Month 200X, Vol.21, No.X, pp.XX–XX *J. Comput. Sci. & Technol.*

FEA-Mesh Editing with Feature Constrained

Chuhua Xian, Shuming Gao, Hongwei Lin*, and Dong Xiao

State Key Lab. of CAD & CG, Zhejiang University, Hangzhou, 310058, P. R. China

E-mail: {xianchuhua, smgao, hwlin, xiaodong}@cad.zju.edu.cn

Received MONTH DATE, YEAR.

Abstract

This paper proposes a framework for FEA-mesh editing with feature constrained. In the framework, cage-based technique is first used to edit the base-decomposition model. Vertices of the constrained feature are transformed into a local form, and then reconstructed after determining the common boundary. Feature rotation constraint derives from the normal change of the boundary plane. Parameters are analyzed before editing operation, and our method permits the user to add constraints on the parameters of the feature. This framework can also keep consistence for the disconnected assembly mesh model. Experimental results show that constrained features hold precisely after mesh editing. Additionally, experimental data validate the efficiency of our method, achieving real-time response.

Keywords CAE, FEA-mesh, Mesh Editing, Feature, Feature Constraints

1 1 Introduction

In mechanical engineering, products are often evaluated by CAE analysis to check whether they satisfy engineering requirements. Usually, a rough 3D model is first created by CAD software, and then CAE tools are employed for its structure or dynamic analysis. If the analyzing results do not satisfy the engineering requirements, the CAD model has to be modified using CAD software and analyzed again by CAE tools. Thus, the CAD model modification and CAE analysis procedure is repeated till a desirable product is generated. In fact, prior to CAE analysis, the CAD model is required to be *re-meshed* for finite element analysis (CAE analysis). The re-meshing procedure is very time-consuming, and the mesh quality is critical to the CAE analysis. Therefore, in order to avoid totally re-meshing the CAD model in each *modification-and-analysis* repetition, we develop a method to edit the CAE mesh directly while keeping some features [1, 2] and constraints. It should be pointed out that, in CAE area, there are two kinds of meshes, volu-

*Corresponding author.

This paper is supported by the National Natural Science Foundation of China under Grant No.60736019.

metric mesh and surface mesh; we only concern the later one in this paper.

Generally, the interactive mesh editing methods can be classified into two categories. One is cage-based (or volume-based) and the other is surface-based.

Cage-based mesh editing techniques are mainly used in computer graphics [3]. These methods usually employ a coarse cage enclosing the mesh model to manipulate the mesh, by representing the mesh vertices as the affine combination of vertices of the outer cage [4, 5, 6]. Cage-based techniques can process disconnected models such as assembly models, while they are difficult to manage the constraints specified at feature regions.

Surface-based mesh editing techniques [7, 8] calculate differential properties at the discrete vertices, and then use the partial equations to reconstruct the surface which leads to solving a sparse linear system. However, if there are no additional constraints, the form feature may be distorted while mesh editing. To keep the form features, Masuda et al. [9] introduces the soft and hard constraints into the deforming framework. The hard constraints which require to be satisfied precisely work on the feature regions, and the soft constraints are fulfilled in the least-square sense. Although this method can preserve the form feature, it only works globally and can not process complex feature regions locally or partly, such as enlarging the radius of a through hole while keeping its height.

In this paper, cage-based and surface-based mesh editing techniques are combined to develop a CAE mesh-editing framework that can manage constraints on features. In our work, the cage-based mesh editing technique manipulates the global deformation, while the surface-based one deals with constrains on feature regions. Our method not only preserves the form of the feature, but also permits the user to keep some inner parameters of the feature, such as the height of column and the length of the extrusion. The main contributions of this paper are three-fold: (1) we propose a mesh-editing framework that can modify parameters and preserve the feature while editing a CAE-mesh model; (2)we introduce a local coordinate transformation which makes vertices of a feature only relate to the common boundary, so that the constrained feature can be reconstructed after determination of the common boundary; (3)we employ the cage-based editing technique to make the constraints of the disconnected parts of the model consistent while mesh editing.

The remainder of this paper are organized as follows. Section 2 gives a general review on related work. Section 3 introduces the mesh editing framework. Section 4 is devoted to applying the cage-based techniques to edit the base decomposition model. To be self-contained, our prior-work about cage generation is also introduce in this section. Then, in section 5, we present constraints on features. Finally, we give some experimental results in

section 6 and conclude this paper in section 7.

2 2 Related Work

In recent decades, a wide variety of mesh-editing techniques have been developed. These techniques can be classified into two categories: cage-based and surface-based.

Cage-based techniques deform shapes by modifying the space in which objects lie [3, 10]. Chen [11] uses free-form deformation(FFD) to generate a series paradigm of CAE-meshes based on a basic design for CAE-based simulation. Inheriting the idea of FFD, cage-based methods construct a cage to envelop a mesh, and the mesh vertices are represented by the affine sums of the cage's vertices and its face normals [4, 5, 6]. Users manipulate the cage to induce a smooth space deformation. The cage-based mesh editing techniques are simple, flexible and efficient. Additionally, they can process disconnected models, such as assembly models in engineering. However, it is difficult to use these approaches to keep the specified feature regions without attaching any constraints on the model.

Based on the theory of partial differential equation(PDE), Yu et al. [8] develop a surface-based mesh-editing method by manipulating the gradient field of the coordinate functions of the mesh, and solving the Poisson equation to get the positions of the vertices. Sorkine [7] introduces a similar technique which uses the discrete Laplacian-Beltrami operater to define differential properties at vertices and imposes positional constraints to form a linear system. These surface-based methods handle the mesh directly, so they are useful for the engineering design.

In order to make use of surface-based techniques for deforming automobile sheet-metal panels, Masudaa et al. [9] develop the soft and hard constraints on the mesh and propose a framework which can preserve the form features of the sheet-metal panel while deforming the model. However, this method can not propagate deformation to disconnected meshes. To fix this problem, in Ref. [12], Masudaa defines virtual links between pairs of disconnected vertices, thus spreading the deformation to disconnected parts. Specifically, this method is employed to deform assembly models [12]. However, since this method needs to select the pairs of vertices from disconnected parts by manual, it is very tedious to deal with complex assembly model. Further more, this method aims at only sheet metal deformation, so, when deforming other types of models in engineering, it only scales the feature globally and does not permit to modify part of the feature parameters. It should be pointed out that, there may be millions of vertices in CAE mesh models. Therefore, the surface-based techniques will lead to solving a very large sparse system, which is a time-consuming task even by applying current sparse solvers [13].

3 3 Framework of Constrained Editing

In this paper, we propose a CAE-mesh editing framework, which provides four basic operators shown in Figure 1, that is:

- the decomposition operator(*feature separation*),
- the editing operator(*shape deformation*),
- the feature constraining operator(*feature analysis*), and
- the reconstruction operator(*model synthesis*).

In our current system, after the users specify features and add constraints, decomposition operator is performed. It can be given by,

$$B_i = S_i \ominus F_i, \tag{1}$$

and

$$D = \bigcup\nolimits_i B_i \oplus R, \tag{2}$$

where S_i is the surface with specified feature F_i, B_i is generated by subtracting the constrained feature F_i from S_i, R is the remainder part of the CAE-mesh, D is the *Base Decomposition Model*(abbr. *BDM*), $\ominus$ is a minus operator, and $\oplus$ is a summation operator. The mesh-editing operation employs the cage-based techniques, which will be discussed in section 4. After the constrained features have been selected, the system will calculate the boundaries and analyze parameter constraints of the specified features, and then the constrained relations will be preserved during the deforming process. Finally, the reconstruction is done by combining the modified $B_i \rightarrow B_i{}'$, the constrained features $F_i \rightarrow F_i{}'$, and $R \rightarrow R'$, that is,

$$M = \bigoplus\nolimits_i (B_i{}' \oplus F_i{}') \oplus R'. \tag{3}$$

4 4 Base-decomposition Model Editing

As presented in Eqs. (1) and (2), the base-decomposition model (*BDM*) is the model getting rid of the constrained features. Since the cage-based techniques are able to handle the disconnected assembly model and perform very efficiently for large mesh with lots of vertices, in this paper, a recent developed cage-based technique, Green Coordinates(*GC*) [6], is employed to deform the models. Given a *BDM* D, we first construct a cage C that envelops the edited part, and represent each vertex on the part as a weighted combination of cage vertices and face normals. As the cage changes, D is deformed in turn by applying the GC weights to the deformed cage.

A good cage is important to the cage-based mesh editing, which should loosely encloses the mesh. The following section presents a method for constructing the cage of a mesh model.

4.1 4.1 Cage Generation

As shown in Figure 2, the cage generation algorithm mainly includes the following steps:

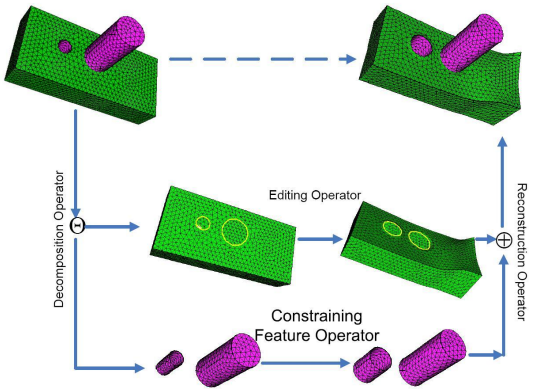

Figure 1: A CAE-mesh editing framework with feature constrained.

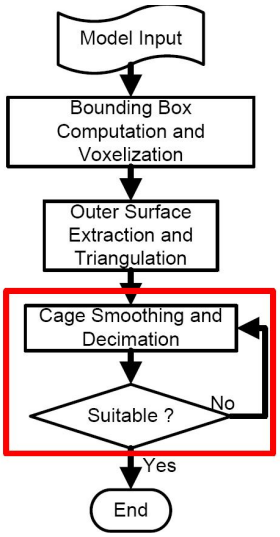

Figure 2: Flowchart of cage generation.

1. Compute the bounding box and voxelize the mesh model;

2. Extract and triangulate the outer faces of the feature voxels;

3. Smooth the cage by an improved mean curvature flow method, and decimate the cage.

4. If the cage is not suitable for editing , goto step 3.

Details on this method can be found in [14]. After generating the initial cage, users can adjust it if necessary.

5 5 Constraints on Features

Feature is a partial part that has special meaning in engineering such as hole or protrusion. Figure 3 shows some features on a part. For a CAE-mesh model, feature is a sub-mesh on the specified region. In this section, we introduce the constraints imposed on a feature.

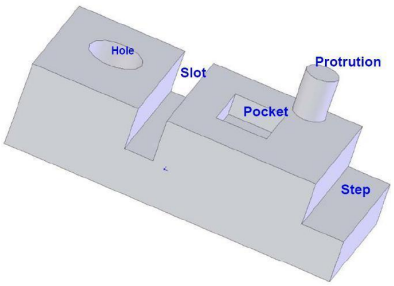

Figure 3: Part and features.

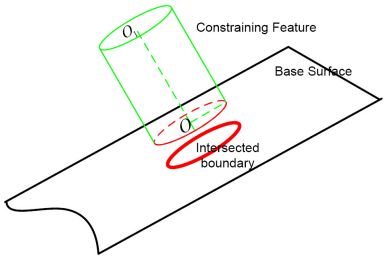

Figure 4: Common boundary of feature and base surface.

5.1 5.1 Local Coordinates Transformation

As mentioned in section 3, constrained features will be separated from the base surface of a mesh model. The common boundary of the feature and base surface will be detected while performing decomposition operator as shown in Figure 4. As stated above, a constrained feature $F(V,T)$ is a sub-mesh of the CAE-mesh, where V is its vertex set and T is its face set. We denote the index set of the vertices in F by Γ_F, and that in the common boundary by Ω_F. Then a local transformation is defined on each vertex $v_i (i \in \Gamma_F - \Omega_F)$ of the feature as,

$$\Lambda(v_i) = \frac{1}{d_{\Omega_F}} \sum\nolimits_{j \in \Omega_F} (v_i - v_j), \qquad (4)$$

where $d_{\Omega_F} = |\Omega_F|$ is the number of vertices in the common boundary. Intuitively, the local transformation (Eq. (4)) presents a local translation from the barycenter of the common boundary to the vertex v_i(see *left* in Figure 5).

To preserve the form of the feature precisely, we add the constraint,

$$\Lambda(v_i{}') = f(R_F \Lambda(v_i)), \qquad (5)$$

where R_F is the rotation matrix which will be introduced in the following, and $f(\bullet)$ is a constraining function about parameters, which will be presented in Section 5.3.

The transformation from the original vector to the vector $\Lambda(v)$ in Eq. (4) can be regarded as a local coordinate defined by the boundary. From the definition in Eq. (4), if we translate the vertex v_i in the feature F by the vector u

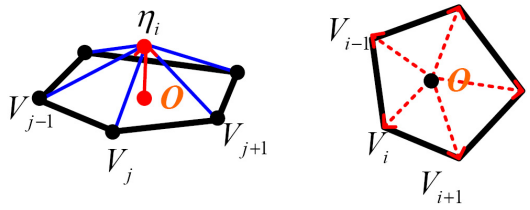

Figure 5: (*left*) Local coordinates translation from boundary vertices; (*right*) common boundary constraints.

to obtain the new vertex $v_i{}'$, we have

$$\begin{aligned}\Lambda(v_i{}') &= \frac{1}{d_{\Omega_F}}\sum\nolimits_{j\in\Omega_F}(v_i{}' - v_j{}') \\ &= \frac{1}{d_{\Omega_F}}\sum\nolimits_{j\in\Omega_F}((v_i+u)-(v_j+u)) \\ &= \frac{1}{d_{\Omega_F}}\sum\nolimits_{j\in\Omega_F}(v_i - v_j) \\ &= \Lambda(v_i).\end{aligned}$$

This means that the reconstruction is not unique and the vertices of the feature depend on the common boundary.

In order to reconstruct the feature after performing editing operator, we need to determine Ω_F. Let $O = \frac{1}{d_{\Omega_F}}\sum_{i\in\Omega_F} v_i$ is the barycenter of the common boundary, which is regarded as a virtual vertex of *DBM* and treated in the same manner as other vertices. For each vertex $v_i (i \in \Omega_F)$, a scalar function Ξ is defined as (right in Figure 5),

$$\Xi(v_i) = ||v_i - O||. \tag{6}$$

We have therefore additional constraints of the form:

$$v_i{}'' - O' = f(\Xi(v_i)(v_i{}' - O')), i \in \Omega_F, \tag{7}$$

where $v_i{}'$ and O' denote the vertices after editing, and $f(\bullet)$ is the same as in Eq. (5). After performing reconstruction operator, Eq.(7) will be satisfied, and feature F can be reconstructed from Eqs.(4) and (5).

5.2 5.2 Rotation Constraint

When we edit the mesh, the constrained features need to be rotated to fit the normal of the base surface. After the boundary has been reconstructed, rotation matrix in Eq. (5) can be determined from the normal of the common boundary. As illustrated in Figure 6, n, n' and O are the original normal, new normal and the barycenter of the common boundary, respectively. The rotation matrix R_F in Eq. (5) is obtained by rotating n to n' around barycenter O. As shown in Figure 5(b), we add virtual connections between O and the boundary vertices, and then n and n' can be obtained by the average normals of adjacent triangular faces. Because the local transformation defined in Eq. (4) only relates to the common boundary, each vertex in the constrained feature has the same rotation matrix. Figure 7 shows some examples deformed with rotation and without rotation, respectively. The constrained features in Figure 7(c) are more reasonable in practice.

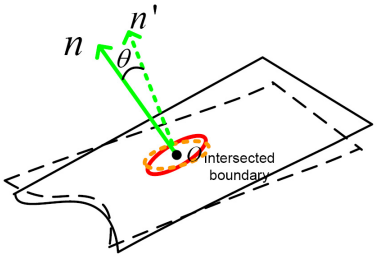

Figure 6: Normal rotation of the common boundary.

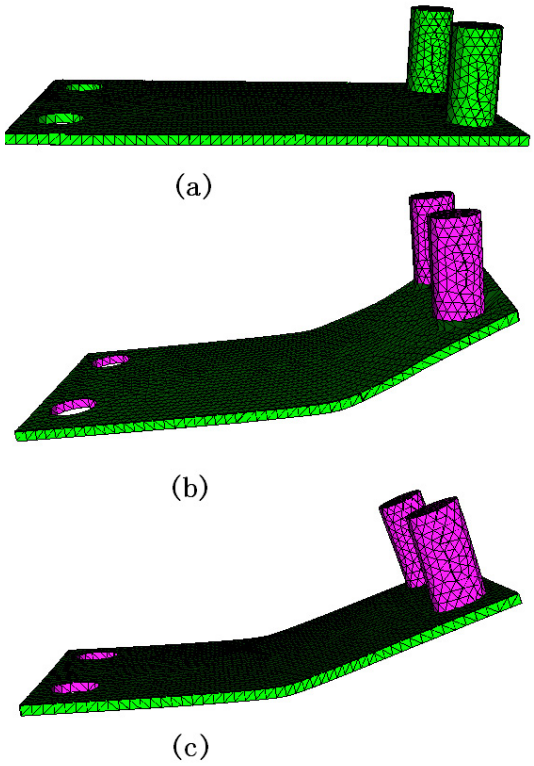

Figure 7: Deformed model. (a)Original model; (b)deformed model without feature rotation; (c)deformed model with feature rotation.

5.3 5.3 Constraint on Parameters

In section 5.1, function $f(\bullet)$ is used to constrain feature vertices. If we want to preserve the form of the feature F precisely and not to modify its parameters, $f(\bullet)$ should be an identity function. It means that only translation and rotation are performed on F. This constraint is sufficient in most applications. But in some cases, we need to modify some parameters of the feature.

Suppose a feature has m parameters, and $\overrightarrow{g_i}(\vec{c}_i, \bullet)$ $(i = 0, 1, ..., m-1)$ is a constraining function on parameter $\vec{c}_i$. Then f in Eq. (5) is rewritten as,

$$f(\bullet) = \sum_{i=0}^{m-1} \overrightarrow{g_i}(\vec{c}_i, \bullet). \qquad (8)$$

In most commercial CAD software nowadays, to design a CAD model, 2D sketch will be first created, and the feature operations, such as

extrusion or revolution, are utilized to generate a feature. Different types of features have different sketches and parameters. In this section, we mainly analyze the parameters of through holes and protrusions with circular and rectangular shape. Other types can be analyzed in a similar way.

Circular hole and protrusion: In traditional design procedure, users first create a circular sketch on a reference plane, and then extrude it to get a cylindrical feature. Suppose $\overrightarrow{n}$ is the normalized normal vector of the reference plane, $\vec{\hbar} = R_F\Lambda(\eta_i)$ (Eq. (5)), $\vec{r}$ is the normalize vector of $\vec{\hbar}-(\vec{\hbar}\cdot\vec{n})\vec{n}$, and $\vec{h}$ is the normalized vector along the height extrusion direction (Figs 8(a,b),9). Then, constraints $\overrightarrow{g}(\vec{r},\bullet)$ on $\vec{r}$ and $\overrightarrow{g}(\vec{h},\bullet)$ on $\vec{h}$ are

$$\overrightarrow{g}(\vec{r},\vec{\hbar}) = s_r\vec{\hbar}\sin\alpha, \tag{9}$$

and,

$$\overrightarrow{g}(\vec{h},\vec{\hbar}) = s_h\vec{\hbar}\cos\beta, \tag{10}$$

where s_r and s_h are scalar factors, α is the angle between the normal of sketch plane $\vec{n}$ and vector variable $\vec{\hbar} = R_F\Lambda(\eta_i)$ in Eq. (5) or $\Xi v_i({v_i}' - O')$ in Eq. (7), and β is the angle between the extrusion direction $\vec{e}$ and $\vec{\hbar}$ (see Figure 9).

Rectangular hole and protrusion: Figure 8(c) and (d) show three parameters, length l, width w and height h. We define $\vec{l}$, $\vec{w}$ and $\vec{h}$ as the normalized vectors along the extruding directions of length, width, and height, respectively. Similarly, we have

$$\overrightarrow{g}(\vec{l},\vec{\hbar}) = s_l\vec{\hbar}\cos\gamma, \tag{11}$$

and,

$$\overrightarrow{g}(\vec{w},\vec{\hbar}) = s_w\vec{\hbar}\cos\vartheta, \tag{12}$$

where γ is the angle between $\vec{l}$ and $\vec{\hbar}$ (Figure 9(c)), and ϑ is the angle between $\vec{w}$ and $\vec{\hbar}$. Constraint on $\vec{h}$ can be treated using Eq. (10).

It should be noted that the FEA-mesh model has no information about the feature sketch, so it is difficult to recognize the sketch plane. In our work, common boundary is recognized automatically, and we use the boundary plane to replace the sketch plane. For the hole feature, there are two common boundaries (see Figure 8). Users can choose one as the reference.

6 6 Implementation and Results

The framework of FEA-mesh editing with feature constrained developed in this paper is implemented with *VC++ 2005* and *OpenGL*, and runs on the PC with Core 4^{TM} 2.4GHZ and 4GB RAM in a single thread. In our experiments, all mesh models are generated by FEA tools.

Figure 10b shows an deformed part without feature constraints. We can see that three circular features have transformed into elliptical shapes. However, with feature constraints in Figure 10c, circular features preserve the shape during deforming the model.

Figure 11 is a bracket model. The editing operation is to elongate its horizontal length and four holes in the soleplate while preserving their shapes.

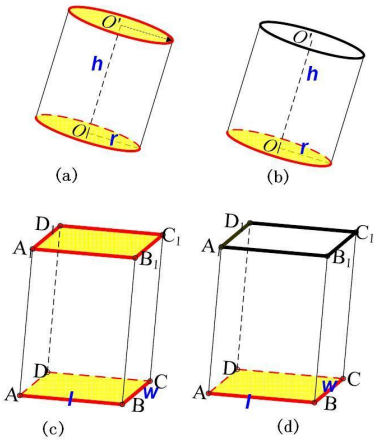

Figure 8: Parameters of features. (a)Circular through hole with two common boundaries; (b)circular protrusion; (c)rectangular holes with two common boundaries; (d)rectangular protrusion.

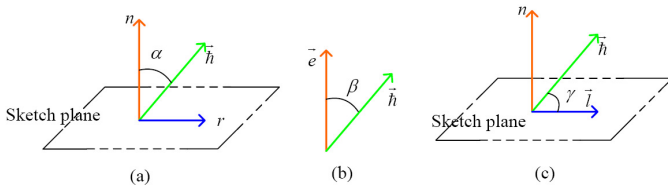

Figure 9: Constraints on parameters.

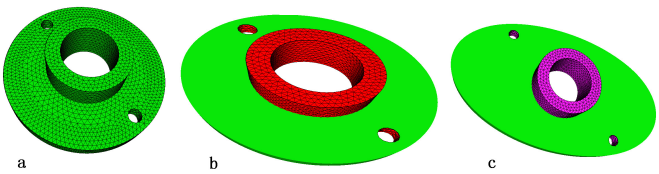

Figure 10: (a) Original model; (b) deformed model without feature constraints, where circular features in red have turned into elliptical shapes; (c) deformed model with feature constraints, where circular features keep the original shape.

Figure 12 and 13 show parameter constraints on features. The radii of circular holes in Figure 12 are shrunk and the height of the protrusion is stretched. The scalar factor s_r is 0.7 in Eq. (9), and s_h is 2.0 in Eq. (10).

The assembly model in Figure 14 consists of two pipes and four bolts. There is a gap between the two pipes. Figure 14 shows the deformed shape with four hole constraints. In Figure 14c, four bolts are constrained and assembly information keeps consistent after editing the model.

Moreover, a car model is shown in Figure 15. For this model, Computational Fluid Dynamics (*CFD*) programs are usually used to calculate various aerodynamics properties in re-

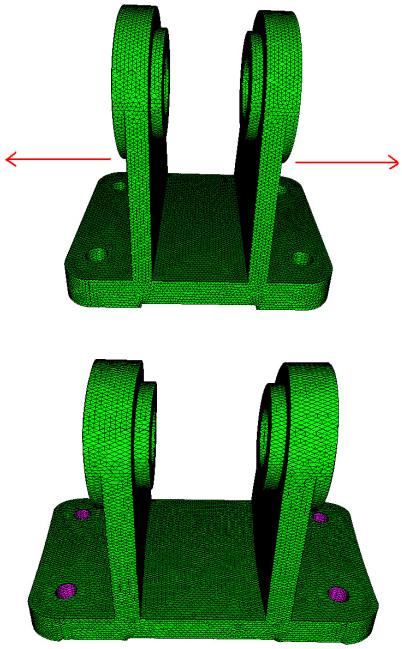

Figure 11: Bracket. Top: original model; bottom: deformed model with four constrained through holes.

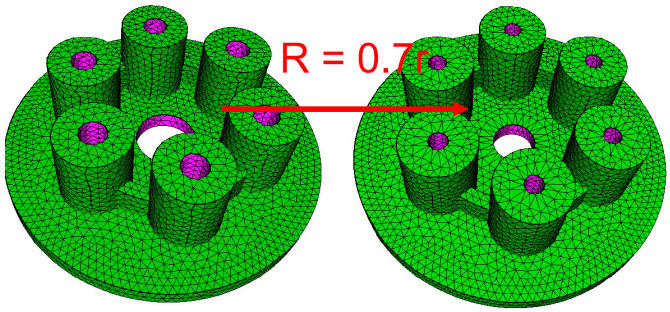

Figure 12: (a) Original model; (b) deformed model with scaling radius.

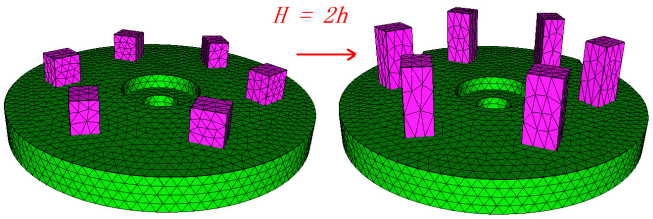

Figure 13: (a) Original model; (b) deformed model with scaling height of the protrusion.

sponse to a set of vehicle exterior parameters. During the simulation, the surface may need to be modified. But the shape of some regions, such as cartwheels, should keep unchanged during the editing operation. In our system, these feature regions are constrained and preserved while performing deformation on the model.

Table 1 shows the time for initialization

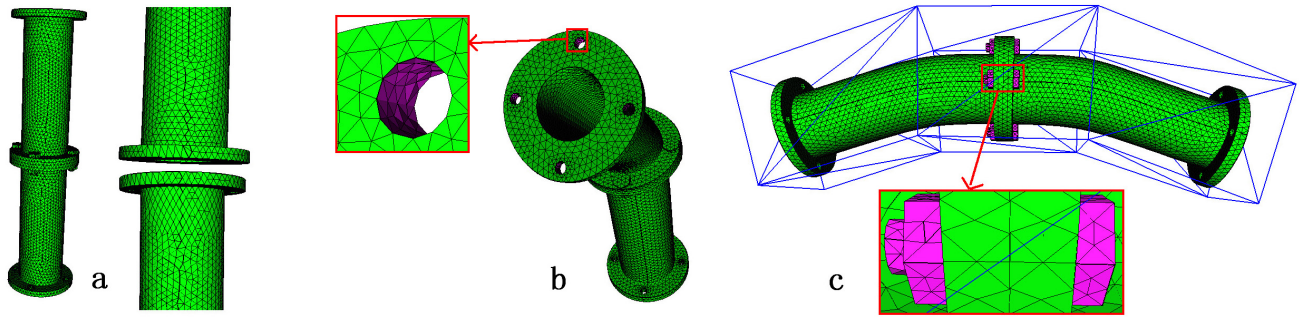

Figure 14: Assembly model. (a) *left.* original model, *right.* two disconnected parts; (b) deformed model with four constrained through holes; (c) deformed model with four constrained bolts.

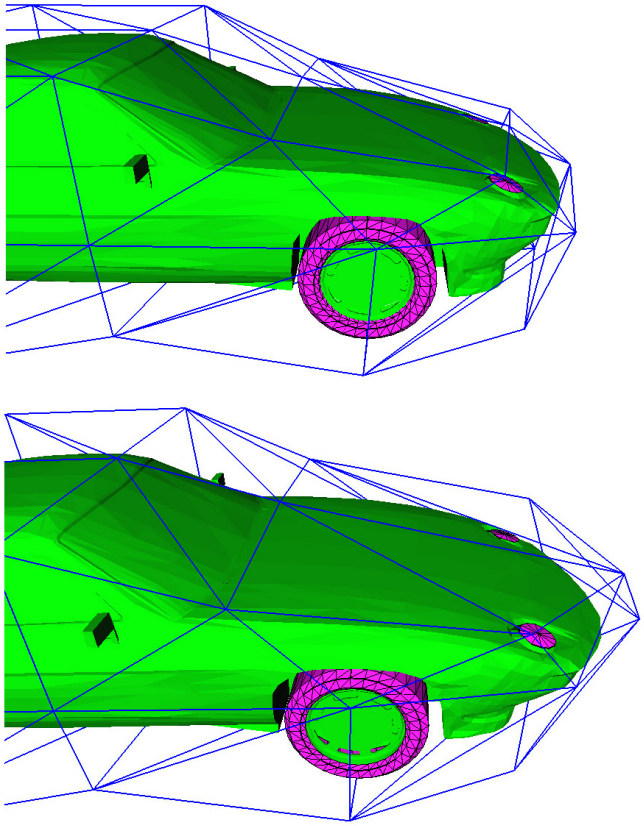

Figure 15: Car. *Top*: original model, where the cartwheels and lights are constrained; *bottom*: deformed model, where the constrained features keep their shape forms.

and mesh editing in Figures 10-15. From the table, we can see that time is mainly spent on the initialization for GC evaluation [6], while the editing time is within 0.001s. This result shows that the method presented in this paper is efficient.

7 7 Conclusion

This paper focuses on developing a CAE-mesh framework with feature constrained. The whole framework consists of two major processes: editing the base-decomposition model

Table 1: Data on models and time for initialization and editing

	#Verts.	#Feature Verts.	#Cage Verts.	Initialization(sec)	Editing(sec)
Fig. 10	4018	6634	4	0.216	0.001
Fig. 11	32746	974	4	1.763	0.010
Fig. 12	6260	1374	4	0.338	0.001
Fig. 13	3661	788	12	0.327	0.001
Fig. 14b	11876	473	20	1.907	0.007
Fig. 14c	11876	1106	16	1.486	0.007
Fig. 15	4123	602	42	1.508	0.007

and imposing constraints on features. We use the cage-based technique to edit the base-decomposition model. Vertex of feature is transformed into a local form, which makes it only relate to the common boundary. Furthermore, rotation matrix of the feature can be obtained by the normal rotation of the common boundary. We analyze constraints on feature parameters. Our method can preserve the feature shape and permit user to add parameter constraints. Experimental results show that assembly mesh models keep consistent when performing editing operation.

In our current system, features are selected by manual. To be more convenient, one of our future work is to recognize the features and assembly information of the CAE-mesh automatically. Another future work is to analyze more types of practical features to improve our system.

References

[1] M. Fontana, F. Giannini, and M. Meirana. A free-form feature taxonomy. *Computer Graphics Forum*, 1999, 18(3): 107-118.

[2] J. Han, M. Pratt, and W. C. Regli. Manufacturing feature reconition from solid models : A status report. *IEEE Transactions on Robotics and Automation*, 2000, 16(6): 782-796.

[3] W. S. Thomas, R. P. Scott. Free-form deformation of solid geometric models. In *Proceedings of SIGGRAPH 1986*, 1986, pp. 151-160.

[4] M. S. Floater, G. Kos, M. Reimers. Mean value coordinates in 3d. *Computer Aided Geometry Design*, 2005, 22: 623-631.

[5] T. Ju, S. Schaefer, J. Warren. Mean value coordinates for closed triangular meshes. In

Proceedings of SIGGRAPH 2005, 2005, pp. 561-566.

[6] Y. Lipman, D. Levin, D. Cohen-Or. Green coordinates. *ACM Transaction on Graphics*, 2008, 27(3): .

[7] O. Sorkine, D. Cohen-Or, Y. Lipman, C. Rossl, H. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2004, pp. 175-184.

[8] Y. Yu, K. Zhou, X. Shi, H. Bao, B. Guo, H. Shum. Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics*, 2004, 23(3): 644-651.

[9] H. Masudaa, Y. Yoshiokaa, Y. Furukawab. Preserving form features in interactive mesh deformation. *Computer-Aided Designed*, 2007, 39(5): 361–368.

[10] R. MacCracken, K. Joy. Free-form deformations with lattices of arbitrary topology. In *Proceedings of SIGGRAPH 1996*, 1996, pp. 181-188.

[11] Y. Chen, P. Stewart, A. Marsan. A mesh feature paradigm for rapid generation of CAE-based design of experiments data. In *Proceedings of ASME DETC 2002*, Montreal, Canada, September 2002.

[12] H. Masudaa, K. Ogawa. Application of interactive deformation to assembled mesh models for CAE analysis. In *Proceedings of the ASME 2007*, Las Vegas, Nevada, USA, September 4-7 2007.

[13] N. Gould, Y. Hu, J. Scott. A numerical evaluation of sparse direct solver for the solution of large sparse, symmetric linear system equation. *Council for the Central Laboratory of the Research Council*, Tech. Rep. RAL-TR-2005-05.

[14] C. Xian, H. Lin, S. Gao.Automatic generation of coarse bounding cages from dense meshes. In *Proceedings of IEEE International Conference on Shape Modeling and Applications (SMI) 2009*, Accepted.