Int j simul model 16 (2017) 3, 527-540
ISSN 1726-4529 Original scientific paper

EFFICIENT SMOOTHNESS-PRESERVING FUSION MODELLING METHOD FOR MESH MODELS

Qin, X. J.[*]; Duan, Z. J.[**]; Zheng, H. B.[*] & Tang, Y.[*,#]

[*] College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou,
P. R. China

[**] College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua,
P. R. China

E-Mail: qxj@zjut.edu.cn, dzjzjnu@zjnu.edu.cn, zhb@zjut.edu.cn, ytang@zjut.edu.cn
([#] Corresponding author)

Abstract

The essential process of preserving the smoothness of stitched meshes in mesh model fusion is challenging. This paper proposes an efficient mesh model fusion method to preserve the smoothness of the stitched surface along two stitching meshes. We first cut parts of the stitched meshes from source mesh models and then extract their boundary contours. Subsequently, the boundary contours are aligned manually and projected onto parallel planes for contour vertex triangulation. Triangular meshes obtained from the stitched meshes by loop subdivision are interpolated using a compactly supported radial basis function to smooth the stitched surfaces. The experimental results on public graph models demonstrate the effectiveness and efficiency of our method in mesh model fusion and smoothness.

(Received, processed and accepted by the Chinese Representative Office.)

Key Words: Modelling, Mesh Fusion, Mesh Smoothing, Triangulation, Interpolation

1. INTRODUCTION AND RELATED WORKS

The recently increasing demand for three-dimensional (3D) geometric models in entertainment, such as films, games, and animation, as well as in military and manufacturing, has led to the rapid development of fast modelling and processing technology for 3D models. The number of methods for obtaining 3D models (e.g., 3D laser scanning [1], industrial CT) is increasing. Furthermore, 3D model libraries have become very large, and model resources have become extremely rich. The use of existing models by creative workers to quickly build new models with more efficient editing operations has become an important direction of research on current 3D model-editing techniques [2].

Mesh fusion with existing models is used by designers to efficiently create thousands of new models and has been researched extensively. Pedersen proposed a method of cutting and pasting implicit surfaces [3], but the method requires the model to be represented by an implicit surface rather than an explicit mesh. Museth et al. [4] presented a level set method based on a surface-editing framework. However, their method is time consuming, and its transformation results in a loss of surface features. Biermann et al. [5] chose semi-regular multi-resolution subdivision surfaces as their surface-pasting representation. However, their method is highly dependent on the complexity of the target and feature regions and is thus not suitable for complex objects. Kanai et al. presented a global harmonic mapping scheme based on a 3D metamorphosis for mesh fusion [6], but as the number of points in the model increases, the amount of computation becomes large. Singh and Parent [7] provided a procedural implicit function defined for the region of a polyhedral object that is star-shaped with respect to a skeletal point. This function is used to construct the transition surfaces that join the polyhedral objects. Adams and Dutre [8] and Pauly et al. [9] proposed the use of Boolean operations on mesh models through sampling to achieve the stitched models from

https://doi.org/10.2507/IJSIMM16(3)CO14

beaked models, for which it is difficult to ensure seamlessness and smoothness in the fusion region.

Bernstein and Wojtan [10] proposed a method for non-closed, non-manifold meshes to handle topological changes. Sorkine et al. [11] proposed a method of mesh editing, but the method is limited by topology. Lin et al. [12] proposed a sketch-based mesh fusion method that adopts a radial basis function (RBF)-based implicit surface to fuse the parts smoothly, but the method lacks the complex detailed features of the model. Jin et al. [13] proposed a mesh fusion algorithm based on an RBF-fitting model; however, their method requires the boundary contours to be parallel.

Previous mesh fusion methods have two shortcomings. First, many methods are limited by the topological structures of their models. For instance, the global harmonic mapping used in Kanai's [6] algorithm requires the topological structure of the source model to be isomorphic to a disk, and Biermann's [5] algorithm requires the target region to be isomorphic to a disk. Second, the algorithm efficiency of many methods is low. For instance, the complexity of Kanai's algorithm is $O(n^2)$, where n is the number of vertices of the fusion region; therefore, its computation is time consuming when the selected region is complex. Museth et al. [4] proposed a time-consuming surface-editing method in which the model is represented as a volume. Recently, Lou et al. proposed a new approach to merge finite-element triangular meshes [14], but the method aims to merge only computer-aided design models. Lai and Yuen presented a blending scheme to blend a triangular mesh and a non-uniform rational basis spline surface [15]. The method provides a simple and effective strategy for merging a mesh and a parametric surface. Centin et al. proposed a Poisson-driven seamless completion of triangular meshes [16].

Aiming at fast modelling with existing models, we present a novel smoothness-preserving fusion method for mesh models. The major contributions of this paper are as follows:

(1) A novel fusion and smooth blending algorithm for discrete mesh surfaces is presented.

(2) The 3D boundary contour triangulation problem is transformed into planar contour triangulation. First, we cut the input mesh models with a cuboid or a sphere to obtain the parts for fusion. Then, we extract the boundary contours of the mesh models to be stitched and project the two boundary contours onto two parallel planes. We then build meshes between the two projected contours. According to the correspondence between the plane and mesh boundary vertices, we can build the stitched mesh surface of the two mesh models. Then, we subdivide the stitched mesh and optimize the subdivision mesh by using compactly supported radial basis function (CSRBF) interpolation. This process results in a smooth mesh surface.

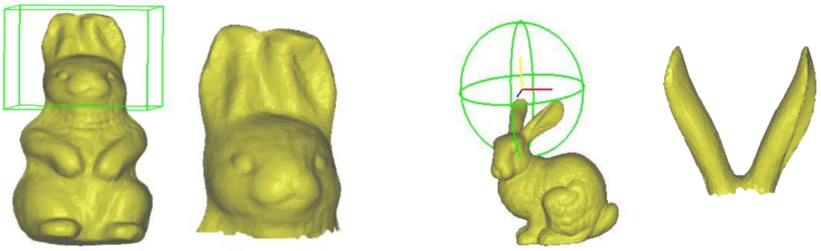
(3) An accelerated algorithm of surface interpolation based on region control is proposed.

Generally, our mesh fusion method produces a natural transition of two mesh models in a blending region while avoiding surface folding issues. Moreover, the speed of reconstructing a transition surface is greatly improved by our interpolation-accelerating algorithm. Our rapid modelling method can be applied to modelling in various fields, including animation games, finite element analysis models [17], and medical surgery simulations [18].

The remainder of this paper is organized as follows: Model segmentation and registration are described in Section 2, and the triangulation of mesh boundary contours is introduced in Section 3. Then, multi-scaled CSRBF-based optimization of the vertices in a blending region is presented in Section 4. After analysing the experimental results in Section 5, we conclude in Section 6.

2. MODEL SEGMENTATION AND REGISTRATION

To create mesh fusion models, we must first cut the required parts from the source mesh models and then align and register the mesh models. Finally, the blending regions of the mesh models must be fused and smoothed. In our method, the mesh boundaries do not need to be cut precisely or smoothly; therefore, simple 3D geometric models, such as planes, cuboids, and spheres, can be used to cut the mesh models. During cutting, only the vertices of the mesh models and the triangulated surfaces must be internally and externally tested. In this manner, we can easily and quickly cut out the required part from the existing model. Fig. 1 illustrates the use of a cuboid and sphere to cut mesh models.



a) Model cut using a cuboid

b) Model cut using a sphere

Figure 1: Cut 3D mesh models.

3. TRIANGULATION OF MESH BOUNDARY CONTOURS

After interactive aligning and registering, the cut models are stitched and triangulated along the boundary contours of the two models. We first extract the boundary contour lines between the models to be fused and then further triangulate the boundary contours to obtain the final stitched model.

3.1 Extraction of boundary vertices and boundary contours

Extraction of the boundary contours of the cut 3D mesh models is the key to and the basic task of contour triangulation. Boundary contour extraction is the extraction of an ordered boundary vertex set from the meshes. Since the distribution of the spatial mesh vertices is irregular, it is difficult to apply the extraction of 2D image boundary contours to a 3D mesh model. Hence, we present a boundary vertex extraction method based on a 3D mesh model.

We have established adjacency relations among the vertices, edges, and triangulated surfaces of 3D mesh model in our 3D mesh model fusion system. The key components of the data structure of a triangular mesh model, such as the vertices, edges, and surfaces, read by the system are as follows:

```
VERTEX{ID; Position[3]; Normal[3];
        Neighbor_Triangle_Facet-list;
        Neighbor_Vertex-list};
EDGE{ID; VERTEX [2];
      TRIANGLE_FACET [2]};
TRIANGLE_FACET {ID; VERTEX [3];
                 Neighbor_Triangle_Facet[3];
                 Facet_ Normal[3]};
```

The data structures of our mesh fusion system include the following three lists, which are recorded by the vertices, edges, and facets: VERTEX-list, EDGE-list and TRIANGLE_FACET-list.

An edge is considered to be the boundary line if only one adjacent TRIANGLE_FACET is in its EDGE structure components.

We traverse the mesh model’s EDGE-list and determine all boundary edges to establish a boundary edge list; however, the edges in the boundary edge list are unordered. Before triangulating the boundary contours of the fusion mesh models, we must sort these boundary edges using the following process:

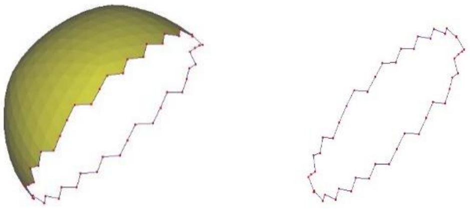
(1) Construct a temporary edge list. Take the first edge from the boundary edge list and add it to the temporary edge list; this is the current edge of the temporary edge list. Remove this edge from the boundary edge list. The boundary contour direction is from the begin-vertex VERTEX[0] to the end-vertex VERTEX[1] of the first edge.

(2) Take the current edge from the temporary edge list. Search for an edge in the boundary edge list for which the IDs of the vertices VERTEX[0] and VERTEX[1] are equal to the current end-vertex VERTEX[1]. Then, add the edge to the temporary edge list. If the vertex ID of the end-vertex VERTEX[1] of the edge is equal to the ID of the end-vertex VERTEX[1] of the current edge, which belongs to the temporary edge linked list, then swap the begin-vertex VERTEX[0] and the end-vertex VERTEX[1] of the edge. The new edge is added to the temporary edge list as the current edge.

Repeat (2) until the boundary edge list is empty.

Update the boundary edge linked list using the temporary edge list and delete the temporary edge list.

The edges of the new boundary edge list are then in order. Fig. 2 shows the mesh model and its boundary contour extracted using the method described above.



a) 3D mesh model b) Boundary contour edge

Figure 2: Contour edge of a 3D mesh model.

3.2 Plane projection of boundary contours

We obtain the ordered contour edges of the fusion mesh models through the above process. The irregular distribution of the boundary vertices in 3D space makes it difficult to triangulate the upper and lower contours directly. We note that the plane contour-based triangulation algorithm is relatively mature. The ordered boundary contour of the mesh surfaces is further projected onto two parallel planes; then, we triangulate the projected contours using the triangulation algorithm of a planar contour. Finally, because there is a mapping relation between the projected planar contour and the vertices of the 3D boundary contour, we can triangulate the projected contour and map it directly onto the 3D boundary contour, consequently establishing the triangulation meshes between two 3D boundary contours.

For the two boundary contours to be fused, set the boundary vertex sets of the upper and lower boundary contours as *P* and *Q*, respectively. Set the number of vertices *m* and *n* in vertex sets *P* and *Q*, respectively, and let $p \in P$ and $q \in Q$. Then, the centres of gravity of the respective contours are the points $\overline{p} = \frac{\sum p}{m}$ and $\overline{q} = \frac{\sum q}{n}$. We define a vector as $\vec{n} = \overline{p} - \overline{q}$. The two parallel projection planes are formed by $\vec{n}$ and $\overline{p}$ and by $\vec{n}$ and $\overline{q}$, where vector $\vec{n}$ is the projection direction.

In Fig. 3, the vertices P_2' and P_3', which are projected onto the plane, might appear to overlap. If the vector $p_3 - p_2$ and the projection facing the normal direction $\vec{n}$ are parallel, then the vertices P_2 and P_3 overlap when projected onto the plane. If they overlap, then vertex P_2' is moved to a new position $p_2' = \theta p_1' + (1 - \theta) p_3'$. As in Fig. 3, p_2 is the vertex of the overlapping projection. p_1 is a projection vertex adjacent to the left of p_2 but not overlapped. Here, $\theta \in (0, 1)$. To solve the problem of overlap, we move the overlapped projection vertex a short distance to the nearby but not overlapped projection vertex.

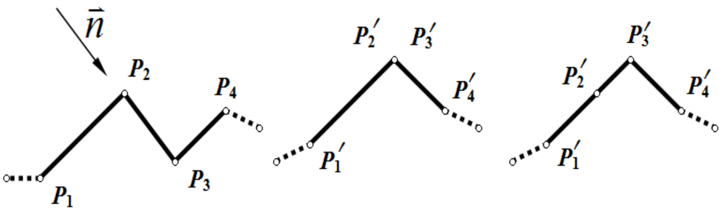


Figure 3: Process of overlapped boundary vertices; 3D boundary vertices (left), overlapped vertices of projection (middle), processed overlapped vertices (right).

3.3 Triangulation of the boundary contour

After projecting the mesh boundary contours onto the two parallel projection planes, we can triangulate the planar contours. Keppel [19] proposed an algorithm for planar contour triangulation that takes the largest volume as the triangulation standard for the upper and lower plane contour lines. However, the algorithm is difficult to apply in practice. Fuchs et al. [20] presented the minimum area, which is easy to calculate and has been widely used, as a standard of surface reconstruction. Christiansen and Sederberg [21] designed a greedy algorithm based on the minimum span length. Ganapathy and Dennehy [22] presented a heuristic triangulation method for planar contours. Based on the ideas of Christiansen's and Ganapathy's algorithms, we have designed a new heuristic algorithm that provides a quick and reasonable solution. However, it might not be the optimal solution since the optimal the plane might not be the optimal space for a boundary contour. The advantage of this method is that it can resolve the inconsistencies in the numbers of upper and lower boundary vertices, thereby satisfying our requirements. A brief description of the heuristic triangulation algorithm of the plane contours is given as follows.

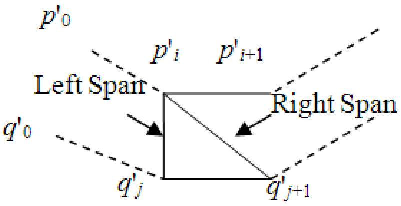


Figure 4: Heuristic triangulation algorithm.

Set the upper boundary contour's boundary vertex set as P and its projection vertex set as P'. Set the lower boundary contour's boundary vertex set as Q and its projection vertex set as Q'. Set the number of vertices of P and Q as m and n, respectively. When projecting a plane, the segment that connects each vertex of the upper and lower contours, as shown in Fig. 4, is called a span.

First, normalize the lengths of the upper and lower projection contours and make the sum of all segment lengths of each contour equal to one. Then, take the upper projection contour as an example.

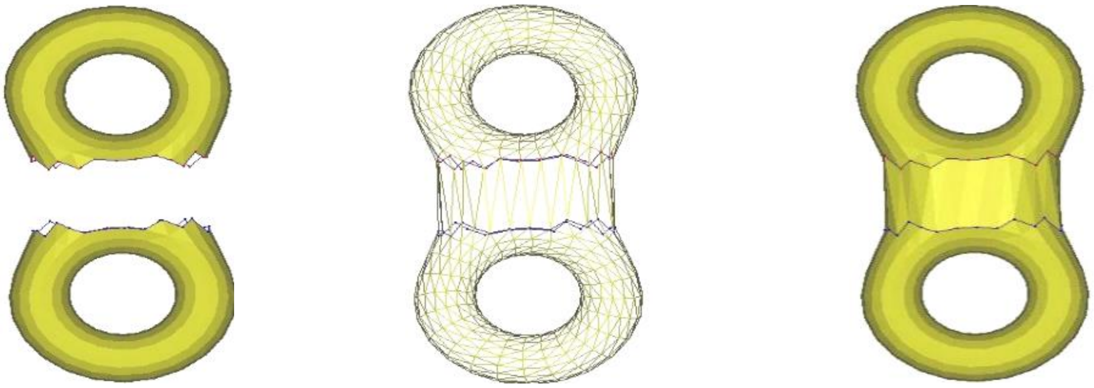
Let us set $w(p'_i, p'_{i+k}) = \frac{|p'_i p'_{i+k}|}{L}$, $w(p'_i, p'_{i+k}) = \frac{|p'_i p'_{i+k}|}{L}$, and $L = \sum_{i=0}^{m-2} |p'_i p'_{i+1}| + |p'_{m-1} p'_0|$.

Then, we use the following rules to determine the connection between the triangulated surfaces. The projection boundary edge and the mesh model of the original boundary vertex have one-to-one correspondence. If the correspondence satisfies the conditions:

$$\left|w(p'_0, p'_i) + w(p'_i, p'_{i+1}) - w(q'_0, q'_j)\right| < \left|w(q'_0, q'_j) + w(q'_j, q'_{j+1}) - w(p'_0, p'_i)\right| \qquad (1)$$

then choose $p_i\, p_{i+1}\, q_j$ as a new triangulated surface. Otherwise, choose $p_i\, q_j\, q_{j+1}$ as a new triangulated surface. The rule requires the added triangulated surface to satisfy the smallest difference in the upper and lower contours, as shown in Eq. (1).

After obtaining the triangulated projection contours, according to the correspondence between the projected vertices and the 3D mesh boundary vertices, we can establish a triangulated blending region of the 3D mesh models. Fig. 5 a shows two mesh models to be fused and their boundary contours. Fig. 5 b shows a fused model using the abovementioned heuristic contour triangulation algorithm. Fig. 5 c shows the rendered fusion model.



a) Models to be fused b) Triangulation of the blending region c) Fusion model with rendering

Figure 5: Boundary contour triangulation of 8-shaped models.

4. MULTI-SCALE CSRBF-BASED OPTIMIZATION OF VERTICES IN THE BLENDING REGION

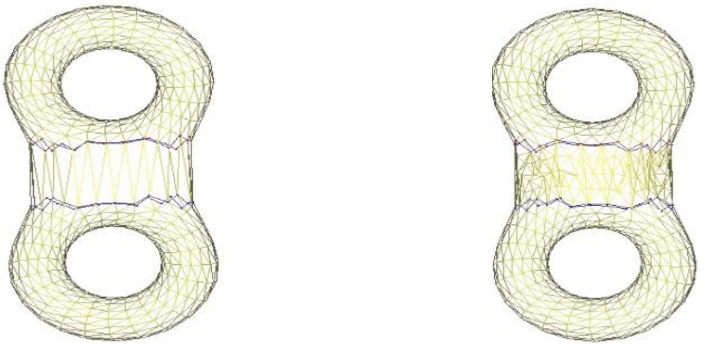
We triangulate the fusion region via the abovementioned method. We extract and triangulate the boundary contours directly to ensure that the region can be seamlessly stitched. However, the model-merging region involves direct connection and lacks a naturally smooth transition. To ensure that the surface of the merging region is a smooth mesh surface, we use implicit function interpolation to establish a smooth transition surface in that region.

The specific process is as follows. First, subdivide the mesh of the blending region of the fusion models into the subdivided meshes. Next, use implicit function interpolation of the CSRBF for all mesh vertices on the original mesh model (excluding the subdivided mesh vertices). This process yields the spatial distance field of the model from CSRBF interpolation. Then, move the subdivided mesh vertices to the zero-valued isosurface under the constraints of the spatial distance field. In this way, we can ensure that the subdivided mesh points approximate the zero-valued isosurface. The smooth transition mesh of the blending region is thereby obtained.

4.1 Mesh subdivision of the blending region

After triangulation of the boundary contour, the generated mesh surface is a directly connected rough surface. To smooth the fusion region, we need to increase the number of mesh vertices to approximate the zero-valued isosurface. We must further subdivide the blending region mesh. Here, we use a simple rule of subdivision through edge-splitting subdivision. First, detect all edges in the fusion region. If the length is longer than a given threshold, then split the edge from the midpoint into two edges. Then, re-triangulate the triangulated mesh surface whose edge has been split. The threshold is determined by the average length of the upper and lower contour edges.

Fig. 6 shows the pre-subdivided and subdivided results, respectively, of the model.



a) Original mesh of the blending region b) Subdivided mesh of the blending region

Figure 6: Mesh subdivision of the fusion region.

4.2 CSRBF fitting of the model mesh vertices

The mesh in the blending region obtained using the abovementioned method of mesh subdivision does not meet the requirements of smooth transition. We must move the subdivided mesh vertices appropriately to obtain a smooth connection of the mesh. Therefore, we must generate an implicit function representation for the mesh vertices.

Generally, the vertex set is converted to an implicit function representation, which is a problem of multi-variant scatter data interpolation. We use variation interpolation technology to obtain the representation $f(p) = 0$ by using an implicit function. Considering that the interpolated surface is smooth, the smoothness of the surface can be evaluated using the thin-plate energy function of computer-aided geometric design.

$$E = \int_{\Omega} [\, f_{xx}^2(x,y) + 2 f_{xy}^2(x,y) + f_{yy}^2(x,y)] \mathrm{d}x \mathrm{d}y \tag{2}$$

In Eq. (2), f_{xx}, f_{xy}, f_{yy} represent the second-order partial derivatives of the function f. Ω is the region covered by the interpolated data vertices. Because the energy E is an estimate of function f on square curvature Ω, when a wrinkle appears on f, E increases. Therefore, minimizing E can render a smooth surface of f.

The above energy function can easily be extended to a high-dimensional case. The general solution of the equation consists of a linear combination of an RBF $\phi(p)$ and a linear term $Q(p)$. The general expression is:

$$f(p) = \sum_{j=1}^{n} \lambda_j \phi(| p - p_j |) + Q(p) \tag{3}$$

where $Q(p)$ is a first-degree polynomial, λ_j is a real-valued weight coefficient, $\phi(\bullet)$ is an RBF, $|\bullet|$ is the Euclidean norm, and $\{p_j\}_{j=1}^{n}$ are the interpolated data points.

To solve for the weights λ_j of Eq. (3), we express the linear term in Eq. (3) as $Q(p) = c_0 x + c_1 y + c_2 z + c_3$. Then, the equation can be converted to solve the following linear system.

$$\begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} & x_1 & y_1 & z_1 & 1 \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2n} & x_2 & y_2 & z_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nn} & x_n & y_n & z_n & 1 \\ x_1 & x_2 & \cdots & x_n & 0 & 0 & 0 & 0 \\ y_1 & y_2 & \cdots & y_n & 0 & 0 & 0 & 0 \\ z_1 & z_2 & \cdots & z_n & 0 & 0 & 0 & 0 \\ 1 & 1 & \cdots & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \\ c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_n \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{4}$$

In Eq. (4), $h_j = f(p_j)$, $\phi_{ij} = \phi(|\, p_i - p_j \,|)$. The equation can be simplified to the following equation:

$$\begin{pmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^{\mathbf{T}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \lambda \\ \mathbf{c} \end{pmatrix} = \begin{pmatrix} \mathbf{h} \\ \mathbf{0} \end{pmatrix}, \tag{5}$$

where $A_{ij} = \phi(|\, p_i - p_j \,|)$, $\mathbf{P} = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & 1 \end{bmatrix}$, $i, j = 1, 2, \cdots, n$.

The coefficient matrix of Eq. (4) is a symmetric matrix. An appropriate RBF ϕ can be selected to transform the coefficient matrix into a positive definite matrix. Thus, a solution to the linear system always exists [23].

In 1995, Wendland constructed a compact, locally supported RBF that guarantees the positive definiteness of the coefficient matrix of the linear system (4) [24]. Here, we use a CSRBF and set $\phi(r) = (1-r)_+^4(4r+1)$, as suggested in [23]. Because the RBF is compactly supported, $\phi(|\, p_i - p_j \,|)=0$, when the distance between P_i and P_j is larger than the supported radius. This CSRBF property can make the linear system's coefficient matrix sparse and bounded, which can reduce the computational complexity of RBF interpolation.

The vertex's value on the CSRBF-interpolated surface is always zero, that is, $f(p_j) = h_j \equiv 0$. Therefore, off-surface vertices are appended to the input data and given non-zero values to avoid the trivial solution that $f(p_j)$ is zero everywhere. These off-surface vertices are generated by projecting along a surface normal.

4.3 Moving the mesh vertices of the blending region and mesh smoothing

We have used the CSRBF method to interpolate the mesh vertices in the original models to obtain the implicit surface. The implicit surface is represented as a 3D distance field. A zero-valued isosurface in the distance field is the smooth transitional surface of the two fusion models. To obtain the smooth transitional fusion surface, we move the new subdivided mesh vertices to the zero-valued isosurface of the interpolation distance field, and obtain the resulting smooth fusion mesh surface.

(1) Moving the mesh vertices of the blending region

Let $f(x) = 0$ be a zero-valued isosurface obtained from the CSRBF interpolation surface. For a given mesh, let X be the vertices in the blending region; s is a moving step, δ is a threshold, and $x \in X$. Let vertex set X have n vertices. Here, vertex $x_i \in X$ and $i = 0, \ldots, n-1$. Then, the centre of gravity of the vertex set X is:

$$\bar{x} = \frac{1}{n}\sum_{i=0}^{n-1} x_i \tag{6}$$

We ensure that the direction from every vertex to the centre of gravity is inward, that is, $\bar{n}_i = \bar{x} - x_i$, and the vertices move inward or outward along that direction. If the surface of the model is complicated, moving along the direction of a vertex normal will cause the surface to become overlapped or staggered, which consequently causes failure of the program iteration. Therefore, we do not directly use the vertex normal as the moving direction. The moving rule of the mesh vertex x is as follows:

If $f(x) < 0$, then the vertex is outside the surface and moves inward with $\frac{\bar{n} \times s}{|\bar{n}|}$. If $f(x) > 0$, then the vertex is inside the surface and moves outward with $-\frac{\bar{n} \times s}{|\bar{n}|}$. Move all vertices X until $|f(x)| < \delta$. In the moving process, if $f_{prev}(x) f_{curr}(x) < 0$, then adjust the step s and set $s = s/3$.

(2) Mesh smoothing

Slight wrinkles may remain on the transition surface of the fusion region after surface interpolation and moving mesh vertices. One method to make the transition surface as smooth as possible is to further subdivide the mesh. We increase the number of mesh vertices to improve the accuracy of the approximation. However, the increase in the number of mesh points is exponential, causing the mesh to be too dense, and the calculation is relatively time consuming. Another method is geometric smoothing, which is simple and fast. Here, we use a geometric smoothing algorithm and loop subdivision [25]. For the moved mesh vertices p of the blending region, its n adjacent vertices are p_0, p_1, …, p_{n-1}, as shown in Fig. 7. Then, p is updated to a new position as:

$$p = (1 - n\beta_n) p + \beta_n \sum_{i=0}^{n-1} p_i \tag{7}$$

where $\beta_n = \frac{1}{n}\left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n}\right)^2\right)$.

The above smooth processing is C^2 continuous on a regular surface and C^1 continuous at a singular point of the regular surface.

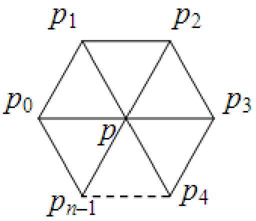
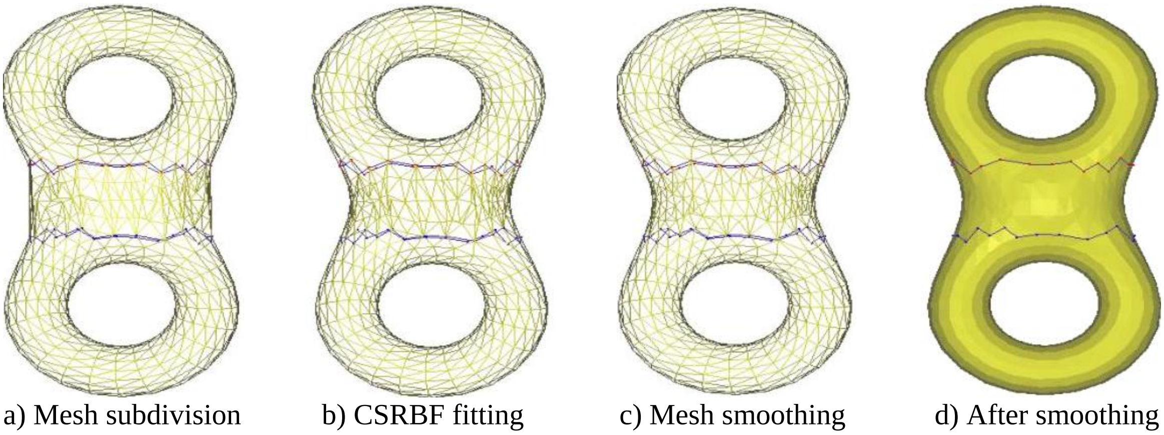


Figure 7: Geometric smoothing operation of mesh vertices.

Fig. 8 shows the result processed by mesh subdivision, zero-valued isosurface approximation, and mesh smoothing of the fusion region. Fig. 8 a shows the result of mesh subdivision in the blending region. Fig. 8 b shows the result of the mesh vertices moving along the zero-valued isosurface. The figure shows that both sides of the blending mesh are contracted, and the transition effect is obvious. Fig. 8 c shows the result of the mesh after smoothing. Fig. 8 d shows the lighting effect of the mesh after smoothing. Compared to Fig. 5 b, the smooth transition effect in the fusion region in Fig. 8 d is obvious.



a) Mesh subdivision b) CSRBF fitting c) Mesh smoothing d) After smoothing

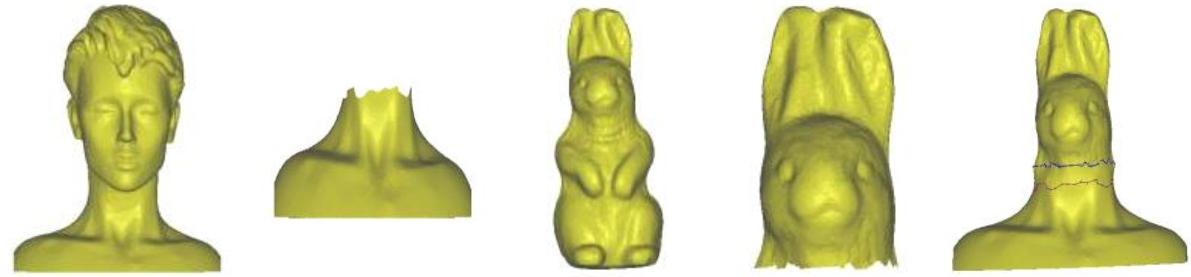
Figure 8: Approximation and smoothing of fusion region.

5. EXPERIMENTAL RESULTS AND ANALYSIS

The experimental programming environment used in this paper is Visual Studio 2005 and DirectX 9.0. The computer configuration is as follows: dual-core CPU, Intel(R) Core (TM2) Duo E7400, 2.0 GB memory, GeForce8400 GS GPU, 512 MB graphics memory.

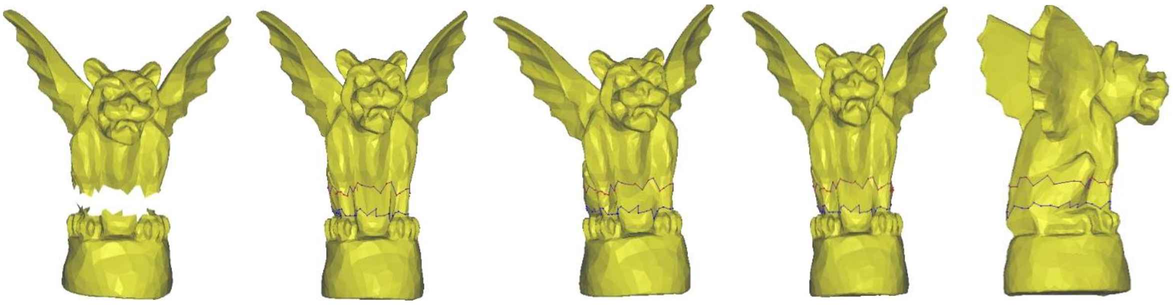
5.1 Fusion and transition between models

Figs. 9 and 10 show the mesh fusion results of our method. Fig. 9 shows that a new model with smooth connections is obtained from the human head and the rabbit model by cutting, aligning, connecting, and fusing. Figs. 9 a and c are the source models, and Figs. 9 b and d are the cut models from the source models. Fig. 9 e is the final fusion mesh model. Fig. 10 is the fracture model with the intermediate mesh being lost. Here, we use our method to repair the fracture model. Fig. 10 b is a direct triangulation mesh connection for the boundary contours. Fig. 10 c shows the subdivision of the connected mesh and the movement of the subdivision vertices with CSRBF interpolation and approximation. Figs. 10 d and e are the results of the repaired region from our mesh fusion method.



a) Head model b) Cut model c) Rabbit model d) Cut model e) Fusion model

Figure 9: Rabbit and head model fusion.



a) b) c) d) e)

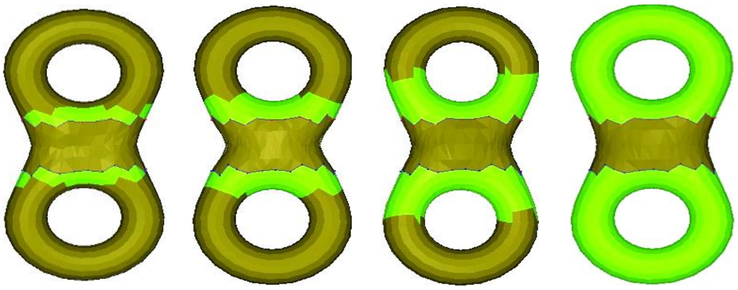
Figure 10: Results of incomplete model repair; a) Fracture model, b) Boundary contour extraction and triangulation, c) Surface subdivision and approximation, d) Mesh smoothing, e) Side view after smoothing.

5.2 Accelerate computation of the fusion process

As some models are complex with dense meshes and many mesh vertices, the speed of CSRBF interpolation becomes very slow because the new positions of the subdivided mesh vertices in the fusion region are determined by the mesh vertices' surrounding boundary contours in the original models. The closer the original mesh points are to the sampling vertices, the more they affect the result. Based on this characteristic, there is no need to use all the mesh vertices of the source models to calculate the interpolation surface with CSRBF because only a portion of the vertices around the boundary contours is required. We present a region-controlled surface interpolation acceleration method.

We obtain the required mesh vertices from the original models by cutting planes. According to the method in Section 3.2, we can obtain the vertices $\bar{p}$ and $\bar{q}$, which are the centres of the upper and lower boundaries, respectively. The normal vector is $\vec{n}$. We set $\bar{c} = \left(\bar{p}+\bar{q}\right)/2$ and define plane α given $\bar{c}$ and $\vec{n}$. Then, by calculating the distances of all vertices to the plane, we obtain the shortest distance l and the longest distance L. Given the proportional coefficient θ, $\theta \in [0,1]$, we can obtain the current distance value as $d=l+\theta(L-l)$. Thus, we can define plane β, which is parallel to plane α. We can control the number of mesh vertices by adjusting θ. The vertices between planes α and β are involved in CSRBF-based implicit surface interpolation.

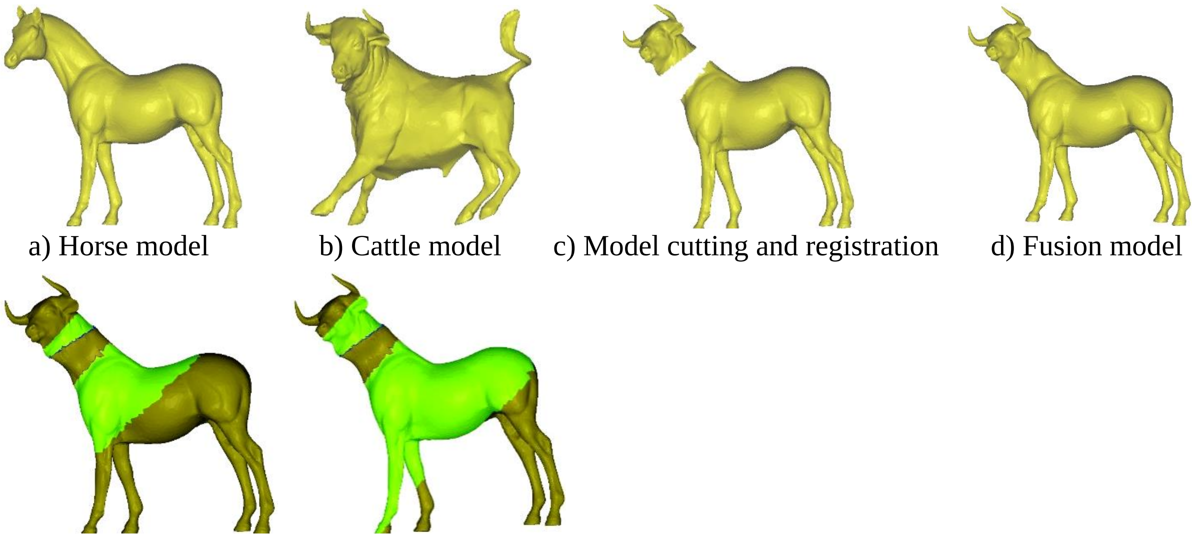
Fig. 11 compares the mesh fusion effects with different θ values. The green parts represent the interpolation region with different θ values. As seen in Fig. 11 a, when θ is 0.2, the interpolation surface is seriously concave because too few mesh vertices are involved in the interpolation calculation, and the meshes' smooth transition effect is poor. Fig. 11 b shows that a more completely smooth transition fusion surface is obtained when θ is 0.3. The fusion effects are relatively the same when θ is 0.5 or 1, as shown in Figs. 11 c and d.



a) $\theta = 0.2$ b) $\theta = 0.3$ c) $\theta = 0.5$ d) $\theta = 1$

Figure 11: Smooth effects with different θ values.

Fig. 12 shows the process of the stitch and smooth transition with our proposed method. Figs. 12 a and b are the source models. Fig. 12 c shows the models cut from the two source models. We align these two cut models to be fused. Fig. 12 d shows the result of fusion and smoothing through the method described in Sections 3 and 4. Figs. 12 e and f show the interpolation region (green part) with θ values 0.3 and 0.5, respectively.



a) Horse model b) Cattle model c) Model cutting and registration d) Fusion model

e) $\theta = 0.3$ f) $\theta = 0.5$

Figure 12: Model fusion of a horse model and cattle model.

Table I shows the time required for CSRBF interpolation for different θ values.

We have conducted many model fusion experiments. When θ is 0.3–0.4, we obtain satisfactory effects of surface interpolation and a smooth transition mesh fusion mesh model. According to the data in Table I, the more experimental model vertices there are, the more obvious the effect of acceleration.

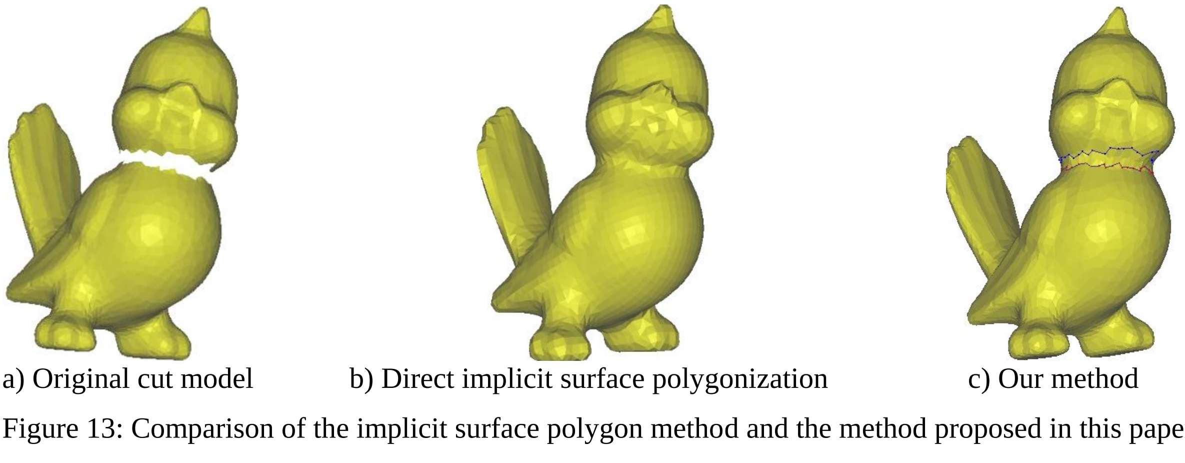
Table I: Interpolated time (seconds) of different models with different θ values.

Model	**Sum of vertices**	$\theta = 0.3$ **Interpolated time**	$\theta = 0.5$ **Interpolated time**	$\theta = 1$ **Interpolated time**
Figure 8-shaped model	718	0.015	0.047	0.172
Horse and cattle	11,944	4.328	10.563	25.438

5.3 Advantages of our fusion method

In this paper, we present a new method for model stitch and fusion. The method can be used to obtain naturally smooth transition surfaces in the models' fusion region. The method of implicit surface polygonization can obtain a better surface smooth transition, which can be used to achieve smooth stitching models [5]. However, as a result of the implicit surface polygonization required to establish the overall implicit surface representation, to re-mesh the entire model by triangulating the zero-valued isosurface from the implicit surface, the original vertex connection topology of the fused models is destroyed and the detailed features of the original surface are lost. In this paper, our proposed method aims only to triangulate the blending region; therefore, our method completely preserves the vertices and triangles of the original mesh model. Fig. 13 compares the results of the direct implicit surface polygonization method [24] and the method in this paper. In Figs. 13 b and c, the scales of the vertices and triangles of the two models are very close. Although the merging region of the models is smooth and natural in Fig. 13 b, the detail in the mouth and tail of the bird is not preserved.

This paper describes an interpolation method that flexibly selects portions of the mesh vertices to interpolate. However, the direct implicit surface polygon method must re-mesh the entire model first. Hence, all the surface vertices of the model must be used for implicit function interpolation; thus, our method improves the mesh fusion efficiency.



a) Original cut model b) Direct implicit surface polygonization c) Our method

Figure 13: Comparison of the implicit surface polygon method and the method proposed in this paper.

6. CONCLUSION

In this paper, we proposed a rapid modelling method based on existing mesh models that extracts desired portions of existing models and creates a new model via mesh fusion. The primary contributions of this paper are as follows:

(1) We propose a processing framework of mesh fusion and smooth transition.

(2) We transform the 3D boundary contour triangulation problem into a planar contour triangulation problem.

(3) The blending region between the meshes uses implicit function interpolation to refine the subdivided mesh, and a smooth transition mesh surface is built.

As we demonstrated in this paper, our method can be used to obtain the smooth transition of the blending region while retaining not only the topology of the original meshes but also the features of the original mesh's surface.

ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China (Grant No. 61672462, 61672463, 61702455), the Science and Technology Plan Projects in Zhejiang Province (Grant No. 2016C33165), the Open Project of Top Key Discipline of Computer Software and Theory in Zhejiang Province (Grant No. ZC323014102).

REFERENCES

[1] Cajal, C.; Santolaria, J.; Samper, D.; Garrido, A. (2015). Simulation of laser triangulation sensors scanning for design and evaluation purposes, *International Journal of Simulation Modelling*, Vol. 14, No. 2, 250-264, doi:10.2507/ijsimm14(2)6.296

[2] Funkhouser, T.; Kazhdan, M.; Shilane, P.; Min, P.; Kiefer, W.; Tal, A.; Rusinkiewicz, S.; Dobkin, D. (2004). Modeling by example, *ACM Transactions on Graphics*, Vol. 23, No. 3, 652-663, doi:10.1145/1015706.1015775

[3] Pedersen, H. K. (1995). Decorating implicit surfaces, *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 291-300, doi:10.1145/218380.218458

[4] Museth, K.; Breen, D. E.; Whitaker, R. T.; Barr, A. H. (2002). Level set surface editing operators, *ACM Transactions on Graphics*, Vol. 21, No. 3, 330-338, doi:10.1145/566570.566585

[5] Biermann, H.; Martin, I.; Bernardini, F.; Zorin, D. (2002). Cut-and-paste editing of multiresolution surfaces, *ACM Transactions on Graphics*, Vol. 21, No. 3, 312-321, doi:10.1145/566654.566583

[6] Kanai, T.; Suzuki, H.; Kimura, F. (1998). Three-dimensional geometric metamorphosis based on harmonic maps, *The Visual Computer*, Vol. 14, No. 4, 166-176, doi:10.1007/s003710050132

[7] Singh, K.; Parent, R. (2001). Joining polyhedral objects using implicitly defined surfaces, *The Visual Computer*, Vol. 17, No. 7, 415-428, doi:10.1007/s003710100115

[8] Adams, B.; Dutre, P. (2003). Interactive boolean operations on surfel-bounded solids, *ACM Transactions on Graphics*, Vol. 22, No. 3, 651-656, doi:10.1145/882262.882320

[9] Pauly, M.; Keiser, R.; Kobbelt, L. P.; Gross, M. (2003). Shape modeling with point-sampled geometry, *ACM Transactions on Graphics*, Vol. 22, No. 3, 641-650, doi:10.1145/882262.882319

[10] Bernstein, G. L.; Wojtan, C. (2013). Putting holes in holey geometry: topology change for arbitrary surfaces, *ACM Transactions on Graphics*, Vol. 32, No. 4, Paper 34, doi:10.1145/2461912.2462027

[11] Sorkine, O.; Cohen-Or, D.; Lipman, Y.; Alexa, M.; Rössl, C.; Seidel, H.-P. (2004). Laplacian surface editing, *Proceedings of the Eurographics / ACM SIGGRAPH Symposium on Geometry Processing*, 175-184, doi:10.1145/1057432.1057456

[12] Lin, J.; Jin, X.; Wang, C. C. L. (2006). Sketch based mesh fusion, Nishita, T.; Peng, Q.; Seidel, H.-P. (Eds.), *Advances in Computer Graphics*, Lecture Notes in Computer Science, Vol. 4035, 90-101, Springer, Berlin, doi:10.1007/11784203_8

[13] Jin, X.; Lin, J.; Wang, C. C. L.; Feng, J.; Sun, H. (2006). Mesh fusion using functional blending on topologically incompatible sections, *The Visual Computer*, Vol. 22, No. 4, Paper 266, doi:10.1007/s00371-006-0004-8

[14] Lou, R.; Pernot, J.-P.; Mikchevitch, A.; Veron, P. (2010). Merging enriched Finite Element triangle meshes for fast prototyping of alternate solutions in the context of industrial maintenance, *Computer-Aided Design*, Vol. 42, No. 8, 670-681, doi:10.1016/j.cad.2010.01.002

[15] Lai, L. M.; Yuen, M. M. F. (2015). Blending of mesh objects to parametric surface, *Computers & Graphics*, Vol.46, 283-293, doi:10.1016/j.cag.2014.09.030

[16] Centin, M.; Pezzotti, N.; Signoroni, A. (2015). Poisson-driven seamless completion of triangular meshes, *Computer Aided Geometric Design*, Vol. 35-36, 42-55, doi:10.1016/j.cagd.2015.03.006

[17] Tan, K. S.; Wong, S. V.; Megat Ahmad, M. M. H. (2016). Development of high fidelity finite element model of motorcycle telescopic front fork, *International Journal of Simulation Modelling*, Vol. 15, No. 3, 436-449, doi:10.2507/ijsimm15(3)4.344

[18] Budak, I., Mirkovic, S., Sokac, M.; Santosi, Z.; Puskar, T.; Vukelic, D. (2016). An approach to modelling of personalized bone grafts based on advanced technologies, *International Journal of Simulation Modelling*, Vol. 15, No. 4, 637-648, doi:10.2507/ijsimm15(4)5.357

[19] Keppel, E. (1975). Approximating complex surfaces by triangulation of contour lines, *IBM Journal of Research and Development*, Vol. 19, No. 1, 2-11, doi:10.1147/rd.191.0002

[20] Fuchs, H.; Kedem, Z. M.; Uselton, S. P. (1977). Optimal surface reconstruction from planar contours, *Communications of the ACM*, Vol. 20, No. 10, 693-702, doi:10.1145/359842.359846

[21] Christiansen, H. N.; Sederberg, T. W. (1978). Conversion of complex contours line definitions into polygonal element mosaics, *ACM SIGGRAPH Computer Graphics*, Vol. 12, No. 3, 187-192, doi:10.1145/800248.807388

[22] Ganapathy, S.; Dennehy, T. G. (1982). A new general triangulation method for planar contours, *ACM SIGGRAPH Computer Graphics*, Vol. 16, No. 3, 69-75, doi:10.1145/965145.801264

[23] Morse, B. S.; Yoo, T. S.; Rheingans, P.; Chen, D. T.; Subramanian, K. R. (2001). Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions, *Proceedings of SMI 2001 International Conference on Shape Modeling and Applications*, 89-98, doi:10.1109/SMA.2001.923379

[24] Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree, *Advances in Computational Mathematics*, Vol. 4, No. 1, 389-396, doi:10.1007/BF02123482

[25] Loop, C. T. (1987). *Smooth Subdivision Surfaces Based on Triangles*, Master Thesis, University of Utah, Salt Lake City