Implicit Fitting and Smoothing
Using Radial Basis Functions with Partition of Unity

Xiaojun Wu, Michael Yu Wang, and Qi Xia
Department of Automation and Computer-Aided Engineering
The Chinese University of Hong Kong, Shatin, NT, Hong Kong
xjwu, yuwang, qxia@acae.cuhk.edu.hk

Abstract

A new scheme for 3D reconstruction of implicit surfaces from large scattered point sets based on the radial basis functions (RBFs) is proposed in this paper. The partition of unity (POU) method and a binary tree is used to organize the point sets into some overlapping local subdomains and reconstructing a local surface for each of the subdomains from non-disjunct subsets of the points, we use only a single point at the offset of the surface to avoid the trivial solution of RBF linear system. When the offset point is chosen properly, the technique is not only efficient but also robust, offering a higher level of scalability. The global solution can be obtained by combining the local solutions with POU equations. We also adapt the methodology of level set propagation of a dynamic surface and employ it for smoothing the reconstructed surfaces. We develop versatile computational framework with many benefits in topological flexibility and numerical efficiency.

1. Introduction

Solid modeling and geometric processing has a very wide range of applications in industry, entertainment and other fields. Measurement techniques using laser scanners or range scanners have found widespread use for part duplication or archiving. An interesting example is to use the technology to digitalize valuable historical remains and conserve them in digital forms, such as the Digital Michelangelo Project carried by Stanford Computer Graphics Laboratory. The use of a range scanner or a laser scanner produces large amounts of unorganized point sets. It is desirable to quickly and robustly reconstruct a continuous surface with attributes from the points. There are several approaches to reconstruct surfaces from 3D scattered datasets, such as moving least squares (MLS), algebraic methods and implicit surface methods[1][2][3]. Implicit surface models are popular since they can describe complex shapes with capabilities for surface and volume modeling and complex editing operations are easy to perform on such models.

Radial basis functions (RBFs) attract more attention recently in data interpolation in multi-dimensions [4][5][6][7]. It is identified as one of the most accurate and stable methods to solve scattered data interpolation problems. Using this technique, an implicit surface is constructed by calculating the weights of a set of radial basis functions such they interpolate the given data points. The surface is represented as the zero level set of the implicit function. Thus, if only the sampled points are used directly, it leads to the trivial solution to the linear system to solve for the weights of the radial basis functions. In practice, some interior or exterior constraints are required to the original data points through offsetting each point along the normal. This is a common practice, but it doubles or triples the number of point dataset. Furthermore, the RBFs are of a global support and the resulting system equations are dense. Therefore, it is difficult to use this technique directly to reconstruct implicit surfaces from large point sets consisting of more than several thousands of points. Although the fast multipole method can be utilized to cope with the large point sets [8][9], the far field expansion in the method has to be done separately for every radial basis function and its implementation is intricate and complicated. Another method of compactly supported RBFs can offer a way to deal with large-scale point sets since the involved linear system becomes sparse [10]. Unfortunately, the radius of support has to be chosen globally, which means the approach is not robust and stable against highly non-uniformly distributed point sets where the density of the samples may vary significantly.

In this paper we describe two contributions to the problems of surface reconstruction and smoothing from large unorganized point sets. Firstly, we take the well-known approach of partition of unity (POU) to subdivide the global domain into overlapping local subdomains. In reconstructing a local surface for each of the subdomains...
from non-disjunct subsets of the points, we use only a single point at the offset of the surface. We demonstrate that, when the offset point is chosen properly, the technique is not only efficient but also robust, offering a higher level of scalability. Secondly, we adapt the methodology of level set propagation of a dynamic surface and employ it for smoothing the reconstructed surfaces. We show that with the implicit’s parameterization of RBF representation of implicit surfaces the surface smoothing is described by an ordinary differential equation governing the evolving surface by the geometric flow. This is a versatile computational framework with many benefits in topological flexibility and numerical efficiency.

2. RBF Formulation

The problem of scattered data interpolation can be stated as given a set of fixed points \( x_1, x_2, \ldots, x_N \in \mathbb{R}^n \) on a surface \( S \) in \( \mathbb{R}^3 \) and a set of function values \( f_1, f_2, \ldots, f_N \in \mathbb{R} \), find an interpolant \( \phi: \mathbb{R}^3 \rightarrow \mathbb{R} \) such that

\[
\phi(x_i) = f_i, \quad i = 1, 2, \ldots, N. \tag{1}
\]

With the use of radial basis functions the interpolation function is defined as

\[
\phi(x) = \sum_{j=1}^{N} \alpha_j g_j(\|x - x_j\|) + p(x) \tag{2}
\]

Where \( p(x) \) is a polynomial, \( \alpha_j \) are coefficients corresponding to each basis and \( \| \cdot \| \) is the Euclidean norm on \( \mathbb{R}^3 \). The basic function \( g \) is a real valued function on \( [0, \infty) \), usually unbounded and of a global support.

There are many radial basis functions for use in (2). Some popular basis functions include thin-plate spline \( g(r) = r^2 \log(r) \), Gaussian \( g(r) = \exp(-cr^2) \), multiquadric \( g(r) = \sqrt{r^2 + c^2} \), inverse multiquadric \( g(r) = \sqrt{r^2 + c^2} \), biharmonic \( g(r) = r \), and triharmonic \( g(r) = r^3 \), where \( r = \|x - x_j\| \). The polynomial \( p(x) \) is appended for achieving polynomial precision according to the basis functions used. Additional so-called natural constraints are needed. For example, if \( p(x) \) is a linear polynomial, the coefficients \( \alpha \) must satisfy the following constraints:

\[
\sum_{j=1}^{N} \alpha_j = 0 \quad \text{and} \quad \sum_{j=1}^{N} \alpha_j x_j = \sum_{j=1}^{N} \alpha_j y_j = \sum_{j=1}^{N} \alpha_j z_j = 0.
\]

The equations (1) and (2) and the constraints give rise to the following linear system:

\[
\begin{bmatrix}
g_{11} & g_{12} & \cdots & g_{1N} & 1 & x_1 & y_1 & z_1 \\
g_{21} & g_{22} & \cdots & g_{2N} & 1 & x_2 & y_2 & z_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\
g_{N1} & g_{N2} & \cdots & g_{NN} & 1 & x_N & y_N & z_N
\end{bmatrix}
\begin{bmatrix}
\alpha_1 \\
\alpha_2 \\
\vdots \\
\alpha_N \\
1 \\
1 \\
\vdots \\
1
\end{bmatrix}
= \begin{bmatrix}
f_1 \\
f_2 \\
\vdots \\
f_N
\end{bmatrix} \tag{3}
\]

The solution of the system composed of the weighting coefficients and the polynomial coefficients for the interpolation function \( \phi(x) \). The derivatives of the RBF interpolation function are given as

\[
\phi'(x) = \sum_{j=1}^{N} \alpha_j g'(\|x - x_j\|)
\]

\[
\phi''(x) = \sum_{j=1}^{N} \alpha_j g''(\|x - x_j\|) \tag{4}
\]

3. RBF POU Interpolations

The concept of partition of unity (POU) is rooted in applied mathematics[11]. The main idea of the partition of unity method is to divide the global domain of interest into smaller overlapping subdomains where the problem can be solved locally on a small scale. The local solutions are combined together by using blending functions to obtain the global solution. The smoothness of the global solution in the overlap regions of two subdomains can be guaranteed by a polynomial blending function. The POU method for RBF based surface reconstruction has been applied by Tobor et al. [12] where all the points in each subdomain are offset along the normal to avoid the RBF trivial solutions. For completeness, the method is described here briefly.

The global domain \( \Omega \) is first divided into \( M \) overlapping subdomains \( \{ \Omega_j \}_{j=1}^{M} \) with \( \Omega \subseteq \bigcup \Omega_j \). For a partition of unity on the set of subdomains \( \{ \Omega_j \}_{j=1}^{M} \), we then need to define a collection of non-negative blending functions \( \{ w_j \}_{j=1}^{M} \) with limited support and with \( \sum_{j=1}^{M} w_j = 1 \) in the entire domain \( \Omega \). For each subdomain \( \Omega_j \), the data set of the points within the subdomain is used to compute a local reconstruction function \( \phi_j \) that interpolates the data points. The global reconstruction function \( \Phi \) is then defined as a combination of the local functions:
\[ \Phi = \sum_{i=1}^{M} \phi(x) w_i(x) \]  \hfill (5)  

The blending functions are obtained from a set of smooth functions \( W_i \) by a normalization procedure

\[ w_i(x) = \frac{W_i(x)}{\sum_j W_j(x)} \]  \hfill (6)  

where the condition \( \sum w_i = 1 \) is satisfied. The weighting functions \( W_i \) must be continuous at the boundary of the subdomains \( \Omega_j \). It can be defined as the composition of a distance function \( D_j : R^n \rightarrow [0,1] \) and a decay function \( V : [0,1] \rightarrow [0,1] \), i.e., \( W_i(x) = V \circ D_i(x) \) \[12\].

The distance function has to satisfy \( D_j(x) = 1 \) at the boundary of \( \Omega_j \). For a 3D axis-aligned box defined from two opposite corners \( S \) and \( T \), the distance function \( D_j \) is chosen as

\[ D_j(x) = 1 - \prod_{r \in \{x,y,z\}} \frac{4(x_r - S_r)(T_r - x_r)}{(T_r - S_r)^2} \]  \hfill (7)  

The main condition in choosing the decay function \( V \) is the continuity between the local solutions \( \phi_i \) in the global reconstruction function \( \Phi \). The following functions are suggested to meet various continuity conditions \[12\]

\[ C^0 : V^0(d) = 1 - d \]
\[ C^1 : V^1(d) = 2d^3 - 3d^2 + 1 \]
\[ C^2 : V^2(d) = -6d^5 + 15d^4 - 10d^3 + 1 \]  \hfill (8)  

4. Reconstruction of RBF Implicit Surfaces

4.1 Binary Tree Decomposition

In order to divide the scattered point sets into local problems, we also use a binary tree structure to organize the point sets as in \[12\]. In our algorithm, a quick sorting scheme is utilized to set up the data structure efficiently. Four parameters of a binary tree construction are needed. They are \( T_{\text{leafnode}} \) that controls the bound of points in a leaf node, \( T_{\maxnum} \) that decides the maximal number of points in each subdomain, \( T_{\minnum} \) that determines the minimal number of points in each subdomain, and \( T_{\text{overlap}} \) that is the overlap quotient. Figure 1 shows the process of a binary tree set up, while some examples of the binary tree decomposition of the data set are illustrated in Figure 2.

4.2 Generation of off-surface points

In the implicit representation of the constructed surface with radial basis functions in equation (1), the data interpolation of the surface satisfies

\[ \phi(x_i) = 0, \quad i = 1, 2, \ldots, N \]  \hfill (9)  

Therefore, the system equation (3) becomes trivial. The problem can be overcome by introducing additional constraints with artificially generated so called off-surface points with non-zero values in \( \phi(x_i) = c \neq 0 \). A common practice, as suggested in \[3\], is to introduce an off-surface point for each data point, usually along the normal of the surface. This technique is shown in Figure 3 for the traditional reconstruction scheme (left figure) and the scheme with the partition of unity (center figure).

A potential problem of the conventional technique is that the off-surface points substantially increase the number of data points for interpolation and hence the number of unknown coefficients to be solved in the linear system (3). For a large scale problem the total variables would double or triple the number of sampled data points. In general, it is not necessary to use such a large number of off-surface points. Theoretically, a single off-surface point might be sufficient for the surface reconstruction. Therefore, we propose to introduce a single off-surface point for the local reconstruction of the RBF interpolation
function \( \phi_i \) in each subdomain. Our scheme is illustrated in Figure 3 (right figure).

<table>
<thead>
<tr>
<th>Traditional RBF scheme</th>
<th>Traditional POU-RBF scheme</th>
<th>Our POU-RBF scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \phi(x_i) = c )</td>
<td>( \phi(x_i) = c )</td>
<td>( \phi(x_i) = c )</td>
</tr>
<tr>
<td>( \phi(x_i) = 0 )</td>
<td>( \phi(x_i) = 0 )</td>
<td>( \phi(x_i) = 0 )</td>
</tr>
</tbody>
</table>

Figure 3. Comparison of three RBF schemes

Theoretically speaking, the off-surface point can be anywhere in the subdomain. But if the off-surface point is located on the boundary of the subdomain, the reconstruction system (3) may become unstable, as to be see later in an example. In our algorithm, an efficient scheme is used to generate the off-surface point in each subdomain. As shown in Figure 4, Bdbox1 and Bdbox2 are the bounding boxes of the neighborhood subdomains, SubD1 and SubD2, respectively. Taking Bdbox1 as an example, we first find the center \( P \) of the bounding box which may lie inside or outside of the object. Then, the data point nearest to the point \( P \) is found. This data point is then offset along its normal with a small distance to yield an off-surface point. This off-surface point is then used for the subdomain to reconstruct the local RBF function \( \phi_i \) using equation (3).

4.3. Influence of the off-surface point position

In the above described procedure, an off-surface point is generated near the center of the subdomain. This is guided by our intuition that the surface reconstruction could be unstable if the offset point is located near or on the boundary of the subdomain. Indeed, we have found that if the off-surface point is near the boundary of the subdomain, the numerical accuracy of the surface reconstruction could be problematic. The offset distance of the off-surface point may have a strong influence on the success of the reconstruction scheme. For a large off-set distance the reconstructed surface may have auto-intersections. The tendency of forming erroneous surface elements would be substantially reduced if the offset distance is kept small as suggested in [6]. In Figure 5, we show the auto-intersection effect for a surface reconstructed with an off-surface point on the boundary of its subdomain. Two different choices of the off-surface point are given. In the first case (left figure), the off-surface point is selected in the center of the subdomain as described above. In the second case (right figure), the off-surface point is taken in the overlap region of two neighboring subdomains. In both cases, a well-defined surface is reconstructed without any numerical difficulties.

In figure 6, we show the effects when the off-surface point locates on the boundary of the subdomain but with different offset distance. From figure 6 (d), we can see that the reconstructed result is not stable when the offset distance is larger. So it is a better choice to generate the one off-surface point in the center of bounding box of the point set in subdomain.
5. Reconstruction Results

In this section we present more results of our reconstruction method and compare our method with the conventional POU-RBF scheme reported in [12]. In our implementation of the partition of unity and binary tree decomposition, we solve the local linear system (3) with the singular value decomposition method for each subdomain. Although the data points are organized with a binary tree, the multiresolution evaluation is not needed and just the leaf nodes are evaluated. We also employ the marching cubes algorithm for surface polygonization for ray-tracing and visualization. In other words, the major difference between our implementation and that reported in [12] is that we use the single off-surface point scheme developed in the paper; while a full set of off-surface points are incorporated in [12]. All our results presented in this paper were performed on an notebook of Intel Pentium 1.5 GHz with 512 MB RAM running WinXP.

In Figure 7 several visual examples of the reconstruction of our proposed method are shown. The quality of the reconstructions is highly satisfactory. Some ray traced examples of reconstructed models are shown in Figure 8. In Table 1 we list the quantitative results of the numerical computations for the reconstruction of six different data sets with different point density, where Ttree stands for the time cost of binary tree set up, Trec is the reconstruction time of the local RBF system, Tpoly means the polygonization time of RBF implicit surface, and Ttotal is total time cost of reconstruction. The conventional method of offsetting all data points is used. As reported in [12], the POU-RBF technique exhibits a linear complexity in the reconstruction time with respect to the number of data points.

These six examples are further studied with the proposed method of using a single off-surface point in each subdomain. The numerical results are summarized in Table 2. It is clearly shown that for the total reconstruction time, the use of a single off-surface point reduces the computational effort substantially, more than an order of magnitude in all six cases.

---

Figure 6. Off-surf point on the boundary of subdomain, (a) Offset distance d=0.02, (b) Offset distance d=0.2

Figure 7. Surface reconstruction examples, (a) sculpture point set and surface model, (b) bunny point set and surface model, (c) squirrel point set and surface model, (d) venus point set and surface model
For the purpose of illustrating the computational efforts and complexity of the reconstruction with respect to the number of points, we further illustrate the computational time (in seconds) for the three major tasks of binary tree decomposition, surface reconstruction, and the surface polygonization, as well as the total computational time. For reference purpose, we also include the processing time of [12] (in Table 3) for datasets of various point density as reported in [12] and Tthin stands for the point thinning time for multi-resolution, where a comparable Intel Pentium 1.7 GHz computer with 512 MB RAM running Linux was used.

In Figure 9, graphs of the computation time for binary tree decomposition, surface reconstruction, the surface polygonization, and the total computational time are plotted respectively, based on the data listed in Tables 1, 2, and 3.

### 6. Surface Smoothing by Curvature Diffusion

In this section we extend our POU-RBF method to the problem of surface smoothing. Unlike the majority of the surface smoothing methods developed in the recent years (see [13][14][15][16][17]), our method is not to smooth a mesh model. Instead, we treat an implicit surface model,
which may be reconstructed from scattered data, as a
dynamic surface model with radial basis function
representation. Smoothing such a RBF implicit surface is
considered as a surface propagation under a velocity field
such as that resulting from the curvature diffusion.
We adapt the methodology of level set propagation of a
dynamic surface. We show that with the implicit’s
parameterization of RBF representation of implicit
surfaces the surface smoothing is described by an
ordinary differential equation governing the evolving
surface by the geometric flow. This is a versatile
computational technique with many benefits in
topological flexibility and numerical efficiency.

As well-known in the field of geometric processing, one
common way to smooth a surface to attenuate noise is
through a diffusion process, such as the curvature
diffusion

\[ \dot{x} = \frac{dx}{dt} = \kappa n \]  

with \( n = \frac{\nabla \phi}{|\nabla \phi|} \)
and

\[ \kappa = \nabla n \]

When we substitute the diffusion process into equation
(10), the original dynamic problem becomes an
interpolation problem for the initial values of the
coefficients \( \alpha \).

To time advance the initial values, we shall use the
collocation method that was employed in the original
problem of surface reconstruction of equation (3). We
simply evaluate equations (11)-(14) at each of the centers
\( x_1, x_2, \ldots, x_N \in R^n \) of the radial basis functions. After
also incorporating the polynomial precision constraints,
for example, $\sum_{j=0}^{N} \alpha_j = 0$, we obtain a set of ordinary
differential equations (ODEs) can be simply written as

$$G\dot{\lambda} = B\lambda$$

(15)

where $b_j = \kappa_j \nabla g_j \cdot n_j$

$$G = \begin{bmatrix}
g_{11} & g_{12} & \cdots & g_{1N} & 1 \\
g_{21} & g_{22} & \cdots & g_{2N} & 1 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
g_{N1} & g_{N2} & \cdots & g_{NN} & 1 \\
1 & 1 & 1 & 1 & 0
\end{bmatrix}, \quad \lambda = \begin{bmatrix}
\dot{\alpha}_1 \\
\dot{\alpha}_2 \\
\vdots \\
\dot{\alpha}_N \\
\alpha_{N+1}
\end{bmatrix}

(16)

The set of coupled non-linear ODEs of equation (15) can
be solved by several different ODE solvers such as the
first-order forward Euler's method and higher-order
Runge-Kutta, Runge-Kutta-Fehlberg, Adams-Bashforth,
or Adams-Moulton methods. In the present study, only
the first-order forward Euler's method is used since it is
the simplest solution algorithm for ODE initial condition
problems and often used for comparison with more
accurate algorithms, which are more complex and tedious
to implement. Using Euler's method, an approximate
solution to equation (15) can be given by

$$\lambda^{(n+1)} = G^{-1} (G - \tau B)\lambda^{(n)}$$

(17)

where $\tau$ is the step size. It should be noted that the step
size should be small enough to achieve the numerical
stability due to the Courant-Friedrichs-Lewy (CFL)
condition and to reduce the truncation error.

Therefore, in the present RBF-implicit method,
smoothing the surface is equivalent to transporting the
scalar implicit function $\phi(t, x) = 0$ by solving the system
of coupled non-linear ODEs of equation (15) and the
propagation of the surface can be performed by using the
approximate solution of an explicit scheme (17).

Finally, we present three examples of surface smoothing
based on the mean curvature flow. In Figs. 10, 11, and 12,
the processes of the smoothing are shown with the time
steps $\tau = 10^{-6}$, $\tau = 0.08$, and $\tau = 0.02$, respectively.

Figure 10. Mean curvature smoothing of a sphere

Figure 11. Mean curvature smoothing example
7. Conclusions

In this paper we describe two approaches to the problem of surface reconstruction and surface smoothing from a large scattered data set. There are many other good techniques proposed previously, such as MPU (Multi-level partition of unit implicits), MLS (moving least squares), and compactly supported RBF to reconstruct surfaces from scattered point set. In MPU an algebraic kernel, piecewise quadratic functions, is used to capture the local shape of the surface, where RBF kernel is not used. Compactly supported RBF can not handle with the ununiformly distributed point set. Our reconstruction method is based on the implicit representation with radial basis functions and the partition of unity and we propose to generate a single off-surface point for each subdomain of the partition of unity, in contrast to the conventional RBF reconstruction methods, where a full set of off-surface points are used,. It is shown that this approach reduces the local reconstruction time substantially.

We further extend the RBF-based modeling method for surface smoothing. We combine the dynamic level set equation with the radial basis functions and develop a set of ordinary differential equations for the propagation of a surface under a given velocity field. When using the common curvature flow for surface smoothing to attenuate noise, this approach yield a new method of surface processing without meshes.

Reference

[10] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial basis functions of minimal


