Smooth Approximation to Surface Meshes of Arbitrary Topology with Locally Blended Radial Basis Functions

Mingyong Pang[1,2], Weiyin Ma[1], Zhigeng Pan[3] and Fuyan Zhang[2]

[1]City University of Hong Kong, mewma@cityu.edu.hk
[2]Nanjing University, panion@graphics.nju.edu.cn
[3]Zhejiang University, zgpan@cad.zju.edu.cn

ABSTRACT

In this paper, we present a new approach for smooth approximation of surface meshes with arbitrary topology and geometry. The approach is based on the well-known radial basis functions (RBFs) for local shape approximation combined with a blending operator and a family of normalized weight functions for global surface construction. Our method first defines a local approximation using locally supported RBF for every vertex of the input mesh. A single global smooth surface is then constructed by blending the local approximations using weight functions associated with mesh vertices. A projection procedure is employed for visualizing the global surface using the local parameterization defined by barycentric coordinates for each facet of the input mesh. The approach provides a robust and efficient solution for smooth surface construction from various 3D mesh models.

Keywords: surface reconstruction, RBF approximation, blending, domain decomposition.

1. INTRODUCTION

Polygonal meshes are widely used in the CG (computer graphics) and CAGD (computer aided geometric design) communities for the representation of 3D shapes due to their simplicity of data structure and flexibility in representing various shapes with arbitrary topology and geometry. Vertices of a mesh are in most cases sampled from a smooth surface of certain object and the mesh provides a discrete approximation of the original surface. In many applications, it is most important to have a smooth surface representation of the underlying object rather than just with a surface mesh and how to construct a smooth surface from a given mesh is thus a very important field of study in CG and CAGD.

In general, there are two classes of techniques for the construction of a continuous surface or function from a given 3D mesh, i.e. parametric or implicit. The later uses implicit surfaces to approximate meshes by building a family of implicit real-valued scalar functions and the constructed surfaces are defined as its zero level-sets. In literature, one may find various related work in this area. Muraki[1] fitted an implicit surface from a given point-set using a linear combination of Gaussian blobs. Hoppe et al[2] used a distance function to represent approximation of a point-cloud. Lim et al[3] distributed a set of spheres in the scattered point set by employing a Delaunay partition and a process of non-linear optimization, and then used blended-union of the spheres to construct an implicit surface from the point set.

The latest trend of implicit surface construction is related to RBF based techniques associated with moving least squares and level-set method interpolating or fitting large scattered point-sets. Carr et al[4] constructed cranium surface from CT data by RBF interpolating method. Yngve et al[5] focused their work on how to simplify the data-set and makes it manageable in calculation of surface construction. Wendland[6] combined compactly supported RBF interpolation with partition of unity for constructing surfaces from large scale point-sets. Morse et al[7] gave a process in using the Wendland's method to construct a surface from a larger data set. In recent studies of surface reconstruction of unorganized point-set, researchers focused their attention on a so-called domain decomposition method[8]. The method first divides the data-set into several tractable segments and then handles them respectively. Ohtake et al[9] used the partition of unity approach, a special domain decomposition, and presents an algorithm to construct surface models from large sets of points. Tobor et al[10] presented another approach similar to Ohake's, but they used RBFs as basis functions in local approximating. At the same time, they used different blending weight functions for constructing a global surface. In [11], Ohtake et al further proposed a hierarchical approach to 3D scattered data interpolation and fitting with compactly supported RBF.

In this paper, we represent an RBF-based method that is capable of constructing a globally smooth approximating implicit surface from a given surface mesh. The method is built upon the idea of domain decomposition and makes use of the connectivity of the input mesh. It first constructs a local RBF approximation for each vertex of the initial mesh, and then blends the local approximations to yield a smooth global surface. A parameterization-based process is employed to visualize the final surface. In the rest of the paper, Sec. 2 provides a theoretical background of our method. In Sec. 3, we present the basic idea and steps of the proposed algorithm and discuss the visualization of the constructed surface. Sec. 4 gives some experimental results produced with our method followed by conclusions of our work in Sec. 5.

2. RBF-BASED SURFACE RECONSTRUCTION

An implicit surface is defined as the underlying surface that satisfies the governing equation $f(\mathbf{v})=0$, for $\mathbf{v}\in R^3$, where f is a continuous implicit function defined in R^3. Implicit surface $f(\mathbf{v})=0$ partitions the space into two halves, i.e. $f(\mathbf{v})>0$ and $f(\mathbf{v})<0$. For a given mesh M, suppose its vertices $\mathrm{V}=\{\mathbf{v}_i=(\mathrm{x}_i,\mathrm{y}_i,\mathrm{z}_i)\}_{\mathrm{i}=0}^{N}$ sampled from a smooth surface S. The objective of surface construction from M is to find an implicit function f such as its zero level-set $f(\mathbf{v})=0$ approximates S in a reasonable manner with

$$\sum_{i=0}^{N} f^2(\mathbf{v}_i)\rightarrow \min . \tag{1}$$

The RBF approach has proven to be very useful in shape modeling. It is built on strict mathematical theory and the resulting surface is a generalized thin-plate spline interpolating the scattered data[7]. For a data-set V acquired by a 3D range scanning device, the points in V should be equipped with unit normals that indicate the surface orientation in order to perform RBF calculation. Otherwise, these normals can be estimated either from the initial scans during the shape acquisition phase or through local least-square fitting to V. For a given mesh M, they can be directly evaluated from the connectivity of M.

In order to avoid trivial solution with which the constant function $f(\mathbf{v})\equiv 0$ is achieved in the entire space of R^3, some off-surface points $\{\mathbf{v}_j\}_{j=N+1}^{M}$, named *offset points*, should be added when RBF method is used. At the same time, value of f at each offset point $\mathbf{v}_j$, denoted by d_j, should also be estimated. In this case, the problem of approximating point-set described in Equ. (1) becomes a problem in finding a smooth implicit function f, such as

$$\sum_{i=0}^{N} f^2(\mathbf{v}_i)+\sum_{i=N+1}^{M}\left(f(\mathbf{v}_i)-d_i\right)^2\rightarrow \min . \tag{2}$$

The smoothness of f above can be measured by minimizing the following energy function [7, 12]:

$$E(f)=\int\left|\frac{\partial^2 f}{\partial x^2}+\frac{\partial^2 f}{\partial y^2}+\frac{\partial^2 f}{\partial z^2}+2\frac{\partial^2 f}{\partial x\partial y}+2\frac{\partial^2 f}{\partial x\partial z}+\frac{\partial^2 f}{\partial y\partial z}\right|dxdydz . \tag{3}$$

The problem of minimizing energy in Equ. (2) can be solved by RBF method, its analytic solution can be represented as [12]

$$f(\mathbf{v})=\sum_{j=0}^{M} w_j\phi(\left\|\mathbf{v}-\mathbf{v}_j\right\|)+P(\mathbf{v}) \tag{4}$$

where, w_j are real value weights and ϕ is a RBF function $\phi: R\rightarrow[0,\infty)$, $P(\mathbf{v})$ is a polynomial of degree one to represent linear and constant parts of f, which ensures that the resulting surface is affine invariable with respect to the input data-set. In Equ. (4), the unknown weights w_j and coefficients p_k of $P(\mathbf{v})$ can be uniquely determined by the following interpolating conditions [12]

$$f(\mathbf{v}_i)=\sum_{j=0}^{M} w_j\phi(\left\|\mathbf{v}_i-\mathbf{v}_j\right\|)+P(\mathbf{v}_i),\ \ i=0,1,\cdots,M \tag{5}$$

and the following orthogonal conditions [12]

$$\sum_{j=0}^{M} w_j=\sum_{j=0}^{M} w_j x_j=\sum_{j=0}^{M} w_j y_j=\sum_{j=0}^{M} w_j z_j=0 . \tag{6}$$

Let $\phi_{ij} = \phi(\| \mathbf{v} - \mathbf{v}_j \|)$, $\mathrm{P}(\mathbf{v}) = \mathrm{p}_0 + \mathrm{p}_1\mathrm{x} + \mathrm{p}_2\mathrm{y} + \mathrm{p}_3\mathrm{z}$ and $f_i = f(\mathbf{v}_i)$, for $i = 0,1,\cdots,M$, then it is easy to derive the following linear system from Equ. (5) and Equ. (6):

$$\begin{pmatrix} \Phi & C \\ C^T & 0 \end{pmatrix} \begin{pmatrix} W \\ P \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix}, \tag{7}$$

where, $\Phi = (\phi_{ij})_{M\times M}$, $F = (f_i)_{M\times 1}$, and

$$C = \begin{pmatrix} 1 & x_0 & y_0 & z_0 \\ 1 & x_1 & y_1 & z_1 \\ \cdots & \cdots & \cdots & \cdots \\ 1 & x_M & y_M & z_M \end{pmatrix}, \quad W = \begin{pmatrix} w_0 \\ w_1 \\ \cdots \\ w_M \end{pmatrix}, \quad P = \begin{pmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{pmatrix}. \tag{8}$$

Since the coefficient matrix of Equ. (7) is symmetric and semi-positive, the system has unique solution and can be solved using LU-decomposition. Depending on types of application, we can select the RBF function as thin-plate spline $\phi(d) = d^2 \log(d)$, Gaussian function[4] $\phi(d) = \exp(-cd^2)$, multi-quadric function $\phi(d) = \sqrt{d^2 + c^2}$ [12], tri-harmonic $\phi(d) = |d|^3$, or Wendland's compactly support RBF function [6] $\phi(d) = (1-d)_+^4(4d+1)$, and so on.

3. PRACTICAL ALGORITHMS FOR SMOOTH SURFACE APPROXIMATION

In this section, we discuss how to construct RBF implicit approximation from a triangle mesh with arbitrary topology. Since an arbitrary polygonal mesh can be transformed into a triangle mesh through local triangulation of the facets, the algorithm discussed here has also been applied to general polygonal meshes. The main algorithm consists of two steps: a) constructing a local interpolation at each vertex of the input mesh; and b) blending local interpolations to a global smooth surface using locally-supported weight functions. The main difference between our algorithm and other RBF based surface reconstruction from point-set data is twofolds. Our method makes use of the connectivity information of a mesh and eliminates an expensive process of computing octree-based partition of space commonly used for the construction of global implicit surface [9]. In addition, we employ a parameterization-based projection procedure to visualize the global surface.

3.1 Construction of Local RBF Approximations

Let $\mathbf{v}_0$ be an arbitrary vertex of a given triangle mesh M, $\{\mathbf{v}_i\}_{i=1}^K$ be the set of vertices incident to $\mathbf{v}_0$, and $\{F_i\}_{i=1}^K$ be the set of facets incident to $\mathbf{v}_0$. An *umbrella* structure, denoted by $U(\mathbf{v}_0)$, is defined as $\{\mathbf{v}_0\} \bigcup \{\mathbf{v}_i\} \bigcup \{F_i\}$ (see Fig. 1), and $\mathbf{v}_0$ is called the *center* of the umbrella.

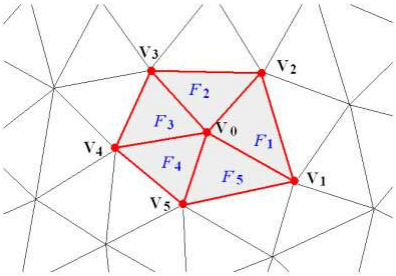


Fig. 1. An umbrella structure centered at $\mathbf{v}_0$ for local approximation.

For an umbrella $U(\mathbf{v}_0)$ centered at $\mathbf{v}_0$, the normal vector at vertex $\mathbf{v}_0$ can be estimated as:

$$N_{\mathbf{v}_0} = \frac{1}{\sum_{i=1}^K A(F_i)} \sum_{i=1}^K [A(F_i) \cdot N(F_i)] \tag{9}$$

where, $A(F_i)$ and $N(F_i)$ are area and normal of facet F_i, respectively. As a result, we can obtain unit normal of every vertex $\mathbf{v}_i$ in mesh M by formula $\mathbf{n}_{\mathbf{v}_i} = N_{\mathbf{v}_i} / \| N_{\mathbf{v}_i} \|$.

In order to evaluate offset points necessary for local RBF approximation, the longest and shortest lengths of edges incident to a vertex, say $\mathbf{v}_0$, can be respectively expressed by maxlen($\mathbf{v}_0$) and minlen($\mathbf{v}_0$) as:

$$\begin{cases} \text{maxlen}(\mathbf{v}_0) = \max\{\|\mathbf{v}_0 - \mathbf{v}_i\|_{i=1}^{K}\} \\ \text{minlen}(\mathbf{v}_0) = \min\{\|\mathbf{v}_0 - \mathbf{v}_i\|_{i=1}^{K}\}. \end{cases} \tag{10}$$

The offset points of $\mathbf{v}_0$ can then be defined as

$$\begin{cases} \mathbf{v}_{0+}^{aux} = \mathbf{v}_0 + [\mathbf{n}_{\mathbf{v}_0} \cdot \text{minlen}(\mathbf{v}_0) \times 0.1] \\ \mathbf{v}_{0-}^{aux} = \mathbf{v}_0 - [\mathbf{n}_{\mathbf{v}_0} \cdot \text{minlen}(\mathbf{v}_0) \times 0.1]. \end{cases} \tag{11}$$

The functional value corresponding to the offset points are defined as $f(\mathbf{v}_{0\pm}^{aux}) = \pm\text{minlen}(\mathbf{v}_0) \times 0.1$. Here, in order to ensure that the offset points is sufficiently close to $\mathbf{v}_0$ while maintaining proper functionality of the auxiliary points and robustness of local approximation, we position the auxiliary points on the surface normal $\mathbf{n}_{\mathbf{v}_0}$ of $\mathbf{v}_0$ and away from the surface with a distance of one-tenth of the shortest edge length $\text{minlen}(\mathbf{v}_0)$ of $\mathbf{v}_0$. In addition, two symmetric offset points $\mathbf{v}_{0\pm}^{aux}$ are selected at each side of the underlying surface at $\mathbf{v}_0$, which is reasonable if the underlying field defined by the implicit function in 3D space is smooth and continuous. The selection of a different functional value would certainly affect the orthogonal gradient distribution of the underlying field. However, as two symmetric auxiliary points are used with respect to the underlying surface, the selection would not be so sensitive to the final constructed implicit surface. As we are only interested in the underlying surface, but not the field itself, the proposed auxiliary points should be appropriate.

For an umbrella centered at $\mathbf{v}_0$, we select discrete point-set to construct its local approximation as $\{\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_K\} \bigcup \{\mathbf{v}_{0\pm}^{aux}, \mathbf{v}_{1\pm}^{aux}, \cdots, \mathbf{v}_{K\pm}^{aux}\}$. Here, we notice $f(\mathbf{v}_i) = 0, i = 0, \cdots, K$. According to Equ. (7), the coefficients of local RBF implicit approximation can be worked out easily and we denote the local surface by $L_{\mathbf{v}_0} : \text{f}_{\mathbf{v}_0}(\mathbf{v}) = 0$.

For a triangle mesh, the valence of an inner vertex is at least 3. In this case, there are at least 12 points used to construct a local surface. For most vertices of the mesh, the average valence should approximately be 6, i.e. there are about 21 points used to construct the local approximation of the umbrella. This guarantees the stability of local construction. In this paper, we select the tri-harmonic, $\phi(d) = |d|^3$, as RBF function in local surface approximation and Fig. 2 shows an example for local approximation of an umbrella.

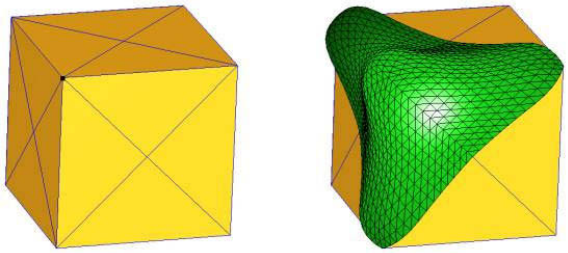


Fig. 2. Local RBF approximation (right, green) of an umbrella in a cube mesh (left).

3.2 Blending for Global Surface Construction

We develop a blending operator, which is similar to that of [13] for other functions, to integrate local approximations into a global surface. The basic requirements are shape local control and smooth blending. At the neighborhood of $\mathbf{v}_0$, the shape of the global surface should basically be determined by the positions of vertices in $U(\mathbf{v}_0)$. Furthermore, the further a vertex in the mesh away from the center of $U(\mathbf{v}_0)$, the weaker it affects the shape of the approximation of the umbrella. When the distance from a vertex to center of $U(\mathbf{v}_0)$ is larger than certain threshold, the vertex won't bring any effect on the shape at all.

Let us consider a bounded domain Ω in R^3 and a set of local non-negative compactly supported weight functions $\{w_i(\mathbf{v}) \mid i = 0, \cdots, N\}$ with $\Omega \subseteq \bigcup_i \text{supp}(w_i)$. These weight functions can always be normalized such that the sum of all weights equals to 1 at all positions of Ω, i.e., the partition of unity property commonly required for acting as basis functions[13]. Let $\{\varphi_i\}$ denote the normalized weight functions of $\{w_i(\mathbf{v})\}$ defined on sub-domains of Ω, we have

$$\varphi_i(\mathbf{v}) = \frac{w_i(\mathbf{v})}{\sum_{j=1}^{N} w_j(\mathbf{v})},\ i = 0,1,\cdots,N. \tag{12}$$

Moreover, let us consider the case that the supported center of w_i is located at vertex $\mathbf{v}_i$ in the initial mesh. The normalized weight functions $\{\varphi_i\}$ will be used as the blending function for defining a global approximation from local approximations associated to individual vertices of the initial control mesh. The global approximation on Ω can be defined as follows [9]:

$$f(\mathbf{v}) = \sum \varphi_i(\mathbf{v}) f_{\mathbf{v}_i}(\mathbf{v}). \tag{13}$$

In the paper, for approximation purposes, we use locally supported weight functions on sphere domain in R^3 as blending weights, which are defined by quadratic B-spline $B(t)$:

$$w_i(\mathbf{v}) = B\left(\frac{\|\mathbf{v} - \mathbf{v}_i\|}{\text{maxlen}(\mathbf{v}_i)}\right). \tag{14}$$

If an interpolation is required, we can use the inverse-distance singular weights:

$$w_i(\mathbf{v}) = \left[\frac{(\text{maxlen}(\mathbf{v}_i) - \|\mathbf{v} - \mathbf{v}_i\|)_+}{\text{maxlen}(\mathbf{v}_i)\|\mathbf{v} - \mathbf{v}_i\|}\right]^2 \tag{15}$$

where

$$(a)_+ = \begin{cases} a, \ if\ a > 0 \\ 0, \ otherwise \end{cases} \tag{16}$$

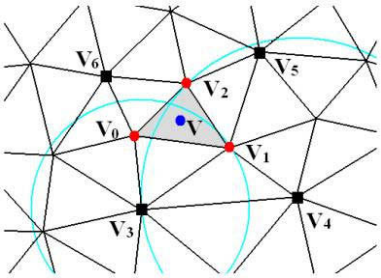


Fig. 3. Blending of local RBF approximations: A surface patch corresponding to the shaded facet on global approximation is defined as a blending surface of a set of local RBF approximations

As mentioned above, for each vertex $\mathbf{v}_0$ of mesh M, its support function is defined in the form of Equ.(14) with the center and radius of the supported domain being $\mathbf{v}_0$ and maxlen($\mathbf{v}_0$), respectively. In this case, the union of all supported domains of vertices of M is exactly Ω that covers the entire mesh.

As an example, we discuss the representation of Equ. (13) in the vicinity of an arbitrary facet $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ of mesh M. Let us first consider the representation at arbitrary vertex $\mathbf{v}$ nearby $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$. We note that, only those local approximations, whose support domain covers the vertex, can affect the value of $f(\mathbf{v})$. In other words, if and only if the distance from the center of an local approximation $L_{\mathbf{v}_k}$ to $\mathbf{v}$ is less than the control radius of $L_{\mathbf{v}_k}$, i.e., maxlen($\mathbf{v}_k$), the approximation can bring an effect on $f(\mathbf{v})$. So, Equ. (13) becomes

$$f(\mathbf{v}) = \sum_{\|\mathbf{v}_i - \mathbf{v}\| < \text{maxlen}(\mathbf{v}_i)} \varphi_i(\mathbf{v}) f_{\mathbf{v}_i}(\mathbf{v}) \tag{17}$$

where, the weight function φ_i can be calculated by Equ. (14) and Equ. (12). On the other hand, on a global surface generated by blending local approximations, each facet of M has a corresponding patch on the global surface and the complete global approximation is just a union of all individual patches corresponding to the parametric domain of individual facets of M . Based on Equ. (17), we can find all local approximations for defining each patch. In Fig. 3, triangle $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ is a facet of M and its corresponding surface patch on global approximation is determined by the local approximations of $L_{\mathbf{v}_0}$, $L_{\mathbf{v}_1}$, $L_{\mathbf{v}_2}$, $L_{\mathbf{v}_3}$, $L_{\mathbf{v}_4}$, and so on. These local approximations can be found by the connectivity of M or an optimized k_d-tree research strategy .

3.3 Global Surface Visualization

Global approximation of the input mesh is an implicit surface represented by a single and complex implicit function. At present, there are two types of algorithms to render implicit surfaces. Ray-tracing based method is only applicable to off-line rendering because it involved heavy computation, whereas, polygonization method implements rapid visualization of implicit surfaces by converting them into polygonal meshes. Among various methods reported so far, the optimized Marching Cubes (MC) based algorithms can provide rapid and efficient polygonization and Marching Triangles based method can give high quality of triangulation. Velho [14] presented a simple and efficient algorithm through a physical process of simulating particle movement. Jin et al [15] combines subdivision and physical simulation process to render the so-called subdivision interpolating implicit surfaces.

Based on Velho's and Jin's work, we propose a piecewise rendering method of the global approximation. For a triangular facet $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ of mesh M , we first partition the planar domain $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ into n^2 sub-triangles as illustrated in Fig.4 by uniformly subdividing the edges into n segments. For each vertex of the sub-triangles, the corresponding vertex on the underlying implicit surface are then computed using a mapping or projection procedure. A refined surface mesh of the constructed implicit surface is finally produced using the newly computed vertices on the underlying implicit surface with the same topological connection as that of the sub-triangles for final visualization. In this paper, the vertices of sub-triangles are named *subdivision-points* and all the sub-triangles form a *subdivision net*. In [14] and [15], the projection or mapping of the subdivision-points onto the underlying surface were driven by simulating particles' free movement along gradient descending direction of the surface function. In the paper, however, we use the normals attached to each vertex of the input mesh to find the projected points of the subdivision-points onto the global approximation surface. This approach avoids an expensive procedure for computing the gradients of implicit surfaces.

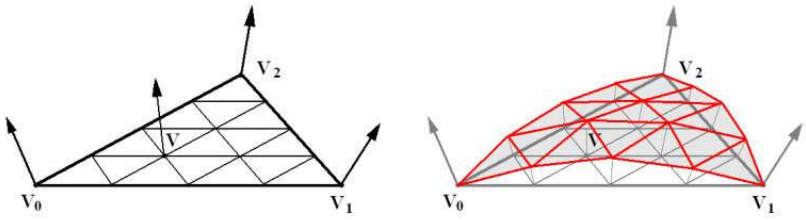


Fig. 4. Patch-wise rendering of implicit surface: 1) a facet of the input mesh after domain partition with corner surface normals and estimated surface normal at an interior point (left); 2) the parametric domain with corresponding surface patch on the underlying surface for visualization (right).

For any subdivision point $\mathbf{v}$ of facet $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$, it is easy to evaluate its barycenter coordinates, (c_0, c_1, c_2) , about points $\mathbf{v}_0$, $\mathbf{v}_1$ and $\mathbf{v}_2$, the vertices of the facet. We define the normal associated to $\mathbf{v}$ as

$$\mathbf{n}_\mathbf{v} = \frac{c_0\mathbf{n}_{\mathbf{v}_0} + c_1\mathbf{n}_{\mathbf{v}_1} + c_2\mathbf{n}_{\mathbf{v}_2}}{\left\| c_0\mathbf{n}_{\mathbf{v}_0} + c_1\mathbf{n}_{\mathbf{v}_1} + c_2\mathbf{n}_{\mathbf{v}_2} \right\|} . \tag{18}$$

Following Sec. 3.1, we know that $\mathbf{n}_\mathbf{v}$ points outwards of the surface. If a point $\mathbf{v}$ is not on the surface, we can find a point, at which the sign of the implicit function is opposite to that of $\mathbf{v}$, along either the positive or negative direction

of $\mathbf{n}_{\mathbf{v}}$. Once two points at different sides of the surface are found, an intersect point, named $\mathbf{n}^p$, of the line segment connecting the two points and the implicit surface can be obtained using a bisection search method. The intersect $\mathbf{n}^p$ will be taken as the projected point of $\mathbf{v}$ on the surface for visualization.

Obviously, the resolution of polygonization of the global approximation can be adjusted by increasing or decreasing the number of subdivisions on edges of the facet. As the normals of a mesh are predefined, the normal vector at each subdivision-point is also uniquely defined. The position of the intersect point on the surface can then be represented by a single pure scalar, which is the distance from $\mathbf{v}$ to the intersection point. The distance is positive when the direction from $\mathbf{v}$ to intersection point is coincident to the direction of the normal of $\mathbf{v}$, otherwise negative.

One should notice that, two neighboring facets, $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ and $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_3$, have common global function along their shared edge $\mathbf{v}_0\mathbf{v}_1$, which is the blended sum of the same local approximations weighted by the same weight functions, so that global surface is continuous along the shared edge. In addition, because the normals of subdivision points on a common edge have identical direction in $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ and $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_3$ respectively (subdivision points on $\mathbf{v}_0\mathbf{v}_1$ have the same barycenter coordinates in $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ and $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_3$), their projected points should also be the same. Hence, there should be no crack left on the global surface.

It should also be noted that domain subdivision discussed in this section and the estimation of local surface normal using equation Equ. (18) would only affect the visualization of the underlying surface. The underlying implicit surface, however, remains the same. The underlying implicit surface is uniquely defined by the local RBF approximations and the blending functions used for construction the global approximation.

4. IMPLEMENTATION, RESULTS AND DISCUSSIONS

We have implemented the algorithm presented in the previous section using C++ on an Intel Celeron PC with 1.4GHz CPU and 512M memory under the WinXP operating system. The algorithm consists of two parts as following:

1) Construction of local approximations:

- For every vertex $\mathbf{v}_i$ of the input mesh, compute its $\mathbf{n}_{\mathbf{v}_i}$, maxlen($\mathbf{v}_i$) and minlen($\mathbf{v}_i$);
- Define two offset points for $\mathbf{v}_i$ using $\mathbf{n}_{\mathbf{v}_i}$ and minlen($\mathbf{v}_i$);
- Construct the local approximating surface for $\mathbf{v}_i$ from vertices in $U(\mathbf{v}_i)$ and their offset points.

2) Visualization of the global approximation by rendering the piecewise surface patch for each facet, $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$, of the input mesh:

- Divide $\Delta\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ into n^2 sub-triangles;
- Evaluate: a) all the subdivision points, their barycenters and normals about the vertices $\mathbf{v}_0$, $\mathbf{v}_1$ and $\mathbf{v}_2$; and b) projected points of the subdivision points on the global surface;
- Replace the vertices of subdivision net on the facet with the corresponding projected points and render the net.

Fig. 5 to Fig. 7 illustrate some examples produced using the algorithms presented in this paper. Fig. 5 shows five simple examples. Illustrations of top row are meshes to be approximated and illustrations of the middle and bottom rows are flat shading with wire-frame and Phong shading renderings of their global surfaces, respectively. The initial control mesh of the first example (the first column on left hand side) is a cube, where the valance of the center vertex of each facet is 4 and its neighboring facets are coplanar. With the second example (the second column), the vertex at the intersection point of three planes has complex local geometry. In the third example (middle column), the initial control mesh is an open mesh consisting of a convex peak and a concave valley. The vertex between the peak and the valley vertices also has complex local structure. The surface meshes in two columns on right hand side take on complex topologies. The results show that all the above shapes can be handled with the presented algorithm.

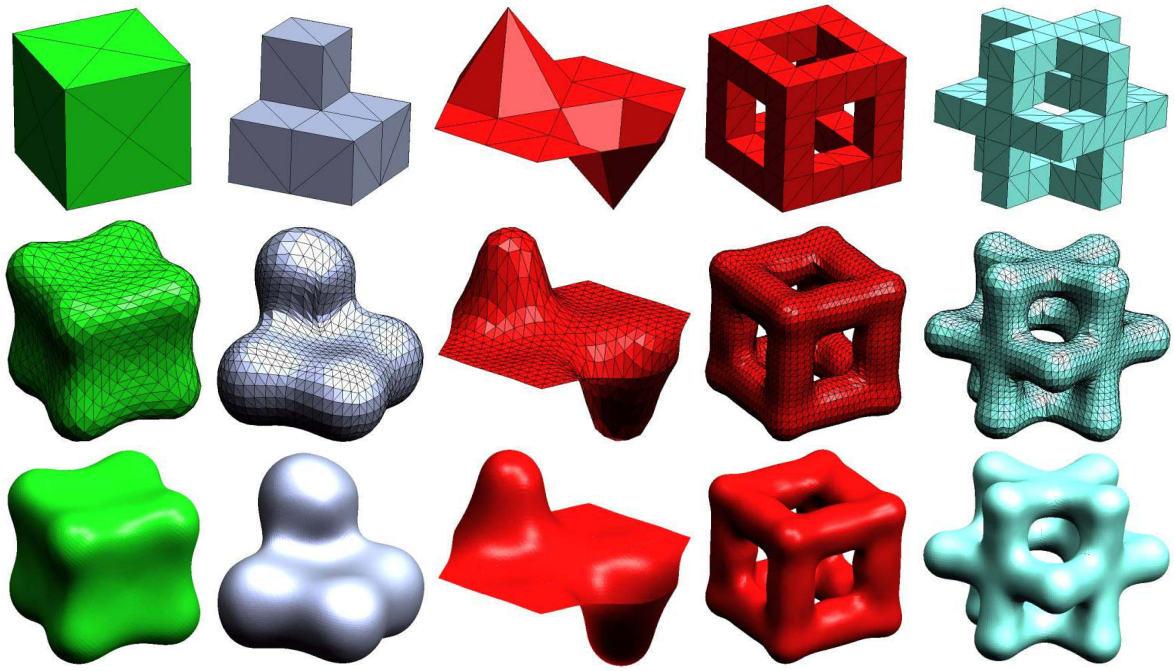
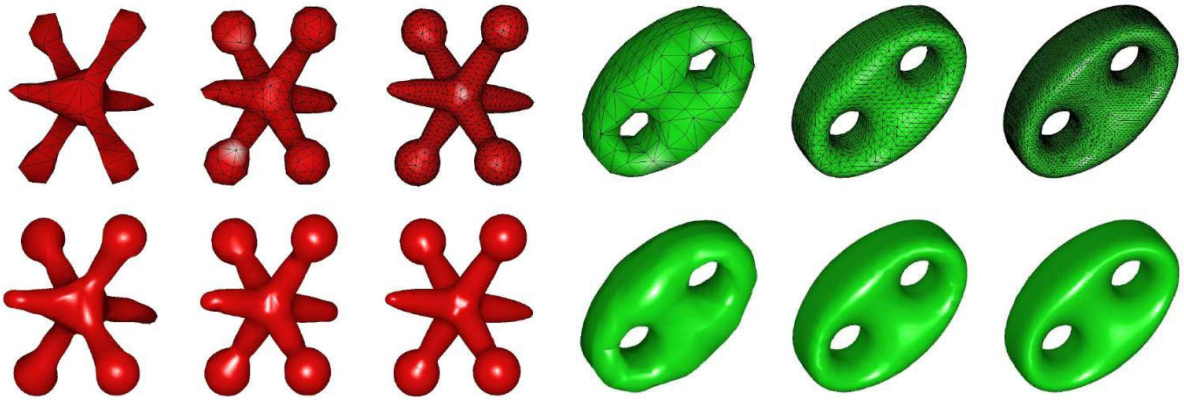


Fig. 5. Illustrations of the constructed surfaces from five different input meshes (top row) with particular vertices or topology and their approximating surfaces (middle and bottom rows).

For a given mesh, because the highest valance of its vertices is a constant, the computing time needed for the construction of a local RBF approximation of one umbrella has its upper limit, which is independent of the number of vertices of the input mesh. Hence, the computing time for the construction of local approximations has linear complexity, $O(N)$, with respect to number N of vertices of the input mesh.

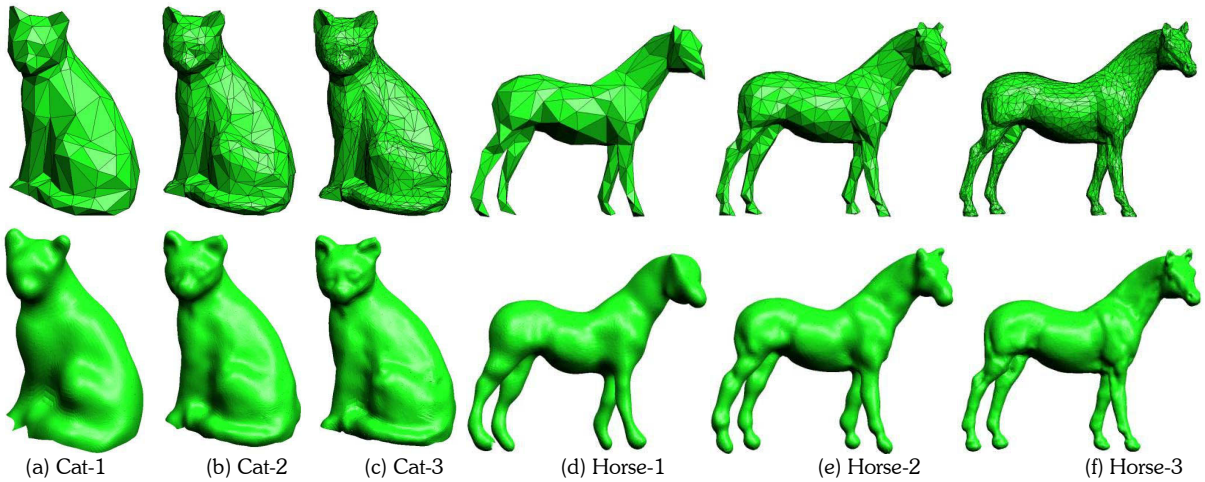
In the step of visualization, the time used for rendering a single patch corresponding to a particular facet is $O(n^2)$, where n is the number of subdivisions for each edge, which is again independent to the number N of the vertices of the input mesh. As a result, the total time complexity for the visualization of the global approximation is $O(N)\times O(n^2)$. When n is decided, which is usually not large, the time needed in the visualization is again linear to the number N of vertices of the input mesh.

Fig. 6 and Fig. 7 present two examples, respectively, constructed using the proposed method. The initial meshes of former are created by MC based sampling procedure from existing smooth surfaces and the control meshes of later are built by mesh simplifier from the dense cat and horse models, which are commonly used in the CG and CAGD communities. These examples demonstrate the capability of the developed algorithms. Tab. 1 summarizes some parameters of the initial control meshes shown in the figures and it also provides the computing time used to construct the local RBF approximations with respect to meshes of different resolution scales. Tab. 2 shows time needed on the second phase for the visualization of the global approximation of various meshes, where "N-segs" indicates the number of subdivisions for each edge for visualization. From the tables, one can find that it is very fast for constructing the local RBF approximations for the vertices of the input meshes and the main computing time of our algorithm is spent on visualization of the resulting surface. The time costs are proportional to the resolution scale of the final meshes in the scale of $O(n^2)$, at the same time, it is linearly proportional to the number N of vertices of the initial input mesh.



(a) Jack-1 (b) Jack-2 (c) Jack-3 (c) Didymous-1 (c) Didymous-2 (c) Didymous-3

Fig. 6. Surface construction of two practical examples generated using various resolution control meshes. The initial meshes (top row) are generated using MC method from smooth surfaces.



(a) Cat-1 (b) Cat-2 (c) Cat-3 (d) Horse-1 (e) Horse-2 (f) Horse-3

Fig. 7. Surface construction of two practical examples commonly used in the CAGD and graphics communities, the control meshes (top row) are obtained from initial dense meshes through mesh simplification.

Mesh	Jack-1	Jack-2	Jack-3	Didymous-1	Didymous-2	Didymous-3
V/T	89/174	155/306	333/662	206/416	297/598	497/998
Time	8.37	14.90	30.23	19.13	27.43	45.07
Mesh	Cat-1	Cat-2	Cat-3	Horse-1	Horse-2	Horse-3
V/T	51/90	145/270	65/898	196/388	244/484	487/970
Time	4.70	13.03	42.33	18.03	44.80	176.33

Tab. 1. Parameters of the meshes shown in Fig. 6 and Fig. 7 and time consumed in constructing their local RBF approximations (V/T: vertices/triangles; Time: ms).

Mesh	Didymous-1	Didymous-2	Didymous-3	Horse-1	Horse-2	Horse-3
3-segs	487	521	814	427	1210	4020
4-segs	824	907	1469	715	1999	6626
5-segs	1206	1405	2278	1035	2987	9929
6-segs	1686	1959	3221	1434	4176	13894

Tab. 2. Computing time in visualization phase of the algorithm (Time unit: ms).

5. CONCLUSION

In this paper, we propose an algorithm for constructing an approximating surface from a triangle mesh with arbitrary topology and geometry using compactly supported radial basis functions. Our algorithm first constructs a local RBF approximation for each umbrella structure centered at an individual vertex of the input mesh. The local approximations are then blended to form a global approximation of the underlying geometry using locally defined weight functions. The visualization makes use of the correspondence, a kind of parameterization, between facets of a mesh and their corresponding patches on the resulting surface. Results show that our algorithm can cope with open or closed mesh with arbitrary topology in the same way. Both the construction process and the visualization procedure are pretty fast. Compared with existed algorithms for constructing approximating surface from meshes, our algorithm has linear time complexity to the number of vertices of the input meshes.

6. ACKNOWLEDGMENT

The work presented in this paper is supported by the Research Grants Council of Hong Kong SAR through research grant #CityU 1131/03E.

7. REFERENCES

[1] Muraki, S., Volumetric shape description of range data using 'blobby model', *Computer Graphics*, Vol. 25, No. 4, 1991, pp 227–235.

[2] Hoppe, H., DeRose, T., and Cuchamp, T., Surface reconstruction from unorganized points, in *Proc. of ACM SIGGRAPH'92*, 1992, pp 71–78.

[3] Lim, C. T., Turkiyyah, G. M., Ganter, M. A. and Storti, D. W., Implicit reconstruction of solids from cloud point sets, in *ACM Symposium on Solid Modeling and Applications*. ACM Press, New York, USA, 1995.

[4] Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., et al, Reconstruction and representation of 3d objects with radial basis functions, in *ACM SIGGRAPH Conference on Computer Graphics*, Los Angeles, CA, August 2001, pp 67–76.

[5] Yngve, G. and Turk, G., Creating smooth implicit surfaces from polygonal meshes, Graphics, Visualization, and Useability Center. *Georgia Institute of Technology*, Tech. Rep. GIT-GVU-99-42, 1999.

[6] Wendland, H., Fast evaluation of radial basis function: Methods based on partition of unity, in *Approximation Theory X: Wavelets, Splines, and Applications*, Chui C. K., Schumaker L. L. and Stockler J., Eds. Nashville TN.: Vanderbilt University Press, 2002, pp 473–483.

[7] Morse, B. S., Yoo, T. S., Rheingans, P., et al, Interpolating implicit surface from scattered surface data using compactly supported radial basis functions, in *Proc. of Shape Modeling International*, 2001, pp. 89–98.

[8] Babuska, I., The partition of unity method, *International Journal of Numerical Methods in Enginerering,* No. 40, 1997, pp 727-758.

[9] Ohtake, Y., Belyaev, A., Alexa, M., Turk G. and Seidel H.-P., Multi-level partition of unity implicits, *ACM Transactions on Graphics*, Vol. 22, No. 3, 2003, pp 463–470.

[10] Tobor, I., Reuter, P. and Schlick, C., Efficient reconstruction of large scattered geometric datasets using the partition of unity and radial basis functions, *Laboratoire Bordelais de Recherche en Informatique, Universite Bordeaux-1*, Tech. Rep. RR-1301-03, 2003.

[11] Ohtake, Y., Belyaev, A. and Seidel, H.-P., 3d scattered data interpolation and approximation with multilevel compactly supported rbfs, *Graphical Models*, Vol. 67, No. 3, 2005, pp 150–165.

[12] Turk, O. G., Modelling with implicit surfaces that interpolate, *ACM Transactions on Graphics*, Vol. 21, No. 4, 2002, pp 855–873.

[13] Chaturvedi, A. K. and Piegl, L. A., Procedural method for terrain surface interpolation. *Computer & Graphics*, Vol. 20, No. 4, 1996, pp 541-566.

[14] Velho, L., Simple and efficient polygonization of implicit surfaces, *Journal of Graphics Tools*, Vol. 1, No. 2, 1996, pp 5–24.

[15] Jin, X., Sun, H. and Peng, Q., Subdivision interpolating implicit surfaces, *Computer & Graphics*, No. 27, 2003, pp 763–772.