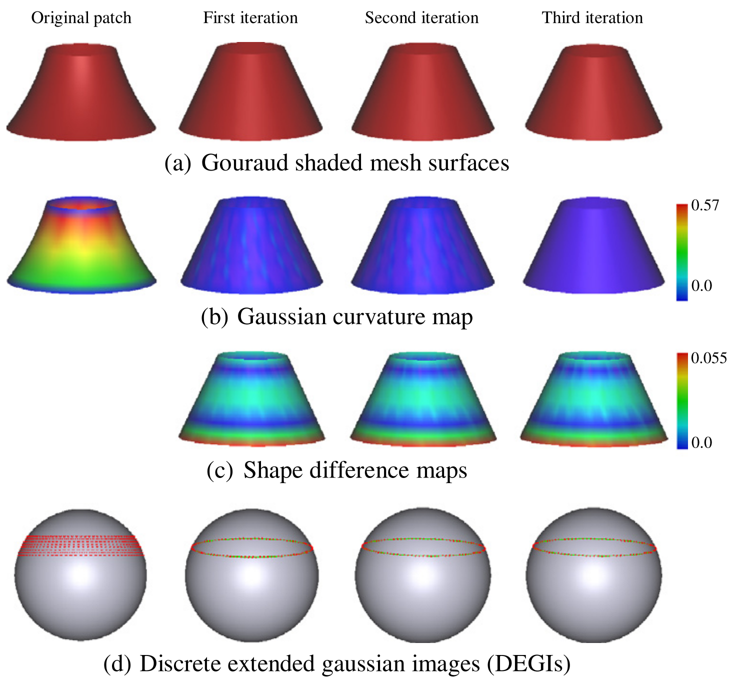
Original patch
First iteration
Second iteration
Third iteration
(a) Gouraud shaded mesh surfaces
0.57
0.0
(b) Gaussian curvature map
0.055
0.0
(c) Shape difference maps
(d) Discrete extended gaussian images (DEGIs)

Fig. 4. Quasi-developable mesh optimization of a cone-like lampshade.

Table 1
Statistical data of a quasi-developable mesh optimization shown in Fig. 4.

Decrease ratio	1st iter.	2nd iter.	3rd iter.
$ratio_{ABD}$	90.9%	39.7%	6.3%
$ratio_{OBD}$	89.2%	54.3%	8.2%

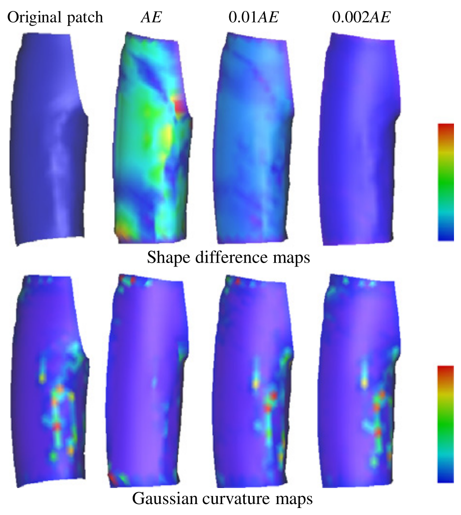

Fig. 5. Quasi-developable mesh optimization of a pant patch in Fig. 1 with different tolerances δ_{err}.

continuity (Fig. 9). Each patch has a fixed boundary and the internal vertices are allowed to move within the deformation tolerance to achieve the best developability. In this seat case, the performance of our algorithm is illustrated in Fig. 10 (Gaussian curvature maps) and Fig. 11 (shape difference maps).

Comparison with previous work. With a similar objective as our method, Wang and Tang (2004) proposed one global and one local quasi-developable mesh optimization method. Their local optimization method considered each mesh vertex and its one-ring neighboring vertices individually. Each vertex was moved along its normal direction to minimize its developability locally. This local vertex movement scheme was also adopted in Decaudin et al. (2006) for triangle transformation.

Table 2
Statistical data of a pant patch optimization in Fig. 5 using different tolerance values (data includes maximum vertex's position difference and maximum Gaussian curvature).

δ_{err}	$ratio_{ABD}$	$ratio_{OBD}$	Maximum vertex's position difference	Maximum Gauss curvature
AE	48.78%	17.95%	0.231	8.94
$0.1AE$	29.50%	13.70%	0.0421	4.10
$0.02AE$	9.06%	5.77%	0.0084	3.86

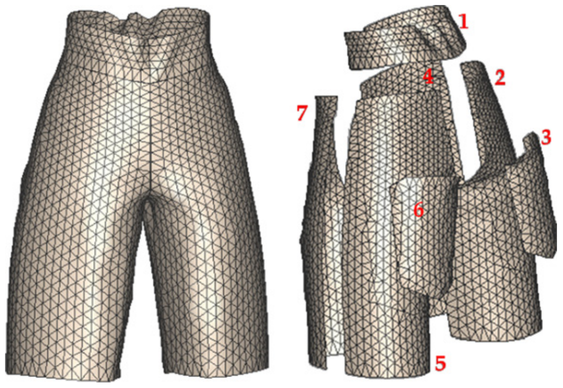

Fig. 6. An assembled pant model and its seven mesh patches. Two pocket patches (3 and 6) exist in front of thigh position.

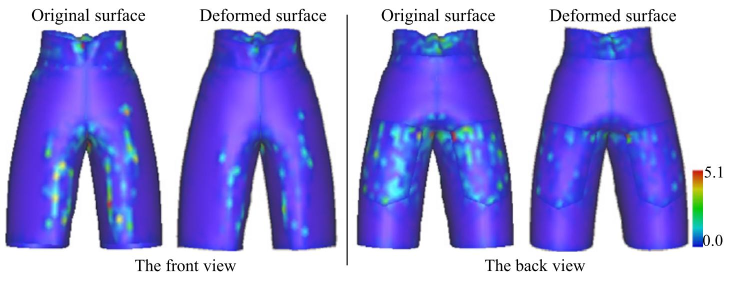

Fig. 7. Gaussian curvature maps of the original surface (Fig. 6) and the deformed pant surface.

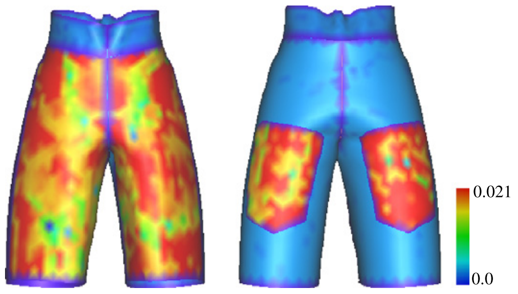

Fig. 8. Shape difference maps of the deformed pant surface shown in Fig. 7, in front and back views.

Though the local optimization method was fast, it may lead to a non-smooth mesh, i.e., wrinkles occur, as mentioned in Wang and Tang (2004), Wang (2008). The global method can provide good results both in improving mesh developability and in maintaining its smoothness, but it has a high computational complexity and is time consuming in Wang and Tang (2004). As a comparison, our method generates visually appealing surfaces using a MLS-style soft constraint on vertices (Yoshioka et al., 2006) and runs fast: in our method, the coefficient matrix is fixed during the iteration process and then needs to be decomposed only once. Since one state-of-the-art work of a novel flattenable Laplacian mesh (Wang, 2008) also has the merits of fast convergence speed and global shape control (by considering interference between the surface and nearby objects) as ours, we further compare the running time of the flattenable Laplacian mesh and our least squares quasi-developable mesh. Table 3 shows the results. Both algorithms are tested at a PC with Intel(R) Core(TM) i5 CPU 750 running at 2.66 GHz using Windows 7 64-bit system. Our method also uses Matlab R2010a for Cholesky factorization and

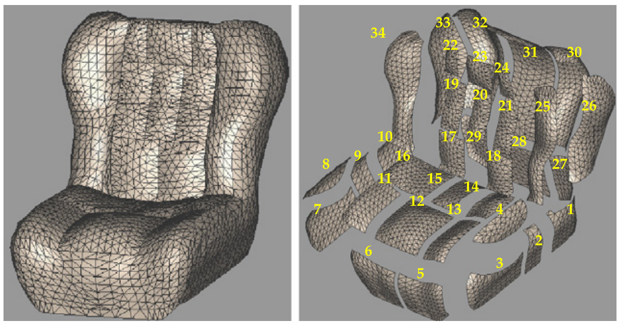

Fig. 9. An assembled seat model and its 34 mesh patches joined with C^0 continuity.

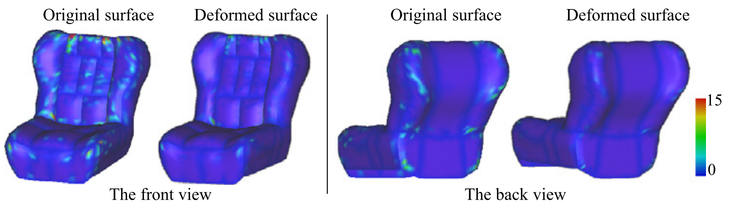

Fig. 10. Gaussian curvature maps of the original surface (Fig. 9) and the deformed seat surface.

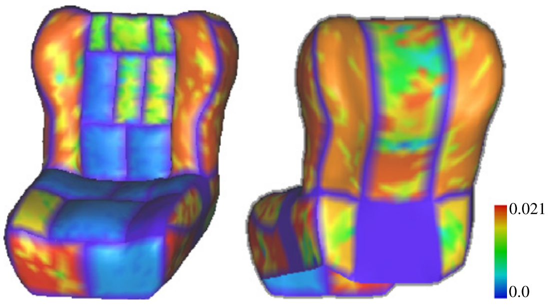

Fig. 11. Shape difference maps of the deformed seat surface shown in Fig. 10, in front and back views.

Table 3
The comparison of the running time (in millisecond) on the seven patches in the assembled pant model (Fig. 6), using the flattenable Laplacian (FL) mesh (Wang, 2008) and our least squares quasi-developable mesh.

Patch id	#Bndry. ver./#Total ver.	# Triangles	FL time	Ours time
1	108/292	474	325	771
2	96/677	1256	2380	1946
3	46/173	298	265	365
4	88/513	936	1323	980
5	96/676	1254	2373	1905
6	46/172	296	253	446
7	88/513	936	1440	981
Total	568/3016	5450	8359	7394

matrix back substitution. From the results shown in Table 3, our method is averagely faster than the flattenable Laplacian mesh (Wang, 2008).

7. Conclusions

This paper studies how to deform a given free-form mesh surface into a quasi-developable surface with small shape change. Our method is based on the property that if the Gaussian image of a surface is a spherical curve, this surface is developable. We first find a skeleton curve globally in DEGI and compute a target unit normal for each triangle in the mesh. Then we deform each triangle to make its new face normal as close to its target normal as possible. This target-normal-guided deformation is described as a global optimization problem using a new mesh developability measure based on DEGI. In particular, the resulting large sparse coefficient matrix is only dependent on the skeleton curve, independent of the updated positions of mesh vertices. This property makes it possible to decompose the coefficient matrix only once during the optimization process, and thus, our method is faster than previous global optimization methods.

Mesh models used in industrial design are usually complex and assembled by multiple patches. Our method allows the user to specify different deformation tolerance δ_{err} for different vertices, so that important features in the model, as well as continuity across the patch boundary, can be easily maintained. This reveals another advantage of the proposed method. The future work includes applying the proposed method in some industrial applications such as garment design (Liu et al., 2010; Ma et al., 2011) and 2D pattern generation of 3D toy models (Liu et al., 2011a).

Acknowledgements

The authors thank the reviewers for their detailed comments that helped improve this paper. The authors also thank Professor C.C.L Wang at CUHK for providing the FL mesh coding and several 3D mesh patch models. This work is supported by the National Basic Research Program of China (2011CB302202), the Natural Science Foundation of China (61100088, 60970099) and the 863 program of China (2012AA011801). The second author was partially supported by Program for NCET and TNList Cross-discipline Foundation, and the third author was partially supported by Guangdong Natural Science Foundation (S2011040000145).

References

Aumann, G., 2003. A simple algorithm for designing developable bezier surfaces. Computer Aided Geometric Design 20 (8), 601–619.
Aumann, G., 2004. Degree elevation and developable bezier surfaces. Computer Aided Geometric Design 21 (7), 661–670.
Botsch, M., Bommes, D., Kobbelt, L., 2005. Efficient linear system solvers for mesh processing. In: IMA Mathematics of Surfaces XI. In: Lecture Notes in Computer Science, vol. 3604, pp. 62–83.
Buss, S.R., Fillmore, J.P., 2001. Spherical averages and applications to spherical splines and interpolation. ACM Transactions on Graphics 20 (2), 95–126.
Chen, H., Lee, I., Leopoldseder, S., Pottmann, H., Randrup, T., Wallner, J., 1999. On surface approximation using developable surfaces. Graphical Models & Image Processing 61 (2), 110–124.
Chu, C., Sequin, C., 2002. Developable bezier patches: properties and design. Computer-Aided Design 34 (7), 511–527.
Chu, C., Wang, C., Tsai, C., 2008. Computer aided geometric design of strip using developable bezier patches. Computers in Industry 59, 601–611.
Davis, T., 2006. Direct Methods for Sparse Linear Systems. SIAM, Philadelphia.
Decaudin, P., Julius, D., Wither, J., Boissieux, L., Sheffer, A., 2006. Virtual garments: a fully geometric approach for clothing design In: Eurographics '06, pp. 625–634.
Fernandez-Jambrina, L., 2007. B-spline control nets for developable surfaces. Computer Aided Geometric Design 24, 189–199.
Frey, W., 2002. Boundary triangulations approximating developable surfaces that interpolate a closed space curve. Int'l J. Foundations of Computer Science 13, 285–302.
Goshtasby, A., 1999. Fitting parametric curves to dense and noisy points In: Int. Conf. Curves and Surfaces, pp. 227–237.
Horn, B., 1984. Extended Gaussian images. Proceedings of the IEEE 72 (12), 1671–1686.
Hoschek, J., Seemann, G., 1992. Spherical splines. RAIRO Model. Math. Anal. Numer. 26 (1), 1–22.
Julius, D., Kraevoy, V., Sheffer, A., 2005. D-charts: quasi-developable mesh segmentation. Computer Graphics Forum 24 (3), 581–590.
Kang, S., Ikeuchi, K., 1993. The complex egi: a new representation for 3d pose determination. IEEE Transaction on Pattern Analysis and Machine Intelligence 15 (7), 707–721.
Liu, Y., Pottmann, H., Wallner, J., Yang, Y., Wang, W., 2006. Geometric modeling with conical meshes and developable surfaces In: Proc. ACM SIGGRAPH '06, pp. 681–689.
Liu, Y.-J., Tang, K., Joneja, A., 2007. Modeling dynamic developable meshes by the Hamilton principle. Computer-Aided Design 39 (9), 719–731.
Liu, Y.-J., Lai, Y.-K., Hu, S.-M., 2009. Stripification of free-form surfaces with global error bounds for developable approximation. IEEE Transaction on Automation Science and Engineering 6 (4), 700–709.
Liu, Y.-J., Zhang, D.-L., Yuen, M.-F., 2010. A survey on cad methods in 3d garment design. Computers in Industry 61 (6), 576–593.
Liu, Y.-J., Chen, Z.-Q., Tang, K., 2011. Construction of iso-contours, bisectors and Voronoi diagrams on triangulated surfaces. IEEE Transaction on Pattern Analysis and Machine Intelligence 33 (8), 1502–1517.
Liu, Y.-J., Ma, C.-X., Zhang, D.-L., 2011a. Easytoy: a plush toy design system using editable sketch curves. IEEE Computer Graphics & Applications 31 (2), 49–57.
Liu, Y.-J., Tang, K., Gong, W.-Y., Wu, T.-R., 2011b. Industrial design using interpolatory discrete developable surfaces. Computer-Aided Design 43 (9), 1089–1098.
Ma, C.-X., Liu, Y.-J., Yang, H.-Y., Teng, D.-X., Wang, H.-A., Dai, G.-Z., 2011. Knitsketch: A sketch pad for conceptual design of 2d garment patterns. IEEE Transaction on Automation Science and Engineering 8 (2), 431–437.
Piegl, L., Tiller, W., 1997. The NURBS Book, 2nd edition. Springer, Berlin.
Pottmann, H., Wallner, J., 1999. Approximation algorithms for developable surfaces. Computer Aided Geometric Design 16 (6), 539–556.
Rose, K., Sheffer, A., Wither, J., Cani, M., Thibert, B., 2007. Developable surfaces from arbitrary sketched boundaries In: Proc. Fifth Eurographics Symp., Geometry Processing, pp. 163–172.
Struik, D., 1988. Lectures on Classical Differential Geometry. Dover, New York.
Tang, K., Liu, Y.-J., 2005. An optimization algorithm for free-form surface partitioning based on weighted Gaussian image. Graphical Models 67 (1), 17–42.
Wang, C.C., 2008. Towards flattenable mesh surfaces. Computer-Aided Design 40 (1), 109–122.
Wang, C., Tang, K., 2004. Achieving developability of a polygonal surface by minimum deformation: a study of global and local optimization approaches. The Visual Computer 20 (8), 521–539.
Wang, C., Tang, K., 2005. Optimal boundary triangulations of an interpolating ruled surface. ASME J. Computing and Information Science in Eng. 5 (4), 291–301.
Wang, C., Wang, Y., Yuen, Y., 2004. On increasing the developability of a trimmed NURBS surface. Engineering with Computers 20 (1), 54–64.
Xu, Z., Suk, M., 1995. Representation and reconstruction of polygons and polyhedra using hierarchical extended Gaussian images. Annals of Mathematics and Artificial Intelligence 13 (3), 377–399.
Yamauchi, H., Gumhold, S., Zayer, R., Seidel, H., 2005. Mesh segmentation driven by Gaussian curvature. The Visual Computer 21 (8–10), 659–668.

Yoshioka, Y., Masuda, H., Furukawa, Y., 2006. A constrained least-squares approach to interactive mesh deformation In: Proc. of the 2006 Int'l Conf. Shape Modeling and Appl., pp. 153–162.

Zeng, L., Chen, M., Yuen, M., 2010. Developable mesh surface approximation by normal guided deformation. Presented at Computer Graphics International Conference.

Zouaki, H., 2003. Representation and geometric computation using the extended Gaussian image. Pattern Recognition Letters 24, 1489–1501.