# Preserving Form-Features in Interactive Mesh Deformation

Hiroshi Masuda
The University of Tokyo

Yasuhiro Yoshioka
The University of Tokyo

Yoshiyuki Furukawa
National Institute of Advanced Industrial Science and Technology

## Abstract

Interactive mesh editing techniques that preserve discrete differential properties are promising to support the design of mechanical parts such as automobile sheet metal panels. However, existing methods lack the ability to manipulate form-features and hard constraints, which are common in engineering applications. In product design, some regions on a 3D model are often required to precisely preserve the surface types and parameters during deformation. In this paper, we propose a discrete framework for preserving the shapes of form-features using hard constraints in interactive shape deformation. Deformed shapes are calculated so that form-features translate and rotate while preserving their original shapes according to manipulating handles. In addition, we show how to constrain the motion of form features using linear constraints. The implemented system can achieve a real-time response for constrained deformation.

**Keywords:** engineering design, geometric feature, mesh processing, deformation, partial differential equation

## 1 Introduction

In product design, 3D models are often created in the early stage of product development, because such models are very effective for preliminary design evaluation by the development team. The evaluation of design concepts in the early stage helps products to meet requirements for manufacturing, cost, safety, quality, maintenance, and so on.

Interactive free-form surface modeling techniques are very important in the early stage of design. Since design concepts are very often changed or discarded in the early stage, it is not reasonable to spend a lot of time on creating detailed 3D models. Although non-uniform rational B-spline (NURBS) surfaces have been widely used to represent free-form shapes in CAD applications, it is very tedious and time-consuming to manipulate many surface patches with a large number of control points.

Free-form deformation (FFD) is a popular interactive technique in computer graphics applications. FFD changes geometric shapes by deforming the space in which the object lies. However, FFD is not necessarily convenient for supporting product design, because it often modifies product shapes in unintended ways, for example, circular holes are deformed to ellipses.

In the last few years, several discrete deformation techniques based on differential properties have been published [Sorkine et al. 2004; Botsch and Kobbelt 2004a; Yu et al. 2004]. They represent the differential properties of a given surface as a linear system and deform the surface so that the differential properties are preserved. In the typical mesh editing technique, the user first selects the fixed region, which remains unchanged, and the handle region, which is used as the manipulation handle, and interactively deforms the shape by dragging the handle in the screen.

Our motivation for studying interactive deformation stems from requirements for the deformation of automobile sheet metal panels. In sheet metal panels, some curves and surfaces are often required to keep their original types, such as circles and cylinders, for manufacturing and assembly reasons. Such partial regions are called *form features* in product design. Sheet metal parts typically consist of the combination of free-form surfaces and form-features, each of which have different characteristics; while free-form surfaces are characterized by the distribution of curvature, form-features are defined by surface types and their parameters[Fontana et al. 1999; Cavendish 1995].

Therefore, it is required for engineering design to maintain the curvature of surfaces and the shapes of form-features.

In this paper, when constraints are approximately satisfied in the least squares sense, we call them *soft constraints*, and when constraints are precisely satisfied, we call them *hard constraints*. In typical shape design, while differential properties are not explicitly specified by the user, the surface types and their parameters are directly specified according to engineering requirements. It is obviously reasonable to treat differential properties as soft constraints, and user-defined constraints as hard constraints.

However, existing methods do not allow the combination of hard and soft constraints. They solve all constraints using the least-squares method and produce compromised solutions, which are not accepted in engineering sense. It is possible to put large weights on certain constraints, but it is difficult to predict weight values that satisfy the allowable margin of error, especially when differential properties are represented using the cotangent weighting method[Meyer et al. 2003], which approximates the mean curvature very well.

In this paper, we propose a novel mesh-editing framework that can manage form-features using hard constraints. Our main contribution in this paper is as follows:

- a mesh deformation framework in which hard and soft constraints are incoeporated;
- a new rotation method for normals based on the logarithms of quaternions;
- the introduction of new constraints for translating and rotating form-features while preserving their original shapes; and
- the introduction of new constraints for maintaining the motion of form-features on a straight line or a plane.

In the following section, we review the related work on mesh editing. In Section 3, we describe our mesh deformation framework, in which hard constraints are incorporated using Lagrange multipliers and a new rotation method is introduced based on quaternion logarithms. In Section 4, we introduce a method for preserving form-features by constraining the relative positions and rotations of

vertices, and then propose new feature constraints which maintain the motion of a form-feature on a line or a plane. We also show a simple feature extraction method. In Section 5, we evaluate our framework and show experimental results. We conclude the paper in Section 6.

## 2 Related Work

Interactive mesh-editing techniques have been intensively studied. Such research aims to develop modeling tools that intuitively modify free-form surfaces while preserving the details of shapes. There are several types of approach for mesh editing, which are based on space deformation (FFD), multiresolution, and partial differential equations (PDE).

FFD is very popular in computer graphics. Such methods modify shapes by deforming 3D space in which objects lie [Sederberg and Parry 1986; Coquillart 1990; MacCracken and Joy 1996; Hu et al. 2001b]. Cavendish [Cavendish 1995] discussed FFD approaches in the context of design support and described that FFD could be used for designing automotive sheet metal panels. We also aim at supporting the design of automotive sheet metal panels, but in our empirical investigation of automobile industry, problems arise when form-features in a 3D model are modified in unintended ways. It is difficult for FFD approaches to manage constraints on form-features, because the manipulation handles do not work directly on geometric shapes.

Multiresolution approaches [Eck et al. 1995; Zorin et al. 1997; Kobbelt et al. 1998; Guskov et al. 1999; Lee 1999] decompose a surface into a base mesh and several levels of details, each of which is represented as the difference between successive resolution levels. A shape is globally deformed at low resolution and locally deformed at high resolution. Botsch and Kobbelt [Botsch and Kobbelt 2004a] applied this technique to interactive mesh editing. A mesh model is decomposed into two-level resolutions and the smooth base is interactively deformed using energy minimization techniques. Geometric details are then recovered on the modified smooth shape. However, it is difficult to control the shapes of form-features precisely in a multiresolution framework.

PDE-based approaches directly deform the original mesh based on geometric constraints. These methods are categorized as non-linear and linear methods. Non-linear methods typically solve Laplacian or Poisson equations using non-linear iterative solvers [Bloor and Wilson 1990; Schneider and Kobbelt 2000; Desbrun et al. 1999; Yamada et al. 1999; Taubin 1995]. Catalano et al. [Catalao et al. 2002] investigated support tools for aesthetic design and showed non-linear PDE approaches are popular in computer-aided design. These methods produce fair surfaces, but it is time-consuming and difficult to deform shapes interactively.

We believe that linear PDE-based approaches are also useful for product design when product shapes undergo frequent design modifications. Linear PDE-based approaches represent differential properties and vertex positions in a linear system. Discrete Laplacian operators are often used to represent differential properties [Sorkine et al. 2004; Botsch and Kobbelt 2004a]. Yu et al. [Yu et al. 2004] introduced a similar technique called Poisson editing, which manipulates the gradients of the coordinate functions of the mesh. Zhou et al. [Zhou et al. 2005] proposed volumetric Laplacian operators for large deformations.

A discrete Laplacian operator on a mesh is defined as the difference vector between a vertex position and the weighted average position of its one-ring neighbors. Since Laplacians are defined in the local coordinate systems[Alexa 2003; Sorkine et al. 2004], one or more vertices must be specified in the global coordinate system to determine all vertex positions. When each vertex is constrained by the differential property, additional constraints for vertex positions lead to over-constraint situations. In existing methods, the least squares method is typically used for calculating compromised solutions. When Laplacians and positional constraints are described as linear system $M\mathbf{x} = \mathbf{b}$, the least squares system is represented as $M^tM\mathbf{x} = M^t\mathbf{b}$, and therefore vertex positions are calculated by $\mathbf{x} = (M^tM)^{-1}M^t\mathbf{b}$. Since $M^tM$ is a sparse symmetrical positive definite matrix, it can be efficiently factorized[Botsch et al. 2005]. After the matrix is factorized once, $\mathbf{x}$ is interactively calculated according to the modification of $\mathbf{b}$.

On the other hand, Welch and Witkin[Welch and Witkin 1992] introduced the combination of soft and hard constraints in variational surface modeling and solved them using Lagrange multipliers, although they did not apply them for interactive mesh editing. Yoshioka et al. [Yoshioka et al. 2006] solved hard constraints using equality-constrainted least sqyares. This method is very effective when constraints are restricted to simple positional constraints. However, it is not useful for form-feature constraints, because a large number of hard constraints with two or more variables make an equality-constrained least squares system less sparse. In this paper, we introduce hard constraints and new form-feature constraints for interactive mesh editing and solve them using the Lagrange multiplier.

Several authors have discussed methods for rotating the Laplacian vectors following the deformation of surfaces. Since the rotation is applied to $\mathbf{b}$ on the right-hand side of the least squares system, Laplacian vectors are rotated in an interactive way. Lipman et al. [Lipman et al. 2004] estimated the local rotations on the underlying smooth surface. Sorkine et al. [Sorkine et al. 2004] linearized elements in a rotation matrix assuming that rotation angles are very small and solved them as a linear system. Lipman et al. [Lipman et al. 2005] encoded rotations and positions using relative positions on local frames and solved them as two separate linear systems. Zayer et al. [Zayer et al. 2005] rotated Laplacian vectors using harmonic functions in $[0,1]$ based on discrete Laplace-Beltrami operators. They defined a unit quaternion at each vertex and interpolated four components of unit quaternions by assigning a single weight 1.0 to all handle vertices. Our approach is similar to Zeyer's method, although we incorporate constraints for the rotations of form-features and assign the logarithms of multiple unit quaternions for handle vertices.

## 3 Framework for Constrained Deformation

### 3.1 Constraints on positions

Let the mesh $\mathbb{M}$ be a pair $(K,\mathbb{P})$, where $\mathbb{P} = \{\mathbf{p_1},\ldots,\mathbf{p_n}\}$ and $\mathbf{p}_i = (x_i,y_i,z_i) \in \mathbb{R}^3$; $K$ is a simplicial complex that contains vertices $i$, edges $(i,j)$, and faces $(i,j,k)$. The adjacent vertices of vertex $i$ are denoted by $N(i) = \{j|(i,j) \in K\}$. The original position of $\mathbf{p}_i$ is referred to as $\mathbf{p_i^0} = (x_i^0,y_i^0,z_i^0)$.

When the normal vector and mean curvature of vertex $i$ are referred to as $\kappa_i$ and $\mathbf{n}_i$, the mean curvature vector $\kappa_i\mathbf{n}_i$ can be approximated using the following discrete form [Meyer et al. 2003];

$$\kappa_i\mathbf{n}_i = L(\mathbf{p}_i) = \frac{1}{4A_i}\sum_{j\in N(i)}(cot\alpha_{ij} + cot\beta_{ij})(\mathbf{p}_i - \mathbf{p}_j), \qquad (1)$$

where $\alpha_{ij}$ and $\beta_{ij}$ are the two angles opposite to the edge in the two

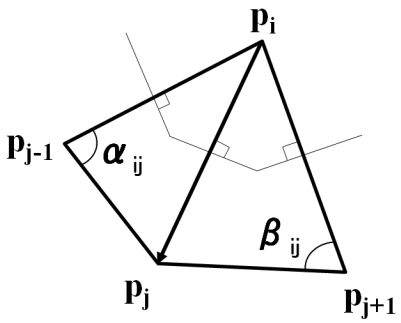


Figure 1: Definition of $\alpha_{ij}$ and $\beta_{ij}$.

triangles that share edge $(i, j)$, as shown in Figure 1. We denote the mean curvature vectors of the original mesh as $\delta_i (i = 1,2,\ldots,n)$.

In this paper, we describe additional constraints other than mean curvature vectors as $\mathbf{f}_j(\mathbb{P}_j) = \mathbf{u}_j \ (\mathbb{P}_j \subset \mathbb{P})$. Then constraints for vertices can be described as the following linear equations:

$$\begin{cases} L(\mathbf{p}_i) = R(\mathbf{n}_i, \theta_i)\delta_i \quad (i = 1,2,\ldots,n) \\ \mathbf{f}_j(\mathbb{P}_j) = R(\mathbf{m}_j, \phi_j)\mathbf{u}_j \quad (j = 1,2,\ldots,m), \end{cases} \tag{2}$$

where $n$ is the number of vertices; $m$ is the number of additional constraints; $R(\mathbf{n}, \theta)$ represents a rotation matrix that rotates a vector around axis $\mathbf{n} \in \mathbb{R}^3$ by angle $\theta \in \mathbb{R}$. $R(\mathbf{n}_i, \theta_i)$ and $R(\mathbf{m}_j, \phi_j)$ are calculated before Equation 2 is solved. We will describe a method for calculating rotation matrices in the next section.

Since the number of constraints in Equation 2 is larger than the one of variables, the exact solution does not exist. Therefore, we classify constraints in Equation 2 into soft and hard constraints, because the positional constraints need to be satisfied as precisely as possible in most engineering applications.

When we represent soft constraints as $A\mathbf{x} = \mathbf{b}$ and hard constraints as $C\mathbf{x} = \mathbf{d}$ in matrix forms, variables $\mathbf{x}$ that minimizes $||A\mathbf{x} - \mathbf{b}||^2$ subject to $C\mathbf{x} = \mathbf{d}$ can be calculated using the Lagrange multiplier as:

$$\min_{\mathbf{x}}(\frac{1}{2}||A\mathbf{x} - \mathbf{b}||^2 + \mathbf{y}^t(C\mathbf{x} - \mathbf{d})), \tag{3}$$

where $\mathbf{y} = (y_1, y_2, \ldots, y_m)^t$ are Lagrange multipliers. This minimization can be calculated using the following linear system:

$$M\tilde{\mathbf{x}} = \tilde{\mathbf{b}} \tag{4}$$

$$M = \begin{pmatrix} A^tA & C^t \\ C & 0 \end{pmatrix}, \quad \tilde{\mathbf{x}} = \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix}, \quad \tilde{\mathbf{b}} = \begin{pmatrix} A^t\mathbf{b} \\ \mathbf{d} \end{pmatrix}$$

This linear system determines the unique solution that satisfies hard constraints exactly. Matrix $M$ can be factorized using sparse direct solvers for linear symmetric systems[Gould et al. 2005].

We will note that when conflicting or redundant constraints are involved in a linear system, they lead to the rank deficiency, and the solver may halt the computation. We can resolve such over-constraint problems by applying Householder factorization to each column in matrix $C$, as shown in [Yoshioka et al. 2006]. If column $j$ in $C^T$ is a redundant constraint, the diagonal and lower elements of column $j$ are equal to zero after the previous $j-1$ columns are processed. Therefore, we can detect the redundant constraints. This process can be calculated very efficiently. See [Yoshioka et al. 2006] for more detail.

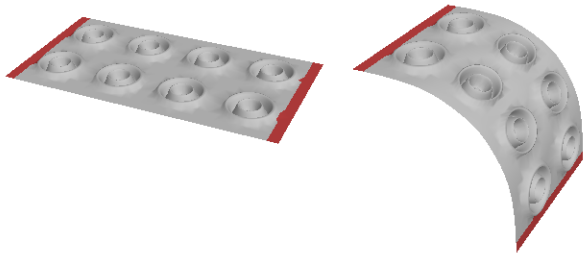

Figure 2: Deformation using rotated mean curvature vectors. Left: original shape; right: deformed shape.

### 3.2 Constraints on rotations

In this section, we describe how to calculate $R(\mathbf{n}_i, \theta_i)$ in Equation 2.

In our framework, we assign the logarithms of unit quaternions to all vertices. A quaternion can be written in the form:

$$Q = (w, x, y, z) = w + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}, \tag{5}$$

where $w, x, y, z \in \mathbb{R}$ and $\mathbf{i}, \mathbf{j}, \mathbf{k}$ are distinct imaginary numbers. When a quaternion has unit magnitude, it is called a unit quaternion and corresponds to a unique rotation matrix. A unit quaternion can be represented using rotation axis $\hat{\mathbf{n}}$ and rotation angle $\theta$:

$$\hat{Q} = e^{\hat{\mathbf{n}}\frac{\theta}{2}} = cos\frac{\theta}{2} + \hat{\mathbf{n}}sin\frac{\theta}{2}, \tag{6}$$

where $\hat{\mathbf{n}}$ is a pure quaternion. The logarithm of a unit quaternion is defined as the inverse of the exponential:

$$\mathbf{q} = ln\hat{Q} = \frac{\theta}{2}\hat{\mathbf{n}}. \tag{7}$$

We assign logarithm $\mathbf{q}_i \in \mathbb{R}^3$ to vertex $i$ and denote the logarithms assigned to all vertices as $\mathbb{Q} = \{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n\}$. When $\mathbf{q}_i$ is equal to 0, the mean curvature vector is not rotated; when $\mathbf{q}_i = \mathbf{v}_j$ is specified, the mean curvature vector is rotated around axis $\mathbf{v}_j/|\mathbf{v}_j|$ by angle $2|\mathbf{v}_j|$.

Shoemake [Shoemake 1985] proposed the spherical linear interpolation between two unit quaternions. Johnson[Johnson 2003] applied the spherical linear interpolation to multiple unit quaternions using the logarithms of unit quaternions. Pinkall and Polthier [Pinkall and Polthier 1993] proposed an interpolation technique using discrete conformal mapping. Zayer et al. [Zayer et al. 2005] applied discrete conformal mapping for interpolating unit quaternions. We introduce similar constraints on the logarithms of unit quaternions:

$$L(\mathbf{q}_i) = \frac{1}{4A_i}\sum_{j \in N(i)}(cot\alpha_{ij} + cot\beta_{ij})(\mathbf{q}_i - \mathbf{q}_j) = 0. \tag{8}$$

Then we introduce additional linear equations other than Equation 8, and describe them as:

$$\mathbf{g}_j(\mathbb{Q}_j) = \mathbf{v}_j \quad (\mathbb{Q}_j \subset \mathbb{Q}, \mathbf{v}_j \in \mathbb{R}^3). \tag{9}$$

As a result, linear equations for rotations can be described as:

$$\begin{cases} L(\mathbf{q}_i) = 0 \quad (i = 1,2,\ldots,n) \\ \mathbf{g}_j(\mathbb{Q}_j) = \mathbf{v}_j \quad (j = 1,2,\ldots,r), \end{cases} \tag{10}$$

where $n$ is the number of vertices; $r$ is the number of additional constraints on rotations. These equations construct a sparse linear system and can be solved using sparse direct solvers[Gould et al. 2005]. The solution of the linear system generates certain energy-minimization surfaces [Pinkall and Polthier 1993] in three dimensional space spanned by the logarithms of unit quaternions.

When all components of $\mathbb{Q}$ are calculated, rotation matrix $R(\mathbf{q}_i/|\mathbf{q}_i|, 2|\mathbf{q}_i|)$ is uniquely determined at each vertex. Figure 2 is an example of a deformed shape, in which the mean curvature vectors are rotated.

# 4 Preserving Form-Features

## 4.1 Preserving the shape of form-feature

We define a form-feature as a partial shape that has an engineering meaning, such as a hole and a protrusion. In mesh model $\mathbb{M}$, a form-feature consists of a subset of vertices $\mathbb{P}$. In this section, we denote a form-feature as $f$, the index set of vertices in form-feature $f$ as $\Lambda_f$.

When form-feature $f$ is translated and rotated according to the motion of handles while preserving its original shape, the following constrains regarding rotations and positions maintain the shape of the form-feature:

$$\begin{cases} \mathbf{q}_i - \mathbf{q}_j = 0 \quad (i,j \in \Lambda_f; (i,j) \in K) \\ \mathbf{p}_i - \mathbf{p}_j = s_f R(\mathbf{n}_i, \theta_i)(\mathbf{p}_i^0 - \mathbf{p}_j^0) \quad (i,j \in \Lambda_f; (i,j) \in K), \end{cases} \tag{11}$$

where $s_f$ is a scaling factor. These equations are added to Equation 2 and 10. Since each vertex in the form-feature has the same rotation matrix, $\mathbf{n}_i = \mathbf{n}_j$ and $\theta_i = \theta_j$ $(i,j \in \Lambda_f)$ in Equation 11.

In some cases, a form-feature has to keep the original direction depending on design intent. Then the following equations are added to Equation 2 and 10 instead of Equation 11:

$$\begin{cases} \mathbf{q}_i = 0 \quad (i \in \Lambda_f) \\ \mathbf{p}_i - \mathbf{p}_j = s_f(\mathbf{p}_i^0 - \mathbf{p}_j^0) \quad (i,j \in \Lambda_f; (i,j) \in K), \end{cases} \tag{12}$$

Figure 3 shows deformed shapes that preserve the shapes of holes as circles. As shown in Figure 3c-e, constraints on relative positions maintain the shapes of form-features. In Figure 3e, four small holes are constrained by Equation 11, but the center hole is constrained to preserve the original direction using Equation 12.

## 4.2 Constraining the motion of form-feature

In computer-aided design, it is useful to constrain the motion of a form-feature. The positions of cylindrical form-features are often specified using the center lines, and the positions of planar faces are often constrained on a specified plane.

In linear constraints, the motion of a form-feature can be maintained on a straight line or a plane by constraining a point in the form-feature. A constrained point can be specified as a linear function of vertex positions in the form-feature. For example, the center position of a circle can be specified as function $0.5(\mathbf{p}_i + \mathbf{p}_j)$ using two vertex positions on the opposite sides.

Here, we generally represent a linear combination of vertex positions as $\mathbf{x} = (x,y,z)$, where each of $x$, $y$ and $z$ is a linear function of coordinates $\{\mathbf{p}_i\}(i \in \Lambda_f)$, respectively.

### 4.2.1 On-plane constraints

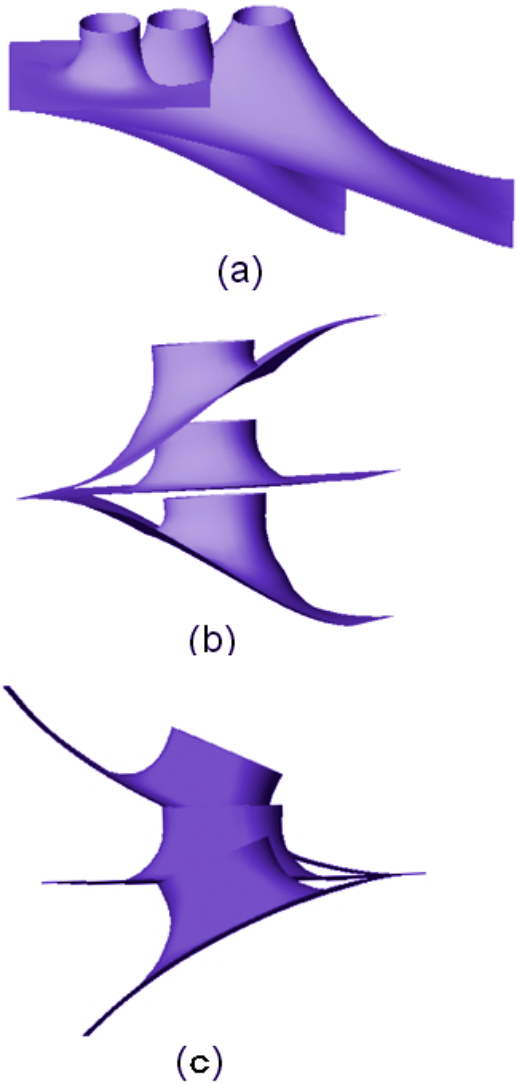


Figure 4: (a) Deformed shapes with a form-feature that moves on a plane. (b) Deformed shapes with a form-feature that moves on a line. (c) Deformed shapes with a rotated form-feature that moves on a line.

A plane is uniquely determined by its normal vector and a point on the plane. We represent the equation of a plane as $\mathbf{n}(\mathbf{x} - \mathbf{p}) = 0$, where $\mathbf{n} = (n_x, n_y, n_z)$ is the normal vector and $\mathbf{p} = (p_x, p_y, p_z)$ is a point on the plane. Then the following equation constrains the motion of a form-feature on a plane:

$$n_x x + n_y y + n_z z = n_x p_x + n_y p_y + n_z p_z \tag{13}$$

Since position $\mathbf{p}$ appears on the right-hand side of Equation 13, planes can be interactively moved to their normal directions. This capability is useful for the modification of an allowable margin to avoid interference.

Figure 4a shows deformed shapes in which a hole feature is constrained to preserve the shape with no rotations and to move on a

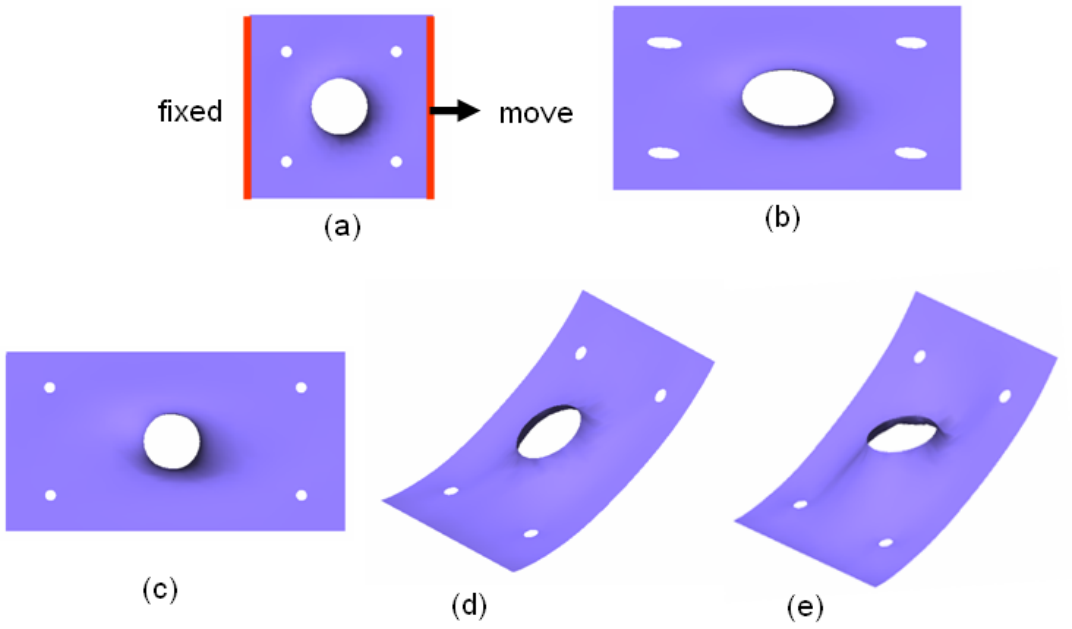


Figure 3: Constrained deformation. (a) Original shape; (b) deformed shape with no form-feature constraints; (c) stretched while preserving the shapes of circles; (d) deformed shape with five rotated circles; and (e) deformed while preserving the direction of the center circle.

plane.

#### 4.2.2 On-line constraints

A straight line can be represented as $\mathbf{x} = k\mathbf{n} + \mathbf{p}$. Let $\mathbf{l}$, $\mathbf{m}$ and $\mathbf{n}$ be unit vectors that are perpendicular each other. Then position $(x, y, z)$ moves on the straight line by using the following constraints:

$$\begin{cases} l_x x + l_y y + l_z z = l_x p_x + l_y p_y + l_z p_z \\ m_x x + m_y y + m_z z = m_x p_x + m_y p_y + m_z p_z \end{cases} \quad (14)$$

Straight lines can be moved interactively to directions that are perpendicular to $\mathbf{n}$. This capability is also useful, because it corresponds to the movement of the center positions of circles on 2D drawings.

Figure 4b-c show the constrained motion of a form-feature. The form-feature is constrained to preserve the direction in Figure 4b and to rotate in Figure 4c while moving on a straight line.

### 4.3 Feature extraction

In our system, the user first selects the region of a form-feature, and then adds linear constraints to the form-feature. It may be tedious work for the user to carefully select the region before specifying constraints. We introduce an interactive mesh segmentation technique for easy selection of feature regions.

So far, many algorithms have been reported for the segmentation of feature regions [Mangan and Whitaker 1999; Chazelle et al. 1997; Shlafman et al. 2002; Katz and Tal 2003; Katz et al. 2005]. Our segmentation algorithm is based on the method proposed by Katz et al. [Katz and Tal 2003], but we apply the method only to user-specified regions.

Figure 5 shows a feature extraction process. First, the user selects a region that includes the boundary of the feature region, as shown in Figure 5b. The region must be selected so that the mesh model is separated into exactly two regions. Then the optimal cut-set is calculated by the maximum flow, minimum cut algorithm. Finally, the feature region is separated, as shown in Figure 5c.

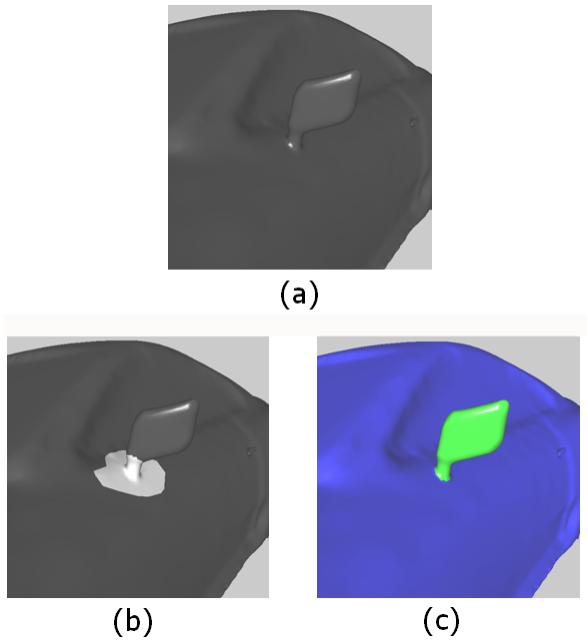


Figure 5: Feature extraction. (a) Original shape; (b) The region selected by the user; (c) The extracted region.

## 5 Experimental Results

Figure 6 shows examples of deformed shapes. In industrial design, character lines are essentially important. If a deformation process modifies the character lines of a product shape, the resultant shape is not accepted by designers. While the character lines are warped in Figure 6a, they are maintained in Figure 6b by defining constraints on the character lines.

Figure 7 shows a front grill part of an automobile model. While no form-feature constraints were specified in Figure7b, form-feature constraints are defined at cavities in Figure 7c so that the shapes and directions of form-features are preserved. Deformation in Figure 7b destroys the design intent, but Figure 7c maintains the characteristic features. In Figure 7d, scaling factors in Equation 12 are modified in an interactive manner.

Figure 8 is a sheet metal panel. 16 cavities shown by arrows are constrained so that they rotate while preserving the original shapes of cavities.

Table 1 shows CPU time for calculating the deformed models in Figure 6-8. The CPU time was measured for setting up matrices and factorizing them on a PC with 1.50-GHz Pentium-M and 1 GB of RAM. Once the matrix was set up and factorized, the shape could be deformed in interactive rate. This result shows that the performance of our framework is good for practical use.

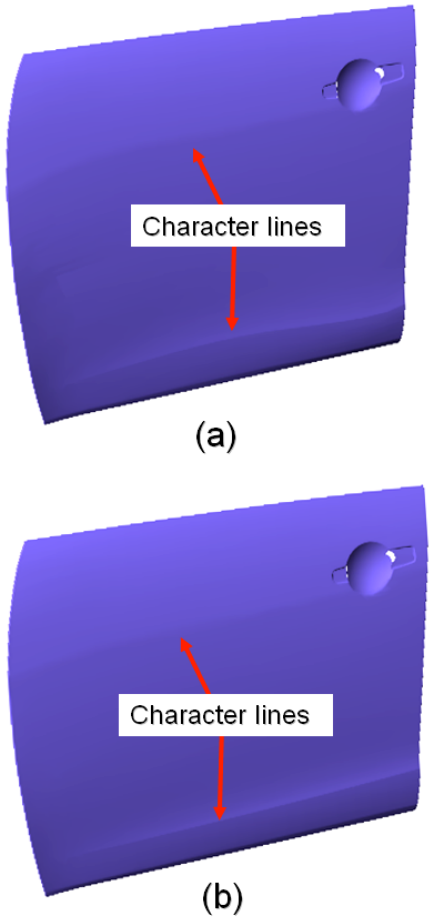


Figure 6: Door panel. (a) Shape deformed without form-feature constraints. The character lines are warped. (b) Shape deformed with form-feature constraints on character lines.

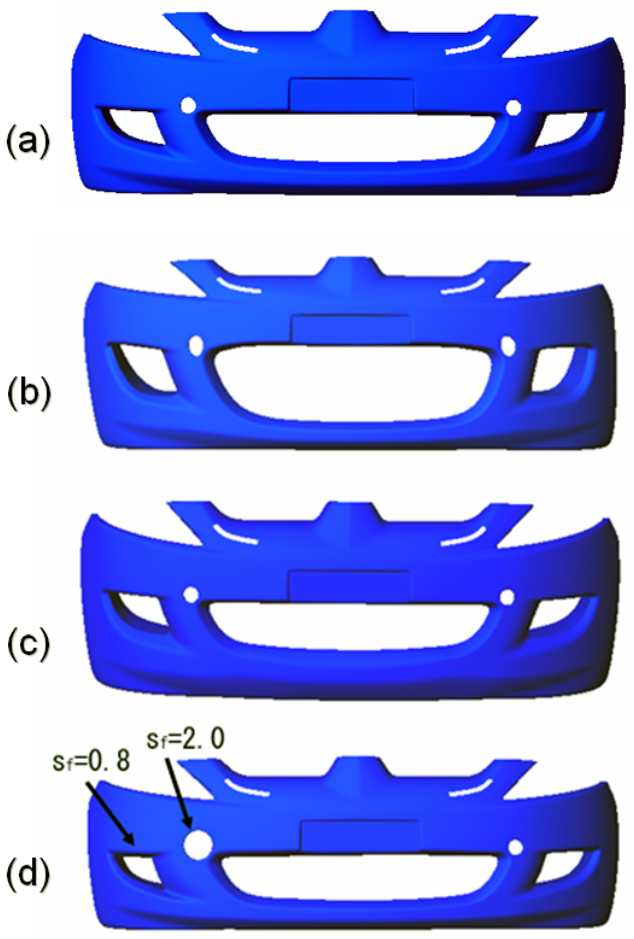


Figure 7: Front grill. (a) Original shape. (b) Shape deformed without form-feature constraints. (c) Shape deformed with form-feature constraints. (d) Interactive scaling of form-features.

| | Vert | Soft | Hard | Feat | Time |
|---|---|---|---|---|---|
| Figure 7 | 3337 | 5796 | 3826 | 3702 | 0.86 |
| Figure 8 | 13974 | 25458 | 9334 | 1072 | 4.69 |
| Figure 9 | 2982 | 6084 | 3308 | 3202 | 0.99 |

Table 1: CPU time for computation. [Vert]: number of vertices; [Soft]: number of soft constraints; [Hard]: number of hard constraints; [Feat]: number of form-feature constraints; [Time]: CPU time(sec).

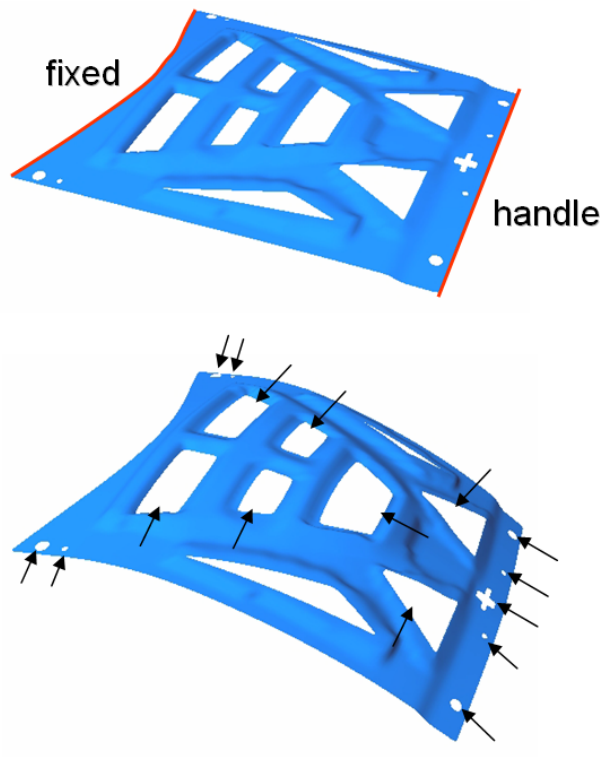


Figure 8: Sheet metal part. Top: original planer shape. Bottom: deformed shape with rotated form-features shown by arrows.

## 6 Conclusions and Future Work

We have presented a discrete framework for incorporating constraints for form-features using hard constraints. Deformed shapes are calculated so that constraints on rotations and positions are satisfied. Constraints on rotations and positions are separately solved as two sparse symmetrical matrices, which are known for the existence of efficient solvers. We showed how to constrain the shape and motion of form-features; linear shape constraints can constrain the shapes of form-features to the same shape and linear motion constraints confine motion to movement on a plane or a straight line. These constraints are convenient for deforming the 3D models of sheet metal panels.

In future work, it will be important to develop more intuitive tools for specifying form-features and constraints on mesh models. Since a complex product shape contains a considerable number of form-features, the automatic detection of features is preferable. In addition, it will be useful to incorporate a geometric reasoning engine into our framework, since commercial geometric engines are available and widely used. Well-known problems on hard constraints include the management of inconsistent and redundant constraints. Since we have developed a mechanism for resolving such constraints, we will incorporate it into our framework.

## References

ALEXA, M. 2003. Differential coordinates for local mesh morphing and deformation. *The Visual Computer 19*, 2-3, 105–114.

BLOOR, M. I. G., AND WILSON, M. J. 1990. Using partial differential equations to generate free-form surfaces. *Computer-Aided Design 22*, 4, 202–212.

BOTSCH, M., AND KOBBELT, L. 2004. An intuitive framework for real-time freeform modeling. *ACM Transactions on Graphics 23*, 3, 630–634.

BOTSCH, M., AND KOBBELT, L. 2004. A remeshing approach to multiresolution modeling. In *Symposium on Geometry Processing*, 189–196.

BOTSCH, M., BOMMES, D., AND KOBBELT, L. 2005. Efficient linear system solvers for mesh processing. In *IMA Conference on the Mathematics of Surfaces*, 62–83.

CATALAO, C. E., FALCIDIENO, B., GIANNINI, F., AND MONTI, M. 2002. A survey of computer-aided modeling tools for aesthetic design. *Journal of Computer and Information Science in Engineering 2*, 11, 11–20.

CAVENDISH, J. C. 1995. Integrating feature-based surface design with freeform deformation. *Computer-Aided Design 27*, 9, 703–711.

CHAZELLE, B., DOBKIN, D. P., SHOURABOURA, N., AND TAL, A. 1997. Strategies for polyhedral surface decomposition: An experimental study. *CGTA: Computational Geometry: Theory and Applications 7*, 4-5, 327–342.

COQUILLART, S. 1990. Extended free-form deformation: a sculpturing tool for 3D geometric modeling. In *Proceedings of SIGGRAPH 1990*, ACM Press, 187–196.

DAM, E. B., KOCH, M., AND LILLHOLM, M. 1998. Quaternions, interpolation and animation. Tech. Rep. DIKU-TR-98/5, Department of Computer Science, University of Copenhagen.

DEMMEL, J., GILBERT, J., AND LI, X. 1995. *SuperLU "User's Guide"*.

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of SIGGRAPH 1999*, ACM Press/Addison-Wesley Publishing Co., 317–324.

ECK, M., DEROSE, T., DUCHAMP, T., HOPPE, H., LOUNSBERY, M., AND STUETZLE, W. 1995. Multiresolution analysis of arbitrary meshes. In *Proceedings of SIGGRAPH 1995*, ACM Press, 173–182.

FONTANA, M., GIANNINI, F., AND MEIRANA, M. 1999. A free form feature taxonomy. *Computer Graphics Forum 18*, 3, 703–711.

GOULD, N. I. M., HU, Y., AND SCOTT, J. A. 2005. A numerical evaluation of sparse direct solvers for the solution of large sparse, symmetric linear systems of equations. Tech. Rep. RAL-TR-2005-005, Council for the Central Laboratory of the Research Councils.

GUSKOV, I., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution signal processing for meshes. In *Proceedings of SIGGRAPH 1999*, ACM Press/Addison-Wesley Publishing Co., 325–334.

HU, S.-M., LI, Y., JU, T., AND ZHU, X. 2001. Modifying the shape of nurbs surfaces with geometric constraints. *Computer Aided Design 33*, 12, 903–912.

HU, S.-M., ZHANG, H., TAI, C.-L., AND SUN, J.-G. 2001. Direct manipulation of ffd: Efficient explicit solutions and decom-

posible multiple point constraints. *The Visual Computers 17*, 6, 370–379.

JOHNSON, M. P. 2003. *Exploiting quaternions to support expressive interactive character motion*. PhD thesis, Massachusetts Institute of Technology, School of Architecture and Planning, Program in Media Arts and Sciences.

KARYPIS, G., AND KUMAR, V. 1998. *MeTis: Unstrctured Graph Partitioning and Sparse Matrix Ordering System, Version 4.0*.

KATZ, S., AND TAL, A. 2003. Hierarchical mesh decomposition using fuzzy clustering and cuts. *ACM Transactions on Graphics 22*, 3, 954–961.

KATZ, S., LEIFMAN, G., AND TAL, A. 2005. Mesh segmentation using feature point and core extraction. *The Visual Computer(Pacific Graphics) 21*, 8-10, 649–658.

KOBBELT, L., CAMPAGNA, S., VORSATZ, J., AND SEIDEL, H.-P. 1998. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of SIGGRAPH 1998*, ACM Press, 105–114.

LEE, S. 1999. Interactive multiresolution editing of arbitrary meshes. *Computer Graphics Forum 18*, 3, 73–82.

LIPMAN, Y., SORKINE, O., COHEN-OR, D., LEVIN, D., RÖSSL, C., AND SEIDEL, H.-P. 2004. Differential coordinates for interactive mesh editing. In *SMI 2004: Proceedings of the international conference on Shape Modeling and Applications*, 181–190.

LIPMAN, Y., SORKINE, O., LEVIN, D., AND COHEN-OR, D. 2005. Linear rotation-invariant coordinates for meshes. *ACM Transactions on Graphics 24*, 3, 479–487.

MACCRACKEN, R., AND JOY, K. I. 1996. Free-form deformations with lattices of arbitrary topology. In *Proceedings of SIGGRAPH 1996*, 181–188.

MANGAN, A. P., AND WHITAKER, R. T. 1999. Partitioning 3D surface meshes using watershed segmentation. *IEEE Trans. Vis. Comput. Graph 5*, 4, 308–321.

MASUDA, H., FURUKAWA, Y., YOSHIOKA, Y., AND YAMATO, H. 2004. Volume-based cut-and-paste editing for early design phases. In *ASME/DETC2004/CIE: Design Engineering Technical Conferences and Computer and Information Engineering Conference*.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2003. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, H.-C. Hege and K. Polthier, Eds. Springer-Verlag, 35–57.

NEALEN, A., SORKINE, O., ALEXA, M., AND COHEN-OR, D. 2005. A sketch-based interface for detail-preserving mesh editing. *ACM Transactions on Graphics 24*, 3, 1142–1147.

PINKALL, U., AND POLTHIER, K. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics 2*, 1, 15–36.

SCHNEIDER, R., AND KOBBELT, L. 2000. Generating fair meshes with g1 boundary conditions. In *GMP 2000: Proceedings of the international conference on Geometric Modeling and Processing*, 251–261.

SEDERBERG, T. W., AND PARRY, S. R. 1986. Free-form deformation of solid geometric models. In *Proceedings of SIGGRAPH 1986*, ACM Press, 151–160.

SHLAFMAN, S., TAL, A., AND KATZ, S. 2002. Metamorphosis of polyhedral surfaces using decomposition. *Computer Graphics Forum 21*, 3.

SHOEMAKE, K. 1985. Animating rotation with quaternion curves. In *Proceedings of SIGGRAPH 1985*, 245–254.

SORKINE, O., LIPMAN, Y., COHEN-OR, D., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *SGP 2004: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, ACM Press, 175–184.

SORKINE, O. 2005. Laplacian mesh processing. In *STAR Proceedings of Eurographics 2005*, 53–70.

TAUBIN, G. 1995. A signal processing approach to fair surface design. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM Press, 351–358.

TOLEDO, S., CHEN, D., AND ROTKIN, V. 2003. Taucs: A library of sparse linear solvers. http://www.tau.ac.il/ stoledo/taucs/.

WELCH, W., AND WITKIN, A. 1992. Variational surface modeling. In *Proceedings of SIGGRAPH 1992*, vol. 26, 157–166.

YAMADA, A., FURUHATA, T., SHIMADA, K., AND HOU, K.-H. 1999. A discrete spring model for generating fair curves and surfaces. In *Pacific Conference on Computer Graphics and Applications*, 270–279.

YOSHIOKA, Y., MASUDA, H., AND FURUKAWA, Y. 2006. A constrained least-squares approach to interactive mesh deformation. In *SMI 2006: Proceedings of the international conference on Shape Modeling and Applications*.

YU, Y., ZHOU, K., XU, D., SHI, X., BAO, H., GUO, B., AND SHUM, H.-Y. 2004. Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics 23*, 3, 644–651.

ZAYER, R., RÖSSL, C., KARNI, Z., AND SEIDEL, H.-P. 2005. Harmonic guidance for surface deformation. *Computer Graphics Forum 24*, 3, 601–609.

ZHOU, K., HUANG, J., SNYDER, J., LIU, X., BAO, H., GUO, B., AND SHUM, H.-Y. 2005. Large mesh deformation using the volumetric graph laplacian. *ACM Transactions on Graphics 24*, 3, 496–503.

ZORIN, D., SCHRÖDER, P., AND SWELDENS, W. 1997. Interactive multiresolution mesh editing. In *Proceedings of SIGGRAPH 1997*, 259–268.