Visual Comput (2006) 22: 266–275
DOI 10.1007/s00371-006-0004-8

ORIGINAL ARTICLE

Xiaogang Jin
Juncong Lin
Charlie C.L. Wang
Jieqing Feng
Hanqiu Sun

Mesh fusion using functional blending on topologically incompatible sections

Published online: 6 April 2006
© Springer-Verlag 2006

X. Jin (✉) · J. Lin · J. Feng
State Key Lab of CAD & CG,
Zhejiang University, Hangzhou, 310027,
P.R. China
{jin,linjuncong,jqfeng}@cad.zju.edu.cn

C.C.L. Wang
Department of Automation and
Computer-Aided Engineering,
The Chinese University of Hong Kong,
Shatin, N.T., Hong Kong
cwang@acae.cuhk.edu.hk

H. Sun
Department of Computer Science &
Engineering, The Chinese University of
Hong Kong, Shatin, N.T., Hong Kong
hanqiu@cse.cuhk.edu.hk

Abstract Three-dimensional mesh fusion provides an easy and fast way to create new mesh models from existing ones. We introduce a novel approach of mesh fusion in this paper based on functional blending. Our method has no restriction of disk-like topology or one-ring opening on the meshes to be merged. First of all, sections with boundaries of the underfusing meshes are converted into implicit representations. An implicit transition surface, which joins the sections together while keeping smoothness at the boundaries, is then created based on cubic Hermite functional blending. Finally, the implicit surface is tessellated to form the resultant mesh. Our scheme is both efficient and simple, and with it users can easily construct interesting, complex 3D models.

Keywords Mesh fusion · Functional blending · Interactive modeling tool

1 Introduction

The function of creating detailed 3D models from existing objects with parts of interest is widely expected in many computer graphics applications. The main challenges are as follows. 1) How to preserve the local surface detail and create a gradual transition between the details of the two surfaces in the vicinity of the joint, and 2) how to adjust the combining process and create a seamless natural result.
Although Boolean operations using point-based representations [1, 21] or CSG modeling based on implicit surfaces [20] can combine several object parts together conveniently, we are only dealing with meshes, since they are now the de facto standard of free-form surface representations.

Surface cut-and-paste provides a good means for the expected function. Biermann et al. [7] presented a set of algorithms based on multiresolution subdivision surfaces to perform surface cut-and-paste operations. They separate both the source and the target surfaces into base and detail, then the source feature is pasted onto the target surface with a user-specified location and orientation. However, the joined objects are required to be topologically equal to a disk for the necessary mappings between source and target in their method. Kanai et al. [17] proposed a mesh fusion scheme based on three-dimensional mesh-based metamorphosis. They first establish polygon correspondences between two meshes using the method in [16]; then generate a smooth transition by interpolating corresponding points from the source to the target positions using the constructed correspondences. The same disk-like topology limitation exists in their method. Fu et al. [15] is the only example we know of in this class that explicitly deals with the non-zero genus case. They firstly construct a base surface passing through the boundary vertices of the selected region using

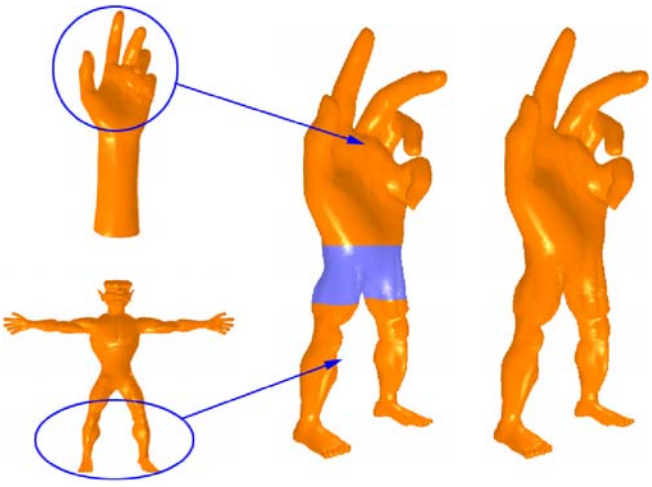


Fig. 1. Modeling by mesh fusion. Left: object parts adopted for fusion. Middle: the model fused by functional blending (the part in blue is a newly created transition surface). Right: final fusion result

the boundary triangulation technique. Then a new detail encoding technique is applied after surface parameterization. Finally, the detail representation is transferred onto the target surface via the base surface.

Differential approaches can also be employed in cut-and-paste for mesh processing. Yu et al. [26] adopted Poisson-based gradient field manipulation in mesh editing. They treated the mesh geometry as scalar functions defined on a mesh surface and introduced the Poisson equation as a mesh solver. This approach is actually a deformation-based method, so when applying it to merge meshes, similar boundary openings are required. In other words, it is hard to directly fuse a mesh with a one-ring opening onto a mesh with two or more openings; this has been elegantly solved in our mesh fusion scheme (e.g., see the example given in Fig. 1). Sorkine et al. [23] provided a Laplacian representation and implemented a surface mesh transplanting operation based on this representation, but the one-ring limitation still exists.

Implicit representation is another technique that can be used. Singh and Parent [22] provided a procedural implicit function defined for the region of a polyhedral object that is star-shaped with respect to a skeletal point. Then, this function was used to construct transition surfaces joining polyhedral objects. With the development of level set theory, Museth et al. [19] provided a level set framework for implanting surface editing operators locally and globally.

In this paper, we develop a functional blending based mesh fusion method to join two or even more objects. Figure 1 is a demonstration of our scheme: we first cut off the palm from a hand model and two legs from an alien; a transition surface is then created to fuse the three parts together smoothly.

The proposed mesh fusion scheme relates to some existing techniques of functional blending. Bedi [5] introduced a functional blending method based on the Bernstein polynomial to generate a surface that blended bifurcated sections. However, his method does not consider boundary smoothness, and the sections must be given in algebraic form. Elber [12] generalized the concept of blending surfaces for functional and ornamental purposes. The extended shaping operations offered in his work can be applied between boundaries of two adjacent surfaces, or to the interior of a single surface guided by arbitrary parametric curves in the domain of the patch. Elber [12] also conducted the Hermite basis functions to generate blending surfaces. However, the rail curves in his approach are represented in parametric form. Moreover, his method cannot deal with bifurcated cases. Differently from these approaches, our method adopts the Hermite basis to give an implicit representation blending two sections that hold openings to be fused. Finally, the implicit transition surface is tessellated into a polygonal mesh joining given parts of interest.

Our work is different from surface reconstruction from parallel planar contours [2–4, 10] and volume-based shape blending [24]. Surface reconstruction approaches do not consider the smoothness across the boundary or the shape control between each pair of slices. Volume-based shape blending focuses on constructing a sequence of interpolated shapes given two or more source shapes, so that blended shapes adjacent to each other in the sequence are geometrically close.

Compared with traditional surface cut-and-paste operations, we neither make a blending between the overlapped areas nor a deformation around the boundary, both of which need to show the compatible topology on boundaries. We choose to create a transition surface to smoothly join these objects together so as to overcome those limitations existing in previous work.

Our work may share some similar ideas with mesh repairing. For example, MPU [20](or RBF [9]) has proven to be a very good method for guessing and generating unknown parts of an incomplete mesh model. However, if we directly adopt those techniques in a global manner, that is treat all vertices of merging objects as constraints and then reconstruct an implicit function from them, it will be time-consuming; if we use them in a local style (make only the boundary vertices as constraints), it is hard to obtain a pleasing result due to inadequate constraints (see our experiment results in Section 4 for more details). Moreover, our scheme offers an explicit means of shape control and can blend the details of merging objects.

Although we can merge several objects (represented in an implicit function, point, etc.) together using set operation in CSG, it does not seem easy to apply this operation directly on a mesh. We also use an implicit function, however, our goal is to fuse several mesh represented objects together. The implicit function just acts as a transition surface, it will finally be converted into meshes.

In general, the major features of our mesh fusion scheme are the following.

1. *Smooth fusion*: Thanks to Hermite interpolation, the tangential continuity across the joining boundaries is preserved.
2. *Detail preservation*: The surface details of combining parts are mixed gradually in the transition surface.
3. *Seamless and natural fusion*: We provide several efficient means to adjust the shape of the transition surface. Our method also allows under-fusion parts to be at an arbitrary distance from each other. Using these control methods, the user can always obtain a nice result.
4. *Small topological genus restriction*: There are no restrictions of disk-like topology or star-shaped blending area as in previous work. The number of openings on the meshes to be fused is also unlimited.

In the following sections, we address the details of our mesh fusion approach. After giving the mesh fusion framework, necessary mathematical and algorithmic realization issues are described. Experiences with our prototype system indicate that our method is both efficient and easy to use for creating complex 3D models. Some interesting experimental results are shown at the end of the paper.

2 Functional blending based mesh fusion

Suppose that we have two polygonal objects M_1 and M_2 to be fused, and the openings on them are defined as Γ_1 and Γ_2. In order to create a new model by merging M_1 and M_2, a surface S needs to be constructed between Γ_1 and Γ_2. The surface is called *blending surface* or *transition surface*. If S can be functionally defined, the modeling method of merging M_1 and M_2 is called *functional blending* [5]. Hermite basis functions can be utilized to generate S between Γ_1 and Γ_2 if the cross tangents of S at the openings are also given.

Since the objects M_1 and M_2 are given in mesh representation, it is difficult to describe the openings by parametric curves, especially for the case of openings with n-rings. Here, we employ an implicit representation to formulate the openings. To simplify the problem, the openings of a given object are assumed to be coplanar (i.e., $\Gamma_i \in P_i$). When using $\boldsymbol{x} = (x, y)$ to represent a point on the plane P_i (x and y are 2-DOF for a point on P_i), the opening can be denoted by an implicit function as $\Gamma_i(\boldsymbol{x}) = 0$. Adopting a third parameter w to blend the change from Γ_1 to Γ_2, a functional can be defined as

$$\Psi(w) = \Gamma_1 F_1(w) + \Gamma_2 F_2(w) + T_1 F_3(w) + T_2 F_4(w), \quad (1)$$

where $F_i(w)$ are the Hermite basis functions with $w \in [0, 1]$,

$$(F_1(w), F_2(w), F_3(w), F_4(w)) = ((w-1)^2(1+2w), w^2(3-2w), w(w-1)^2, w^2(w-1)),$$

Γ_i are implicit functions defined on the openings of given objects, and T_i represent the change of Γ_i along the w direction. It is easy to find that $\Psi(0) = \Gamma_1(\boldsymbol{x})$ and $\Psi(1) = \Gamma_2(\boldsymbol{x})$. Thus, with the change of w from 0 to 1, $\Psi(w)$ gives a transition function that is blended from $\Gamma_1(\boldsymbol{x})$ to $\Gamma_2(\boldsymbol{x})$.

For any specific value $w = w_0$, a curve is implicitly defined by the function $\Psi(w_0) = 0$. Therefore, the functional $\Psi(w) = 0$ with $w \in [0, 1]$ actually defines the blending surface S between Γ_1 and Γ_2. After tessellating $\Psi(w) = 0$, the mesh fusion result of given objects can finally be determined.

3 Mathematical and algorithmic realization

For implementing the above functional blending, we need to address the following problems.

- The implicit definition for openings.
- The description of transition surfaces between openings.
- The shape control of transition surfaces.
- The tessellation of functional surface.

3.1 Implicit definition of openings

In order to present the openings Γ_i on M_i, usually polygonal rings, we need to create an implicit definition for each Γ_i. Since the openings belonging to one object are assumed to be coplanar, we can develop a function $\Gamma_i(x, y)$ to define the polygons on Γ_i implicitly by $\Gamma_i(x, y) = 0$, where x and y are local coordinates on the plane P_i containing Γ_i.

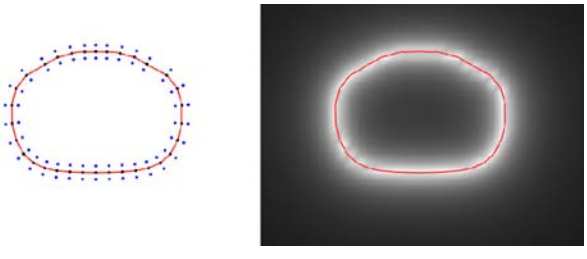


Fig. 2. Implicit definition of openings by scattered data interpolation. Left: example pairs of boundary points and offset points. Right: the determined RBF interpolation function illustrated by a grey image

One method to define such a function is to use two-dimensional *scattered data interpolation* [25], which can be described as finding a smooth unknown map $\mathcal{R}^2 \to \mathcal{R}$ interpolating a given set of distinct nodes $\{\{x_i, y_i\}\}_{i=1}^N \subset \mathcal{R}^2$.

We firstly determine two types of points on P_i: *boundary points* and *offset points*. Every polygon vertex on Γ_i is a boundary point on P_i with its function value assigned to zero. Meanwhile, the offset points of polygon vertices on Γ_i are computed in both inner and outer sides. For example, in the left-hand side of Fig. 2, the black points are the boundary points on an opening, while the blue ones are their relevant offset points. We assign the outer offset points with a positive constant value, and the inner points with a negative one. Choosing different values leads to different blending surfaces, and this will be illustrated later in the shape control section.

Then we use the *radial basis function* (RBF) as the interpolation function $\Gamma(\cdots)$. A radial basis function is usually expressed in the form

$$\Gamma(\boldsymbol{x}) = p(\boldsymbol{x}) + \sum_{i=1}^{N} \lambda_i \phi(|\boldsymbol{x} - \boldsymbol{x}_i|), \tag{2}$$

where $p(\cdots)$ is a linear basis function

$$p(\boldsymbol{x}) = p_0 + p_1 x + p_2 y, \tag{3}$$

with x and y representing the x- and y-components of point $\boldsymbol{x}$ and the basis function $\phi(\cdots)$ is a real function with its value falls in the interval $[0, \infty)$. For all test examples in this paper, we adopt the thin-plate radial basis function $\phi(\boldsymbol{r}) = |\boldsymbol{r}|^2 \log(|\boldsymbol{r}|)$. To uniquely define the function $\Gamma(\boldsymbol{x})$, we need to determine the coefficients of the linear basis $\{p_0, p_1, p_2\}$ and the radial basis function weights λ_i. There are a total of $N+3$ unknowns. However, based on the condition $\Gamma(\boldsymbol{x}_i) = f_i$ $(i = 1, \ldots, N)$, only N equations are given, so the following orthogonality conditions are introduced to give three more constraints:

$$\sum_{i=1}^{N} \lambda_i = \sum_{i=1}^{N} \lambda_i x_i = \sum_{i=1}^{N} \lambda_i y_i = 0. \tag{4}$$

By adopting ϕ_{ij} to denote $\phi(\boldsymbol{x}_i - \boldsymbol{x}_j)$, the linear equation system to determine $\Gamma(\boldsymbol{x})$ can be written as

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{C} \\ \boldsymbol{C}^T & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda} \\ \boldsymbol{P} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{0} \end{bmatrix}, \tag{5}$$

where

$$A_{ij} = \phi_{ij}, \quad i, j = 1, 2, \ldots, N,$$
$$C_{i1} = 1, \quad C_{i2} = x_i, \quad C_{i3} = y_i, \quad i = 1, 2, \ldots, N,$$
$$\boldsymbol{\Lambda} = \{\lambda_1, \lambda_2, \cdots, \lambda_N\}^T,$$
$$\boldsymbol{P} = \{p_0, p_1, p_2\}^T,$$
$$\boldsymbol{F} = \{f_1, f_2, \cdots, f_N\}^T.$$

The system is symmetric and positive definite unless all the vertices are colinear, so there exists a unique solution [6]. By solving the above linear equation system, the function of openings is uniquely determined by the function values assigned on boundary points and offset points. The right-hand side of Fig. 2 adopts a grey image to illustrate the interpolation function determined by the polygonal opening given in the left-hand side.

Another simple and convenient method to define $\Gamma_i(\cdots)$ is to use a 2D signed distance-field (SDF). For a point $\boldsymbol{x} = (x, y)$ on the plane P_i containing Γ_i , the function $\Gamma_i(\boldsymbol{x})$ for $\boldsymbol{x} \in P_i$ returns the signed Euclidean distance from $\boldsymbol{x}$ to Γ_i, where negative means that $\boldsymbol{x}$ is inside a ring of Γ_i and positive means that it is outside. $\Gamma_i(\boldsymbol{x}) = 0$ represents the case of $\boldsymbol{x}$ being exactly on Γ_i. We sample the 2D signed distance function on regular planar grids for efficiency, and the value inside a grid is calculated through bilinear interpolation of the values stored at grid nodes. The more grids that are adopted, the more accurately function $\Gamma_i(\cdots)$ is defined by SDF.

We have adopted both RBF and SDF to give the implicit definition of openings. Both schemes can generate nice results. We will give a detailed comparison between them in Sect. 4.

3.2 Surface description

After determining the implicit definition of openings on two given objects, Hermite basis functions are employed to generate the transition surface between openings. However, as mentioned earlier, not only Γ_i but also the changes of Γ_i along w direction (i.e., T_i) are needed to formulate the blending surface. Here, we determine T_i through numerical differences.

Without loss of generality, an opening Γ_i on a given object M_i is determined by defining a raw plane P_i to intersect the object and removing polygons on a specified side of the plane. For the elements passing through P_i, they are subdivided into two elements by P_i. Before removing the useless elements, two offset planes P_i^+ and P_i^- of P_i are generated to intersect M_i, so two intersection curves Γ_i^+ and Γ_i^- are computed. Using the above implicit definition method, two functions, $\Gamma_i^+(\boldsymbol{x})$ and $\Gamma_i^-(\boldsymbol{x})$, can be defined on Γ_i^+ and Γ_i^-. Then, the function describing the change of Γ_i is formulated by numerical difference [11] as

$$T_i = \frac{\Gamma_i^+(\boldsymbol{x}) - \Gamma_i^-(\boldsymbol{x})}{2\Delta h}, \tag{6}$$

where Δh is the distance between an offset plane and P_i. An example of computing T_i by offset planes is illustrated in Fig. 3.

To simplify the implementation, the raw planes that we currently adopt in testing examples are all parallel to the xy plane. A function $w = \eta(z)$ is introduced to describe

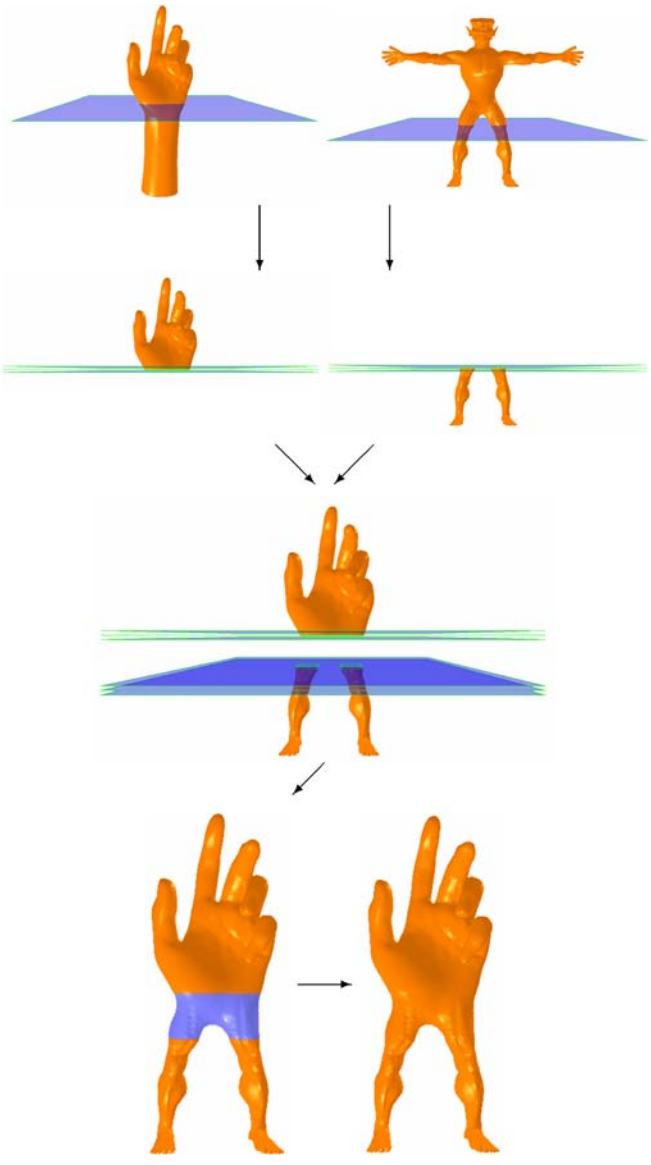


Fig. 3. Using offset planes to formulate the T_i for surface description

the mapping between the z-coordinate and the blending parameter w. Together with $w = \eta(z)$ and T_i defined in Eq. 6, the blending functional given in Eq. 1 is determined. In future work we will discuss the case where raw planes on M_1 and M_2 are not parallel to each other.

3.3 Shape control

In this section, we will discuss the shape control of a transition surface taking the RBF case as an example. Similar conclusions can be drawn for the SDF case.

The shape of the transition surface is controlled by adjusting the mapping function $\eta(\cdots)$ and the RBF parameters in our approach.

Two constraints should be imposed on the function $\eta(\cdots)$. Firstly, its value should be $\eta(0) = 0$ and $\eta(1) = 1$ in order to satisfy the position continuity at the two ends of $\Psi(w)$. The cross-derivative of $\Psi(w)$ can be found as

$$\frac{\partial \Psi}{\partial z} = \Gamma_1 \frac{\partial F_1}{\partial w}\frac{\partial \eta}{\partial z} + \Gamma_2 \frac{\partial F_2}{\partial w}\frac{\partial \eta}{\partial z} + T_1 \frac{\partial F_3}{\partial w}\frac{\partial \eta}{\partial z} + T_2 \frac{\partial F_4}{\partial w}\frac{\partial \eta}{\partial z}.$$

Secondly, to ensure the tangent continuity cross the ends of $\Psi(w)$, the constraints $\frac{\partial \Psi}{\partial z}|_{z=0} = T_1$ and $\frac{\partial \Psi}{\partial z}|_{z=1} = T_2$ are

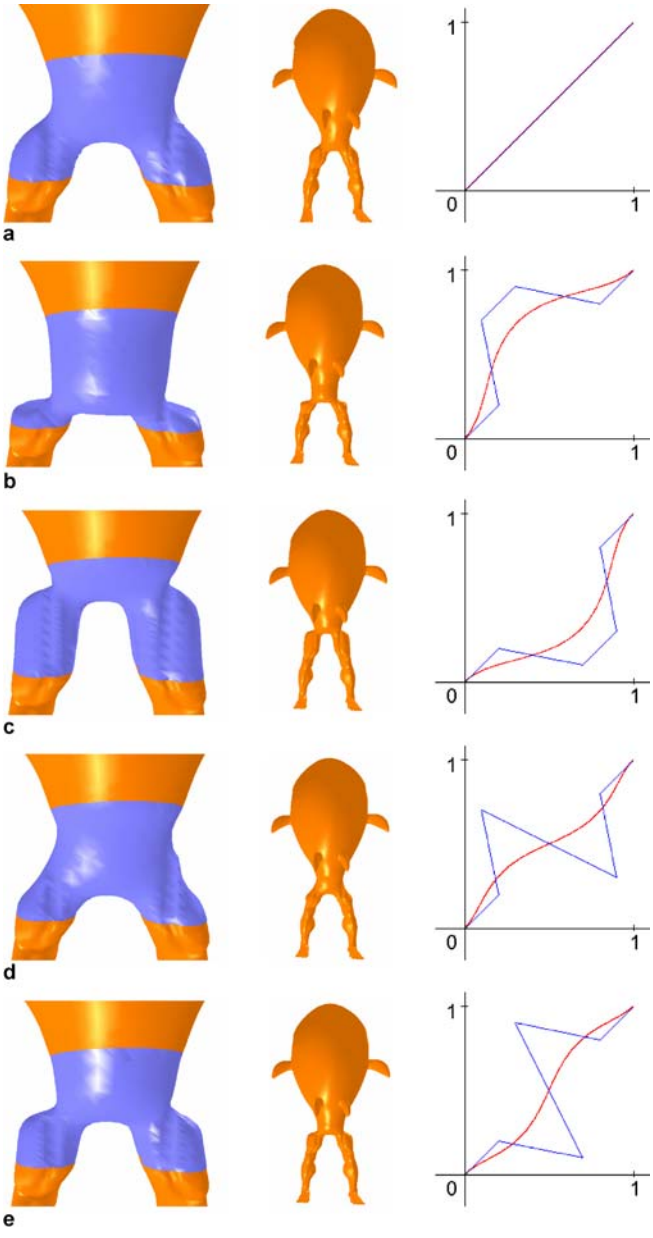


Fig. 4. The blending surface generated using different mapping functions. Left: the resultant blending surfaces. Right: the corresponding mapping functions

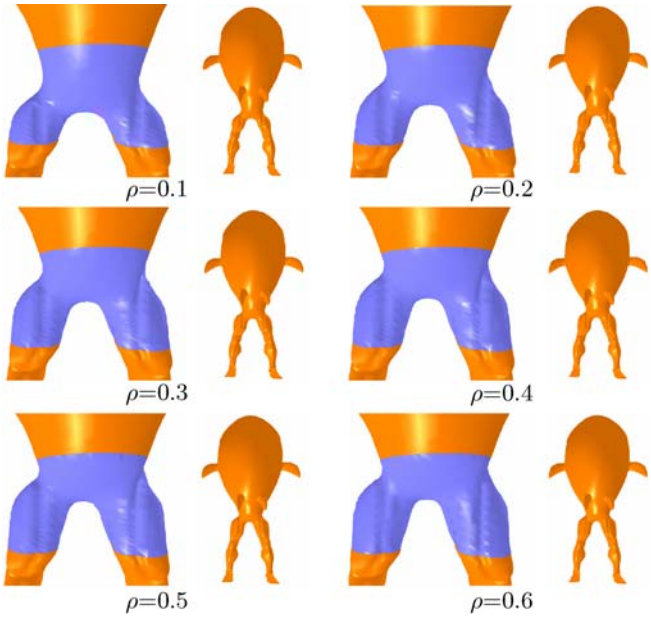


Fig. 5. Blending surface varies with the change of RBF parameter ρ

set. They can be reformed into

$$\left.\frac{\partial \eta}{\partial z}\right|_{z=0} = 1 \quad \text{and} \quad \left.\frac{\partial \eta}{\partial z}\right|_{z=1} = 1.$$

In our current implementation, a sixth-order Bezier curve C is utilized to represent $w = \eta(z)$. Thus, to satisfy the continuities on $\Psi(w)$, we fix the first and last two control points of C on the line $w = z$ (e.g., see the mapping functions in Fig. 4). For the property of a Bezier curve, please refer to [13]. The remaining two control points are adopted to change the mapping function's curve so that the shape of $\Psi(w)$ is adjusted. In Fig. 4, five different mapping functions $w = \eta(z)$ are applied to fuse the same openings. Similarly to the speed control curve in computer animation, different mapping functions lead to different resultant transition surfaces. For example, by adopting the shape control curve shown in Fig. 4(d), the blending surface transits quickly at the boundaries but slowly in the middle.

We can also adjust the shape of blending surfaces by choosing different RBF parameters. When determining the interpolation functions, the function values at offset points are requested. This is defined as an RBF parameter ρ, which can be used to shift the saddle point of a transition surface forwards or backwards. For example, see Fig. 5, where given the same openings, the surface's shape varies with the change of RBF parameters.

3.4 Surface tessellation

The only remaining issue for implementation is how to tessellate the functional surface defined by $\Psi(w)$ and connect it with the parts of interest on given objects. The surface $\Psi(w)$ is defined by $w : 0 \to 1$, directly applying polygonization on the full range of w cannot preserve topology consistency at the openings. Therefore, we give the following modification to prevent cracks. Firstly, the surface $\Psi(w)$ in the range $w \in [\varepsilon, 1-\varepsilon]$ is tessellated by the polygonizer of [8], where ε is a very small positive constant (e.g., $\varepsilon = 0.01$). Polygonal boundary curves B_1 and B_2 created at the plane $w = \varepsilon$ and $w = 1-\varepsilon$ by the polygonizer must have similar a shape and topology to Γ_1 and Γ_2. Then, vertex correspondences are established with a greedy algorithm that iteratively increments the current vertex on either B_i or Γ_i by choosing the one that gives the shortest Euclidean distance. This is similar to the methods used in [14]. After that, linking triangles are constructed according to the corresponding vertices, where every linking triangle should only have one edge on either B_i or Γ_i. Smooth seams are achieved using the above procedure without any postprocessing.

4 Results

We implemented the above mesh fusion approach on a standard PC with Intel Pentium IV 2.4 GHz CPU and 512 MB RAM running Windows XP. The results shown below are all generated on our prototype system.

As described earlier, both the signed distance-field (SDF) and the radial basis function (RBF) can be adopted

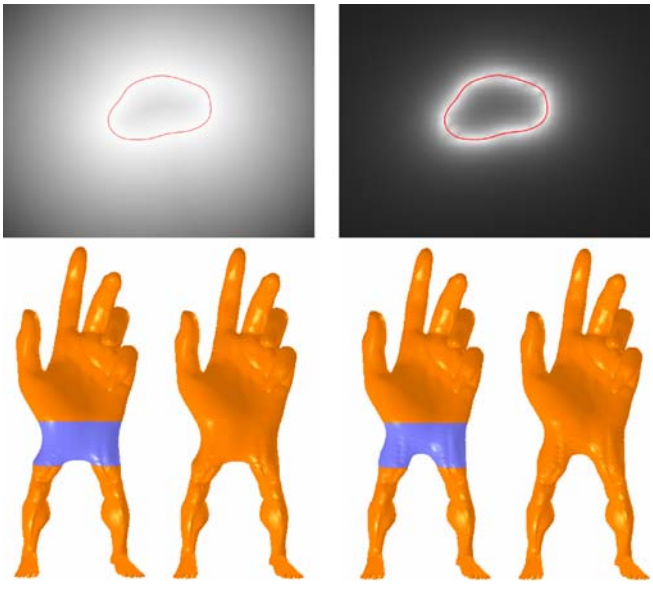


Fig. 6. Mesh fusion results using SDF vs. RBF. Left: signed distance-field represented in a grey image and the surface generated by using SDF. Right: RBF illustrated by a grey image and its corresponding resultant model

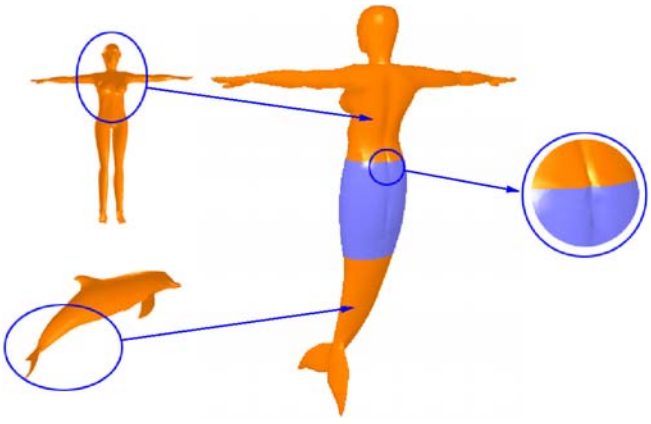


Fig. 7. The details of combining objects parts are preserved in the transition surface

to give the implicit definition of openings. Both schemes provide nice results as shown in Fig. 6. The related mesh fusion results of the "palm-man" example, given in Fig. 1, are both smooth and natural. Although we used the RBF scheme to generate most of our examples, the SDF scheme is conducted in the scenario that meshes to be fused have very different resolutions. For example, when fusing a simple cube model (which has only six faces and twelve triangles) with a complex Buddha model, we may need to remesh the cube model in the RBF scheme to better approximate its opening with RBFs. However, the SDF scheme can be directly applied without remeshing.

As we indicated in Sect. 1, our approach can preserve the surface details of combining parts in the transition surface as shown in Fig. 7.

We have experimented with 3D RBF as the representation of the transition surface. We define position and normal constraints on each boundary vertex following the method in [25]. It was found that the transition surface tends to shrink, as shown in Fig. 8. This is because using

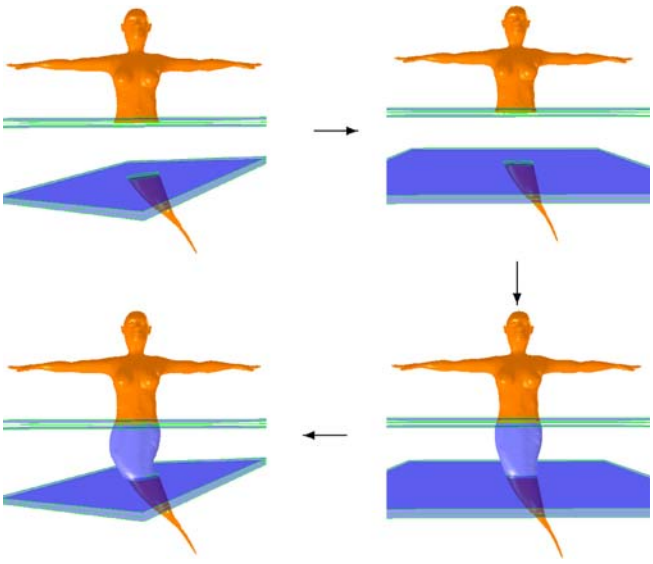


Fig. 9. Mesh fusion with non-parallel sections

the constraints defined on the openings is not enough to generate a natural transition RBF surface (especially when they are far from each other). We believe that more constraints should be added to make it feasible and we will explore this in the near future.

Our approach can be utilized to implement the 3D surface cut-and-paste operation originally introduced in [7]. For example, a bear head is pasted onto a goblet in Fig. 10. Also, the one-ring based surface sewing process in [14] can be well finished by our mesh fusion scheme (see Fig. 11 where a mermaid is constructed from a dolphin and a female model).

The examples in Fig. 10 are classified into the category of 1-1 fusion. In fact, our mesh fusion scheme can do more than that – not only 1-1 fusion but also an *n*-*m* fusion surface can be easily modeled. See Fig. 12 where two heads of a dino-pet are cut and fused onto its neck.

Fig. 8. Fusion using 3D RBF directly

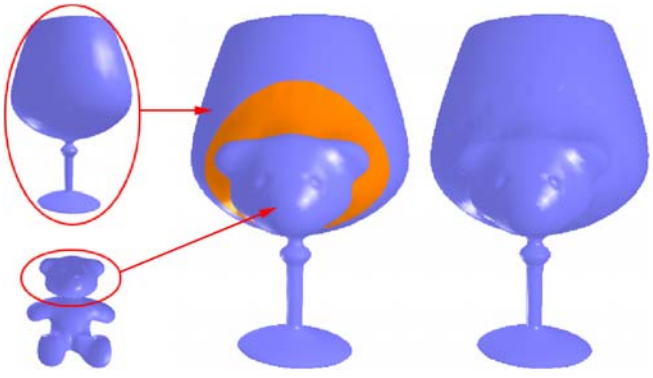


Fig. 10. Pasting a bear head onto a goblet (1-1 fusion)

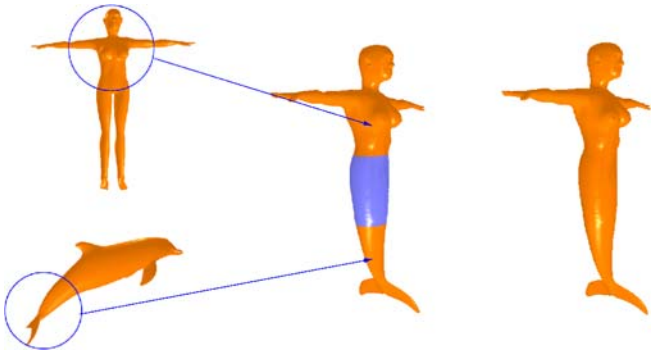


Fig. 11. A mermaid is created from a dolphin and a female model (1-1 fusion)

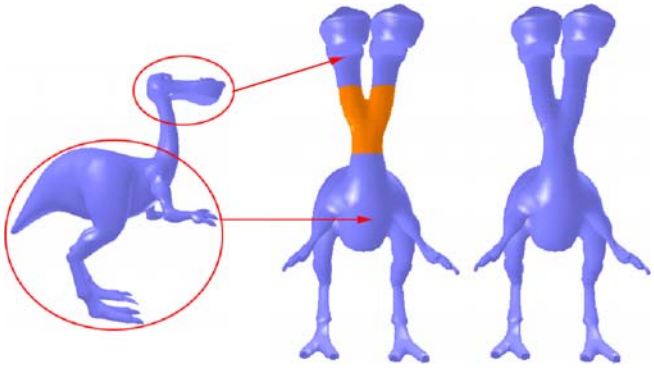


Fig. 12. Merging two heads onto the neck of a dino-pet (1-2 fusion)

Another example is given in Fig. 13, where a two-tailed mermaid is fused from a woman and two dolphins. These two examples are both 1-2 fusion in that one object has a one-ring opening while the other model has a two-ring opening.

The last three examples are from real applications. In Fig. 14, a monster is created from a ferret, an alien and a dinosaur. Figure 15 demonstrates how to construct a moon-boat from a moon, two wine bottles, a goblet and

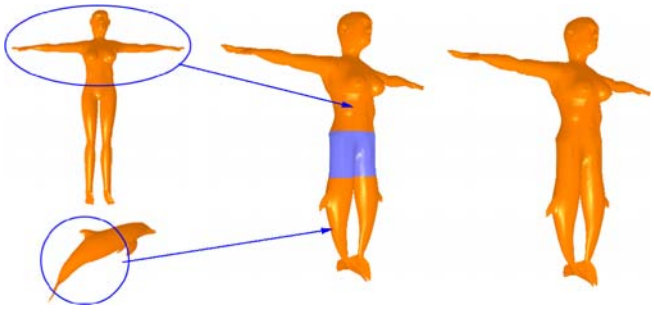


Fig. 13. A two-tailed mermaid is created from two dolphins and a female model (1-2 fusion)

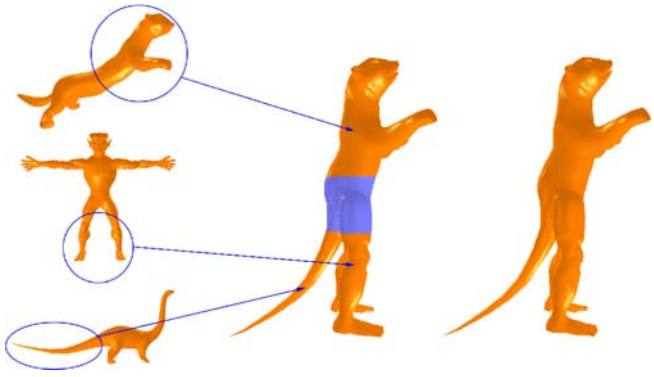


Fig. 14. A monster is constructed using a ferret, an alien and a dinosaur (1-3 fusion)

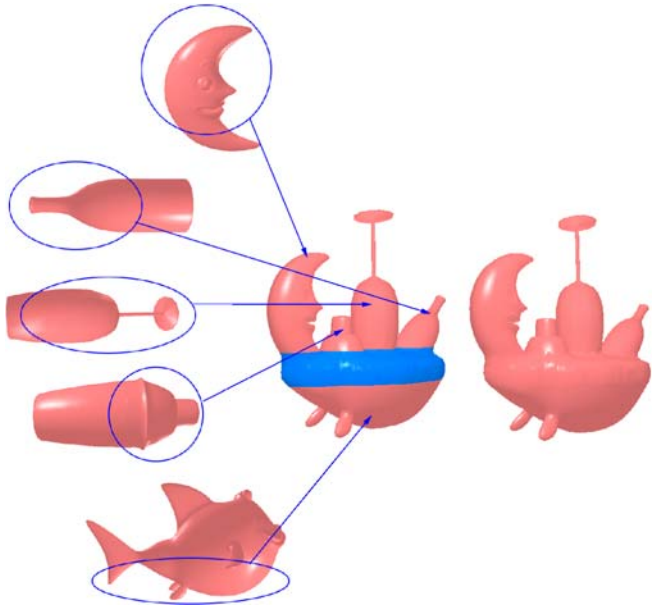


Fig. 15. A moon-boat is created from the parts of a moon, two wine bottles, a goblet and a fish (1-4 fusion)

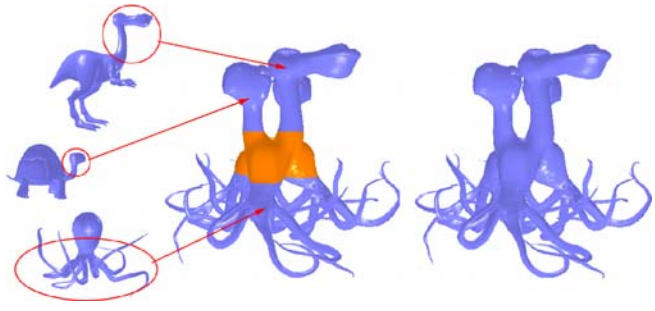


Fig. 16. A three-headed monster fused from a dino-pet, two turtles and three octopi (3-3 fusion)

Table 1. Timing for cut operation and RBF fitting process

Number of opening vertices	Cut time	RBF fitting time
30	0.008 s	0.002 s
61	0.016 s	0.009 s
102	0.030 s	0.048 s
156	0.046 s	0.101 s
213	0.057 s	0.170 s

Table 2. Surface tessellation timing

Figure	Number of transition surface point	Surface tessellation time (RBF)	Surface tessellation time (SDF)
Fig. 8	5168	–	17.779 s
Fig. 9	2110	6.980 s	–
Fig. 10	1312	6.200 s	–
Fig. 11	1520	–	5.465 s
Fig. 12	2930	20.100 s	–
Fig. 13	4194	–	29.201 s
Fig. 14	2930	–	12.864 s

a fish. Figure 16 is a 3-3 fusion example blended from a dino-pet, two turtles and three octopi. The mesh fusion operation in all examples can be finished on our prototype system in half a minute. In other words, the mesh fusion can be used as an interactive tool.

Generally, our mesh fusion scheme can easily generate various expected results. However, fusion with non-parallel sections is expected sometimes. The method presented in this paper can be easily extended to satisfy this by defining a mapping function converting non-parallel sections into parallel ones, so that the above fusion scheme can still be applied to generate the mesh for fusion. After that, the resultant mesh is converted back to the previous non-parallel case using smooth deformation techniques. More specifically, taking the model shown in Fig. 9 as an example, we firstly convert the non-parallel sections into parallel ones by applying a rigid transformation on the tail part. After fusion, we use the Laplacian mesh editing technique [23] to transform the transition mesh surface back into the original non-parallel scenario. Thus, the final fused model is obtained; the whole procedure is illustrated in Fig. 9.

Table 1 lists the data statistics and timing for the cut operation and RBF fitting process. Table 2 quantifies the surface tessellation times for the examples presented in this paper.

5 Conclusion and discussion

In this paper, we proposed a novel functional blending-based mesh fusion scheme that provides a fast and easy way to create new models from existing ones. Different from other approaches, our method has no restriction of disk-like topology or one-ring openings on the meshes under composition. The surface details are also preserved well in the transition surface. Smooth and natural results can be generated easily and quickly with some intuitive shape control means.

There are several avenues for future work. Firstly, when the merging objects have different resolutions, a gradual transition of triangle resolution between low resolution objects and high resolution objects is expected. Moreover, in order to preserve detail, a higher sampling rate in high curvature areas is needed. We plan to consider an adaptive particle system to sample the model, and then reconstruct the transition surface from the sampling particles.

Another possible area of future research is similar to the work presented in [12]. For our current Hermite blending, the blendings of positions and tangents are integrated. If these two blendings are separated, a more accurate cross-section shape control can be expected.

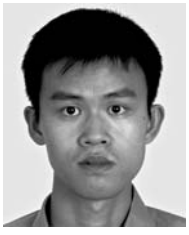
Finally, we will explore the feasibility of using our functional blending scheme fields where the blending of two sections that are topologically different is encountered. This extension makes sense, because it is a problem of importance in computer-aided design. We will also investigate our functional blending scheme in cases with nested openings.

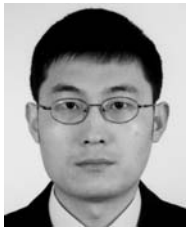
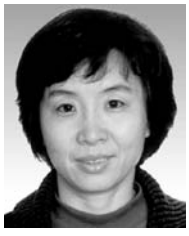
Acknowledgement This work was supported by the 973 program (grant no. 2002CB312101), the National Natural Science Foundation of China (grant no. 60573153, 60533080), the Fok Ying Tung Education Foundation (Grant No. 91069), the Natural Science Foundation of Zhejiang Province (grant no. R105431) and the Program for New Century Excellent Talents at University (Grant No. NCET-05-519). Prof. Hanqiu Sun is supported by RGC Research Grant (No. 4181/03E).

References

1. Adams, B., Dutré, P.: Interactive boolean operations on surfel-bounded solids. ACM Trans. on Graph. **22**(3), 651–656 (2003)
2. Bajaj, C.L., Coyle, E.J., Lin, K.N.: Arbitrary topology shape reconstruction from planar cross sections. Graph. Models Image Process. **58**(6), 524–543 (1996)
3. Barequet, G., Sharir, M.: Piecewise-linear interpolation between polygonal slices. Comput. Vis. Image Understanding **63**(2), 251–272 (1994)
4. Barequet, G., Shapiro, D., Tal, A.: Multilevel sensitive reconstruction of polyhedral surfaces from parallel slices. Visual Comput. **16**(2), 116–133 (2000)
5. Bedi, S.: Surface design using functional blending. Comput. Aided Des. **24**(9), 505–511 (1992)
6. Bellman, R.E.: Introduction to Matrix Analysis. McGraw-Hill, New York (1960)

7. Biermann, H., Martin, I., Bernardini, F., Zorin, D.: A cut-and-paste editing of multiresolution surfaces. ACM Trans. on Graph. **21**(3), 330–338 (2002)
8. Bloothmental, J.:An implicit surface polygonizer. Graphics Gems **IV**, 324–349 (1994)
9. Carr, J.C., Beaston, R.K., Cherrie, J.B., Mitchell, T.J., Fright, W.R., McCallum, B.C., Evans, T.R.: Reconstruction and representation of 3d objects with radial basis functions. In: Proc. of SIGGRAPH'2001, pp. 67–76 (2001)
10. Cong, G., Parvin, B.: Robust and efficient surface reconstruction from contours. Visual Comput. **17**(4), 199–208 (2001)
11. Conte, S., de Boor, C.: Elementary Numerical Analysis: An Algorithm Approach. McGraw-Hill, New York (1980)
12. Elber, G.: Generalized filleting and blending operations toward functional and decorative applications. Graphical Models **67**(3), 189–203 (2005)
13. Farin, G.: Curves and Surfaces for CAGD. Morgan Kaufmann, San Francisco (2002)
14. Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal A., Rusinkiewicz, S., Dobkin, D.: Modeling by example. ACM Trans. on Graph. **23**(3), 652–663 (2004)
15. Fu, H., Tai, C.L., Zhang, H.: Topology-free cut-and-paste editing over meshes. Geometric Modeling and Processing'2004, IEEE Computer Society Press, pp. 173–184 (2004)
16. Kanai T., Suzuki, H., Kimura, F.: Three dimensional geometric metamorphosis based on harmonic maps. Visual Comput. **14**(4), 166–176 (1998)
17. Kanai, T., Suzuki, H., Mitani, J., Kimura, F.: Interactive mesh fusion based on local 3D metamorphosis. In: Proc. of Graphics Interface'99, pp. 148–156 (1999)
18. Lorensen, W., Cline, H.: Marching cubes: A high resolution 3D surface construction algorithm. Comput. Graph. **21**(4), 163–169 (1987)
19. Museth, K., Breen, D., Whitaker, R., Barr, A.: Level set surface editing operators. ACM Trans. on Graph. **21**(3), 330–338 (2002)
20. Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., Seidel, H.: Multi-level partition of unity implicits. ACM Trans. on Graph. **22**(3), 463–470 (2003)
21. Pauly, M., Keiser, R., Kobbelt, K.P., Gross, M.: Shape modeling with point-sampled geometry. ACM Trans. on Graph. **22**(3), 641–650 (2003)
22. Singh, K., Parent, R.: Joining polyhedral objects using implicitly defined surfaces. Visual Comput. **17**(7), 415–428 (2001)
23. Sorkine, O., Lipman, Y., Cohen-Or, D., Alexa, M., Rössl, C., Seidel, H.P.: Laplacian surface editing. In: Proc. of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, Eurographics Association, pp. 179–188 (2004)
24. Turk, G., O'Brien, J.: Shape transformation using variational implicit functions. In: Proc. of SIGGRAPH'1999, pp. 335–342 (1999)
25. Turk, G., O'Brien, J.: Modelling with implicit surfaces that interpolate. ACM Trans. on Graph. **21**(4), 855–873 (2002)
26. Yu, Y., Zhou, K., Xu, D., Shi, X., Bao, H., Guo, B., Shum, H.: Mesh editing with Poisson-based gradient field manipulation. ACM Trans. on Graph. **23**(3), 664–651 (2004)

XIAOGANG JIN is a professor of the State Key Lab of CAD & CG, Zhejiang University, People's Republic of China. He received his BSc degree in Computer Science in 1989, and MSc and PhD degrees in Applied Mathematics in 1992 and 1995, all from Zhejiang University. His research interests include implicit surface modeling, space deformation, computer animation and realistic image synthesis.

JUNCONG LIN is a PhD candidate of the State Key Lab of CAD & CG, Zhejiang University. He received his BSc degree in Environmental Engineering in 2003 from Zhejiang University. His research interests include mesh editing and modeling.

CHARLIE C. L. WANG is currently an assistant professor at the Department of Automation and Computer-Aided Engineering, the Chinese University of Hong Kong. He gained his BEng (1998) in Mechatronics Engineering from the Huazhong University of Science and Technology, and his MPhil (2000) and PhD (2002) in Mechanical Engineering from the Hong Kong University of Science and Technology. He received the best paper award of 2001 ASME DETC/CIE conference. He is a member of ASME and IEEE. His current research interests include design automation and optimization, geometric modeling, CAD/CAM, reverse engineering, and computer graphics.

JIEQING FENG is a professor at the State Key Lab of CAD & CG, Zhejiang University, People's Republic of China. He received his BSc degree in Applied Mathematics from the National University of Defense Technology in 1992 and his PhD in Computer Graphics from Zhejiang University in 1997. His research interests include space deformation, computer-aided geometric design and computer animation.

DR. HANQIU SUN received her BSc in Electrical Engineering from Huazhong University of Science and Technology, China. She received her MSc in Electrical Engineering from the University of British Columbia and her PhD in Computer Science from the University of Alberta, Canada. She then worked at the University of Alberta and the University of Winnipeg as a lecturer and later as an assistant professor until 1996. Dr. Sun then joined the Computer Science and Engineering Department of CUHK. Dr. Sun has published more than 100 refereed technical papers. Her current research interests include interactive animations, virtual & augmented reality, hypermedia, and realistic haptics simulation.