Multi-Scale Reconstruction of Implicit Surfaces with Attributes from Large Unorganized Point Sets

Ireneusz Tobor
LaBRI - CNRS - INRIA
Universite Bordeaux 1
tobor@labri.fr

Patrick Reuter
LaBRI - CNRS - INRIA
Universite Bordeaux 1
preuter@labri.fr

Christophe Schlick
LaBRI - CNRS - INRIA
Universite Bordeaux 1
schlick@labri.fr

Abstract

We present a new method for the multi-scale reconstruction of implicit surfaces with attributes from large unorganized point sets. The implicit surface is reconstructed by subdividing the global domain into overlapping local subdomains using a perfectly balanced binary tree, reconstructing the surface parts in the local subdomains from non-disjunct subsets of the points by variational techniques using radial basis functions, and hierarchically blending together the surface parts of the local subdomains by using a family of functions called partition of unity. The subsets of the points in the inner nodes of the tree for intermediate resolutions are obtained by thinning algorithms. The reconstruction is particularly robust since the number of data points in the partition of unity blending zones can be specified explicitly. Furthermore, the new reconstruction method is valid for discrete datasets in any dimension, so we can use it also to reconstruct continuous functions for the surface's attributes. In a short discussion, we evaluate the advantages and drawbacks of our reconstruction method compared to existing reconstruction methods for implicit surfaces.

1 Introduction

Real-world datasets are often provided as an unorganized set of a large amount of discrete data. The main (and maybe the most difficult) problem to solve, in order to exploit the scattered data, is to efficiently and precisely reconstruct a continuous function starting from this dataset.

In the field of geometric modeling, this problem has become of major importance due to the rapid development of 3D acquisition technologies such as range scanners, where the geometry and its appearance is acquired as a large unorganized point set with attributes. For further processing of the geometry and its appearance, it is desirable to quickly and robustly reconstruct a continuous surface with attributes that is passing either exactly through the points, or, when there is noise in the data, nearby the points. A multi-scale representation of the surface is advantageous, because different precisions of the surface and its appearance are required depending on the type of further processing and the provided time (e.g. detailed visualization, quick preview, modeling, collision detection, storage, progressive network transfer, detail analysis, physically-based operations).

In this paper, we present a new method for the multi-scale reconstruction of surfaces with attributes from large unorganized point sets. The geometry of the resulting surface at any resolution is defined as an implicit surface as the zero-set of a reconstructed defining function $f : \mathbb{R}^3 \rightarrow \mathbb{R}$, and the appearance at a point on the surface is given by the evaluation of the reconstructed continuous functions for the attributes at the desired resolution. Consequently, we can consider the surface reconstruction method as a combination of multi-scale scattered data reconstructions for every involved function.

In order to obtain a multi-scale reconstruction method for large scattered data, we combine three well-known methods: variational techniques using radial basis functions (RBF) are used to solve a set of small local reconstruction problems, thinning algorithms are used to obtain subsets of the data for intermediate resolutions, and the partition of unity (POU) method combines the local solutions together to get the final reconstruction. As we will see, this combination is not only robust and efficient, but also offers a high level of scalability, and moreover it enables the adaptive selection of different precisions of the multi-scale reconstruction.

The contributions of this paper are threefold. First, we present a new multi-scale reconstruction method for large scattered data. Second, we show how this method can be applied to reconstruct implicit surfaces with attributes from large unorganized point sets. Third, in a short survey, we compare existing multi-scale reconstruction methods for implicit surfaces to our new method and evaluate the advantages and drawbacks.

The paper is organized as follows. In Section 2, we present some relevant previous work about the reconstruction of implicit surfaces from unorganized point sets. Section 3 provides the theoretical background for our new re-

construction method. In Section 4, we present our multi-scale reconstruction method and apply it to the reconstruction of implicit surfaces with attributes in Section 5. Section 6 presents experimental results, and in Section 7 we evaluate the advantages and drawbacks of our reconstruction method compared to existing multi-scale reconstruction methods for implicit surfaces before we conclude and indicate some directions to future work.

2 Previous work

About 20 years ago, in an extensive survey, Franke [11] identified radial basis functions (RBFs) as one of the most accurate and stable methods to solve scattered data interpolation problems. The pioneering work to interpolating surfaces using RBFs starting from unorganized point sets can be attributed to Savchenko et al. [25] and Turk and O'Brien [28]. Using these techniques, the implicit surface is calculated by solving a linear system. Unfortunately, since the RBFs have global support, the equations lead to a dense linear system. Hence, both techniques fail to reconstruct implicit surfaces from large point sets consisting of more than several thousands of points.

To overcome this problem, by using Wendland's compactly supported RBFs [29], Morse et al. [21] showed how to reconstruct implicit surfaces from larger point sets since the involved linear system becomes sparse. This algorithm was further improved by Kojekine et al. [17] by organizing the involved sparse matrix into a band-diagonal sparse matrix which can be solved more efficiently. Unfortunately, the radius of support has to be chosen globally, which means that the method is not robust against highly non-uniformly distributed point sets where the density of the samples may vary significantly.

A different approach to interpolate large point sets has been proposed by Carr et al. [6] based on a fast evaluation of RBF technique by Beatson et al. [4, 3] using fast multipole methods. Unfortunately, the far field expansion has to be done separately for every radial basis function, and is very complex to implement.

Another fast evaluation method of RBFs, based on the partition of unity method, was proposed by Wendland in a theoretical survey [30] and a more practical sketch [31] on which we have based our approach.

Ohtake et al. reconstruct implicit surfaces from a large number of points by a multi-scale approach [23] with compactly supported radial basis functions, but it is not feasible for the approximation of noisy data and the number of points is still limited due to the use of compactly supported radial basis functions.

Two other methods to reconstruct implicit surfaces without using radial basis functions have drawn much attention recently. In the first one presented by Alexa et al. [2, 1] called Point Set Surfaces, an implicit surface is reconstructed using a projection operator based on the method of moving least squares [18, 19]. The resulting implicit surface is defined by all points that project on themselves, and the defining function is never calculated explicitly. Unfortunately, the computation of the projection operator is rather expensive since a non-linear optimization problem is involved. Point Set Surfaces were further enhanced in order to get a multi-scale representation of the point set [10].

In the second one, called MPU implicits, developed recently by Ohtake et al. [22], the partition of unity method [12] is used to reconstruct implicit surfaces. By using weighted sums of different types of piecewise quadratic functions capturing the local shape of the surface, implicit surfaces from very large point sets can be reconstructed while preserving sharp features.

Unfortunately, the latter techniques are only feasible to reconstruct implicit surfaces from unorganized points and a reconstruction of a continuous functions for the appearance of the surface from the attributes of the points is not straightforward.

3 Background

3.1 Variational techniques with radial basis functions

Given the set of N pairwise distinct points $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$ of dimension d: $\mathbf{p}_k \in \mathbb{R}^d$, and the set of values $\{h_1, \ldots h_N\}$, we want to find a function $f : \mathbb{R}^d \to \mathbb{R}$ satisfying the constraints

$$f(\mathbf{p}_i) = h_i \quad i = 1 \ldots N. \tag{1}$$

Variational techniques with radial radial basis functions reconstruct the following function that is satisfying these constraints

$$f(\mathbf{p}) = \sum_{i=1}^{N} \omega_i \phi(||\mathbf{p}, \mathbf{p}_i||) + \pi(\mathbf{p}). \tag{2}$$

We denote here $||\mathbf{p}_i, \mathbf{p}_j||$ the Euclidean distance, ω_i the weights, $\phi : \mathbb{R} \to \mathbb{R}$ a conditionally positive definite basis function, and π a polynomial of degree m depending on the choice of ϕ. Some popular basis functions are shown below [16]:

biharmonic	$\phi(r) = r$	with π of degree 1
pseudo-cubic	$\phi(r) = r^3$	with π of degree 1
triharmonic	$\phi(r) = r^3$	with π of degree 2
thin plate	$\phi(r) = r^2 \log(r)$	with π of degree 1

More precisely, the polynomial $\pi(\mathbf{p}) = \sum c_\alpha \pi_\alpha(\mathbf{p})$ with $\{\pi_\alpha\}_{\alpha=1}^{Q}$ is a basis in the d-dimensional null space containing all real-valued polynomials in d variables and of order at most m with $Q = \binom{m+d}{d}$.

As we have an under-determined system with $N+Q$ unknowns ($\omega_\mathbf{i}$ and c_α) and N equations, so-called natural additional constraints for the coefficients ω_i are added in order to ensure orthogonality, so that

$$\sum \omega_i c_1 = \sum \omega_i c_2 = \ldots = \sum \omega_i c_Q = 0. \tag{3}$$

The equations (1), (2), and (3) determine the following linear system:

$$A\mathbf{x} = \mathbf{b} \tag{4}$$

$$A = \begin{bmatrix} \Phi & P^T \\ P & 0 \end{bmatrix} \tag{5}$$

$$\Phi = \left[\phi(||\mathbf{p_i}, \mathbf{p_j}||) \Big|_{\substack{i=1\ldots N \\ j=1\ldots N}} \right]$$

$$P = \left[\pi_\alpha(\mathbf{p_i}) \Big|_{\substack{i=1\ldots N \\ \alpha=1\ldots Q}} \right]$$

$$\mathbf{x} = [\omega_1, \omega_2, \ldots, \omega_N, c_1, c_2, \ldots, c_Q]^T \tag{6}$$

$$\mathbf{b} = [h_1, h_2, \ldots, h_N, \underbrace{0, 0, \ldots, 0}_{Q \text{ times}}]^T \tag{7}$$

The solution vector $\mathbf{x}$ is composed of the weights ω_i and the polynomial coefficients c_α for equation (2) and represents a solution of the interpolation problem given by (1).

If an approximation of the constraints of equation (1) rather than an interpolation is desired, for example when there is noise in the data, one solution is to slightly modify the diagonal of matrix $\Phi \leftarrow \Phi - 8N\pi\rho\mathbb{I}$ (see Carr [6, 7]), where the parameter ρ controls the fitting tolerance (i.e. the result is getting smoother when ρ is increased).

3.2 Partition of unity method

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. More precisely, the global difficult problem is decomposed into several smaller local problems and their local solutions are combined together by using weighting functions that act as smooth blending functions to obtain the global solution.

Consider the global domain Ω and divide it into M overlapping subdomains $\{\Omega_i\}_{i=1}^M$ with $\Omega \subseteq \bigcup_i \Omega_i$. On this set of subdomains $\{\Omega_i\}_{i=1}^M$, we construct a partition of unity, i.e. a collection of non-negative functions $\{w_i\}_{i=1}^M$ with limited support and with $\sum w_i = 1$ in the entire domain Ω.

For each Ω_i, a set $P_i = \{\mathbf{p} \in P | \mathbf{p} \in \Omega_i\}$ is constructed, and a local reconstruction function f_i is computed. The global reconstruction function f_{pou} is then defined as a combination of the local functions

$$f_{pou}(\mathbf{p}) = \sum_{i=1}^{M} f_i(\mathbf{p}) w_i(\mathbf{p}). \tag{8}$$

The condition $\sum w_i = 1$ is obtained from any other set of smooth functions W_i by a normalization procedure

$$w_i(\mathbf{p}) \quad = \quad \frac{W_i(\mathbf{p})}{\sum_j W_j(\mathbf{p})}. \tag{9}$$

The weighting functions W_i have to be continuous at the boundary of the subdomains Ω_i. We define the weighting functions W_i as the composition of a distance function $D_i : \mathbb{R}^n \to [0,1]$, where $D_i(\mathbf{p}) = 1$ at the boundary of Ω_i and a decay function $V : [0,1] \to [0,1]$, i.e. $W_i(\mathbf{p}) = V \circ D_i(\mathbf{p})$.

For a 3D axis-aligned box defined from the two opposite corners S and T we use the following distance function D_i

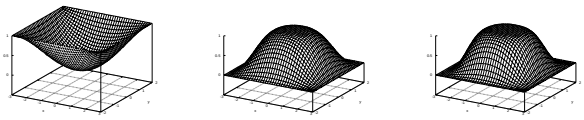
$$D_i(\mathbf{p}) = 1 - \prod_{r \in x,y,z} \frac{4(\mathbf{p}_r - S_r)(T_r - \mathbf{p}_r)}{(T_r - S_r)^2}. \tag{10}$$

The choice of the decay function V determines the continuity between the local solutions f_i in the global reconstruction function f_{pou}. We suggest to use one of the following decay functions that were chosen by including some simple constraints similar to the construction of base spline functions ($V(0) = 1, V(1) = 0, V'(0) = V'(1) = 0$, etc.).

$$\text{continuity } \mathbb{C}^0 : \quad V^0(d) = 1 - d$$

$$\text{continuity } \mathbb{C}^1 : \quad V^1(d) = 2d^3 - 3d^2 + 1$$

$$\text{continuity } \mathbb{C}^2 : \quad V^2(d) = -6d^5 + 15d^4 - 10d^3 + 1$$



(a) $D(\mathbf{p})$ (b) $(V^1 \circ D)(\mathbf{p})$ (c) $(V^2 \circ D)(\mathbf{p})$

Figure 1. 2D interpretation of the distance function D and two different weighting functions for $\mathbb{C}^1$ and $\mathbb{C}^2$ continuity.

The graphs of the distance function D and two different weighting functions are shown in Figure 1. Besides the weighting functions, an appropriate method for the domain decomposition into overlapping subdomains has to be chosen, and in the following section we present a method that uses a binary tree.

4 Multi-scale reconstruction method for large scattered data

4.1 Overview

In this section, we present our new multi-scale reconstruction method for large scattered data based on the background presented in the preceding section. First, the global

domain of interest is subdivided recursively into two overlapping local subdomains containing an equal number of points. The recursion terminates when the number of points in the local subdomains falls below a threshold resulting in a perfect binary tree (Section 4.2).

Second, in order to obtain the intermediate resolutions of the reconstructed functions, we determine subsets of the unorganized point sets at all the inner nodes of the binary tree by using a thinning algorithm (Section 4.3). Then, we compute the local reconstruction functions in the leaf nodes and in the inner nodes by variational techniques with radial basis functions.

The evaluation of the global reconstruction function at the highest resolution is given by a partition of unity blending of the local reconstruction functions by pairs that is propagated bottom-up from the leaf nodes, and the root node contains the global reconstruction function. The evaluation at intermediate resolutions is obtained by selecting the appropriate precision at every branch of the binary tree and the bottom-up partition of unity blending of the local reconstruction functions by pairs (Section 4.4).

The particular charm of the new method is, that the local reconstructions are calculated from the same number of points in every local subdomain. Furthermore, the number of points in the *overlapping zone* can be specified explicitly, insuring an important constraint to stabilize the partition of unity blending between the local subdomains.

4.2 Binary tree domain decomposition into overlapping subdomains

Starting from the unorganized point set P and its bounding box being the entire domain Ω^0, the binary tree domain decomposition method adaptively subdivides Ω^0 using a binary tree. The tree is built in a top-down recursive process starting from the root node, where the entire domain Ω^0 is subdivided into two overlapping subdomains Ω_1^1 and Ω_2^1 containing an equal number of points n^1 in the respective point sets P_1^1 and P_2^1. All subdomains Ω^l at level l are themselves subdivided recursively into two overlapping subdomains Ω_1^{l+1} and Ω_2^{l+1} containing an equal number of points n^{l+1} in the respective point sets P_1^{l+1} and P_2^{l+1}. The recursion terminates, when the number of points in a subdomain falls below a threshold T_{leaf}. In this way, a *perfect binary tree* of level L is established, i.e. all leaf nodes are at the same level L in the tree, and all internal nodes have the degree two. Furthermore, in all 2^L leaf nodes, the equal number of points n^L is restricted to $T_{leaf}/2 < n^L \leq T_{leaf}$.

Let us now give some insight how to subdivide a domain Ω^l into two overlapping subdomains Ω_1^{l+1} and Ω_2^{l+1} containing an equal number of points n^{l+1} in the respective point sets P_1^{l+1} and P_2^{l+1}.

First, the number of points $n_o^l = \mathrm{Card}(P_1^{l+1} \cap P_2^{l+1})$ in the overlapping zone $\Omega_1^{l+1} \cap \Omega_2^{l+1}$ has to be specified explicitly as an *overlap quota* $q \in]0,1[$ of the number of points n^l:

$$n_o^l = qn^l \tag{11}$$

In order to avoid a too low or too high number of points in $\Omega_1^{l+1} \cap \Omega_2^{l+1}$ in the interior nodes of the binary tree resulting in either an undesired behavior when sticking together the functions of the local subdomains, or in a too high computational overhead, we clamp n_o^l to $[T_{min}, T_{max}]$. Then, the number of points n^{l+1} in the subdomains can be calculated as follows:

$$n^{l+1} = \left\lceil \frac{n_o^l + n^l}{2} \right\rceil \tag{12}$$

The extent of the two overlapping subdomains Ω_1^{l+1} and Ω_2^{l+1} is calculated by the following steps. First, the longest axis of Ω^l is determined. Second, we collect the point sets P_1^{l+1} (respectively P_2^{l+1}) containing the n^{l+1} points with the lowest (respectively highest) values with respect to the longest axis. In practice, we rearrange the points P^l with respect to their values of the longest axis. By setting $i_1 = n^{l+1}$ and $i_2 = n^l - n^{l+1} + 1$, we rearrange the points so that $\mathbf{p}_i < \mathbf{p}_{i_1}$ for $1 \leq \mathbf{p}_i \leq \mathbf{p}_{i_1}$ and $\mathbf{p}_i > \mathbf{p}_{i_2}$ for $\mathbf{p}_{i_2} \leq \mathbf{p}_i \leq \mathbf{p}_{n^l}$:

Finally, Ω_1^{l+1} and Ω_2^{l+1} are defined by the bounding boxes of P_1^{l+1} and P_2^{l+1}, respectively. Consider the resulting recursive algorithm for the binary tree domain decomposition shown in Algorithm 1 that has to be called using `Decompose(P, 0)`.

Algorithm 1 Decompose(P, l)

Require: points P^l, level l
Ensure: binary tree
  set Ω^l be the bounding box of P, set $n^l = \mathrm{Card}(P^l)$
  if $n^l > T_{leaf}$ **then**
    set $n_o^l = qn^l$
    clamp n_o^l to $[T_{min}, T_{max}]$
    determine the longest axis of Ω^l
    rearrange the points in P^l according to the longest axis
    determine the points P_1^{l+1} and P_2^{l+1}
    Decompose(P_1^{l+1}, l+1)
    Decompose(P_2^{l+1}, l+1)
  end if

Figure 2 illustrates the first levels of the resulting binary tree with the corresponding overlapping subdomains.

4.3 Multi-scale reconstruction by thinning

For the highest resolution reconstruction, the local reconstruction functions in all the leaf nodes of the perfect binary tree are calculated using variational techniques with radial basis functions. Recall, that this can be done efficiently since the equal number of points n^L in the leaf nodes is restricted to $T_{leaf}/2 < n^L \leq T_{leaf}$.

For the determination of the intermediate resolutions of the multi-scale reconstruction, we reconstruct local reconstruction functions in the inner nodes of the binary tree. To

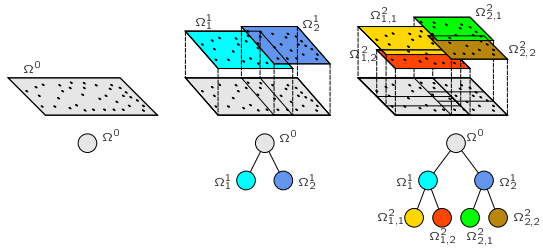


Figure 2. First steps of the binary tree domain decomposition into overlapping subdomains.

enable the use of variational techniques with radial basis functions, we recursively use a data thinning algorithm to reduce the number of points P' in the inner nodes to T_{leaf} points from the union of the points $P_1 \cup P_2$ in the two child nodes.

Since the thinning algorithm is applied recursively starting at the bottom of the binary tree, each thinning step requires only a small reduction from at most $2T_{leaf}$ to T_{leaf} constraints. Consequently, we can apply a near-optimal and rather costly thinning algorithm, and we decided to use the greedy thinning algorithm proposed by Iske [15]. It is based on the definition of the *separation distance*, i.e. the smallest distance between two points of the point set P

$$s_P = \min_{1 \le i < j \le \mathrm{Card}(P)} ||\mathbf{p}_i, \mathbf{p}_j||,$$

and the definition of a *removable point*, i.e. a point where the separation distance is attained:

$$\mathbf{r}_P \text{ is removable if } s_P = \min_{\mathbf{p}_i \in P} ||\mathbf{r}_P, \mathbf{p}_i||$$

In every iteration of Iske's algorithm, the separation distance of the point set is determined and the removable point is chosen and eliminated. Of course, the required number of iterations depends of the number of points to be removed (Algorithm 2).

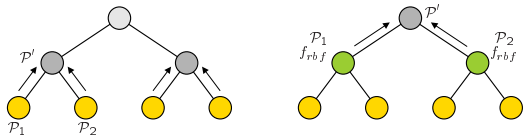
Algorithm 2 Thin(P, k)

Require: point set P, new point set size k,
Ensure: lower resolution pointset P'
  $P' = P$
  repeat
    find the separation distance $s_{P'}$
    choose one removable point $\mathbf{r}_{P'}$
    $P' = P' \setminus \{\mathbf{r}_{P'}\}$
  until Card(P') == k

Summing up, once the binary tree is set up as explained in the preceding section, the multi-scale reconstruction is determined by the Algorithm 3 that has to be called with `LowResolution(root node)`, and some steps of this algorithm are illustrated in Figure 3.

Algorithm 3 LowResolution(b)

Require: binary tree's node b
Ensure: node's low resolution reconstruction f_{rbf}
  if b is a leaf node **then**
    Calculate local RBF reconstruction for P^l
  else
    LowResolution(b's left child)
    LowResolution(b's right child)
    find low resolution constraints $P' = \mathrm{Thin}(P_1 \cup P_2, T_{leaf})$
    Calculate local RBF reconstruction of P'
  end if



(a) Simplification step

(b) Reconstruction of simplified point set.

Figure 3. Recursive creation of the multi-scale representation.

4.4 Adaptive multi-scale evaluation

For the evaluation of the global reconstruction function f at a point $\mathbf{p}$, the local reconstructions at the desired resolutions are blended together by recursively applying the partition of unity method bottom-up in the binary tree. In order to obtain the desired resolution, by starting from the root node, we decide at every node of the binary tree whether the RBF reconstruction f_{rbf} from the lower resolution point set is precise enough depending on an error condition ε, otherwise the reconstruction function f^l_{pou} at level l is given by the partition of unity blending of the local reconstruction functions of the two child nodes f_1^{l+1} and f_2^{l+1}:

$$f^l_{pou}(\mathbf{p}) = \frac{f_1^{l+1}(\mathbf{p})W_1^{l+1}(\mathbf{p}) + f_2^{l+1}(\mathbf{p})W_2^{l+1}(\mathbf{p})}{W_1^{l+1}(\mathbf{p}) + W_2^{l+1}(\mathbf{p})} \qquad (13)$$

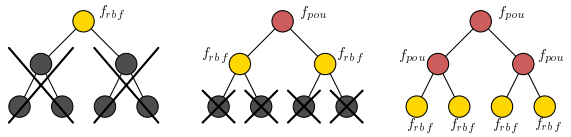
The weighting functions W_1^{l+1} and W_2^{l+1} for the local subdomains Ω_1^{l+1} and Ω_2^{l+1} are constructed as explained in Section 3.

Consider the recursive Algorithm 4 illustrating the adaptive multi-scale evaluation at a point $\mathbf{p}$ according to the error condition ε by applying the recursive partition of unity blending that has to be called by `Evaluate(p, 0, ε)`. Of course, for an evaluation at the highest resolution, the algorithm has to be called by `Evaluate(p, 0, false)`.

Depending on the application, different error conditions ε can be used, such as for example the size of the local subdomains, or the distance between the higher resolution point set and the lower resolution point set. Figure 4 illustrates an example of the multi-scale evaluation with the very simple error condition ε being a given level. In the yellow nodes,

Algorithm 4 Evaluate($\mathbf{p}$, l, ε)

Require: point $\mathbf{p}$, level l, error condition ε
Ensure:
  if $\mathbf{p}$ not in Ω^l **then**
    return 0
  end if
  if l is the leaf level **then**
    return $f_{rbf}(\mathbf{p})$
  else if error condition ε is true **then**
    return $f_{rbf}(\mathbf{p})$
  else
    f_1 = Evaluate($\mathbf{p}$, $l+1$, ε) at the left child
    f_2 = Evaluate($\mathbf{p}$, $l+1$, ε) at the right child
    compute W_1, W_2 from children
    return $f_{pou} = (f_1 W_1 + f_2 W_2)/(W_1 + W_2)$
  end if



(a) Level 0 (b) Level 1 (c) Level 2

Figure 4. Evaluation at a given level.

the RBF reconstruction f_{rbf} is evaluated, in the red ones the reconstruction f_{pou} is given by a partition of unity blending of the child nodes, and finally the gray nodes are not processed at all.

4.5 Scalability

Concerning the computational complexity, we are interested in the scalability of both the reconstruction of the global reconstruction function f starting from a point set P consisting of N points, and the scalability of the evaluation of the global reconstruction function f at a point $\mathbf{p} \in \mathbb{R}^3$.

The reconstruction of the global reconstruction function f involves the following principal steps:

1. Determine the domain Ω^l from the bounding box of the point set P^l and recursively subdivide it into overlapping domains after collecting the point sets P_1^{l+1} and P_2^{l+1}.

2. Compute the subsets of the unorganized point set in the inner nodes of the binary tree.

3. Compute the local reconstruction functions f^L for all local subdomains Ω^L in the 2^L leaf nodes and in the $2^L - 1$ inner nodes.

The first step is the recursive domain decomposition process. In every node of the binary tree, it is understood that the bounding box as well as the longest axis can be determined in linear time with respect to the number of points in the node. According to Sedgewick [27], the points can also be rearranged in linear time with respect to the longest axis. Since the number of points in the overlapping zone is bounded above by T_{max}, the number of points per node at every level is bounded, and hence the binary tree can be created in $O(N \log N)$. In the second step, the subsets of the points at every of the $2^L - 1$ inner nodes have to be calculated from the point sets of the two child nodes. Recall, that this is done in a bottom-up recursive process starting from the leaf nodes. Since the number of points in the leaf nodes is bounded by the constant T_{leaf}, and in the inner nodes we apply the thinning algorithm to obtain a subset with at most T_{leaf} out of $2T_{leaf}$ points, we consider that the computation of a subset can be done in constant time. Consequently, the computation of the subsets of the points at every of the $2^L - 1$ inner nodes can be done in $O(N)$. The third step involves the computation of the local RBF reconstructions in all 2^L leaf nodes and in the $2^L - 1$ inner nodes. Finding the local RBF reconstruction in one node is in $O(1)$, because the number of points n^L per leaf node is bounded by the constant T_{leaf}. Moreover, since the number of leaf nodes is proportional to N, all local RBF reconstructions can be determined in $O(N)$ time.

Concerning the evaluation of the global reconstruction function f at a point p, the following three steps are involved:

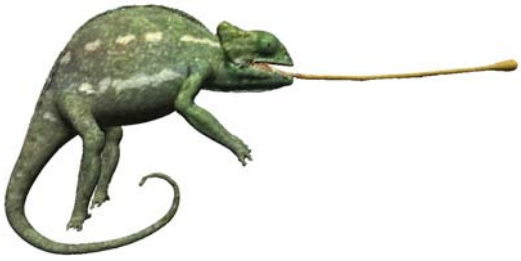
1. Find all local subdomains Ω^L with $p \in \Omega^L$.

2. Evaluate all local reconstruction functions f^L and the corresponding weighting functions W^L.

3. Propagate the evaluations of the local reconstruction functions bottom-up in the binary tree by using a partition of unity blending.

Since there are only a constant number of subdomains including the point p, the first and third step, i.e. traversing the binary tree from the root node and propagating the evaluations bottom up, require $O(\log N)$ time. The evaluation of one local reconstruction function can be done in constant time because the number of points is bounded by T_{leaf}, and since there is a constant number of concerned subdomains for p, step two can be done in $O(1)$.

Summing up, the implicit surface can be reconstructed in $O(N \log N)$ time, and the global reconstruction function f can be evaluated in $O(\log N)$ time. Consequently, our new multi-scale reconstruction method can be considered as efficient according to Schaback [26]. Note that we experienced a better scalability with our reconstruction method in practice. During the surface reconstruction, the binary tree creation in $O(N \log N)$ can be done very rapidly, and the results in Section 6 show an overall linear reconstruction and logarithmic evaluation time with respect to the number of points N.



(a) Surface and attributes reconstruction

(b) No distortions after applying a twist operator.

Figure 5. Chameleon (101 687 points). Reconstruction and deformation of the geometry and the texture.

5 Multi-scale reconstruction of implicit surfaces with attributes

5.1 Reconstruction of the implicit surface

In order to reconstruct an implicit surfaces from the set of N distinct points $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$ to be on the surface, we set up a general reconstruction problem. By using our new multi-scale reconstruction method presented in the preceding section, we want to determine the defining function of the implicit surface that is satisfying the constraints

$$f(\mathbf{p}_i) = 0 \quad i = 1 \ldots N. \tag{14}$$

It is easy to see that the constraints of equation (1) to determine the local RBF reconstruction functions with $h_1 = h_2 = \ldots = h_N = 0$ are not sufficient, because resolving the linear system of equation (4) yields the trivial solution $f(\mathbf{p}) = 0$. To overcome this problem, we introduce so-called *off-surface constraints* according to [28, 6] as an additional set of points $P' = \{\mathbf{p}'_1, \ldots, \mathbf{p}'_N\}$ where $f(\mathbf{p}'_i) = h'_i \neq 0$. The involved *off-surface points* $\mathbf{p}'_i$ are computed starting from the initial points $\mathbf{p}_i$ and moving them along the normal vector: $\mathbf{p}'_i = \mathbf{p}_i + \kappa \mathbf{n}_i$. The normal is usually obtained during data acquisition, however, when the normal is not available, it can be estimated from neighboring points [14].

5.2 Reconstruction of the attributes

We also use our new multi-scale reconstruction method to determine a continuous function for every attribute a_i of the points $\mathbf{p}_i$ in the unorganized point set P. We refer to attributes as a genus for ambient, diffuse and specular color channels, reflectance and transmittance coefficients, and others. We regard every attribute a_i for a point $\mathbf{p}_i \in P$ separately and reconstruct an attribute function f_a for every attribute that is satisfying the N constraints

$$f_a(\mathbf{p}_i) = a_i \qquad i = 1, \ldots, N. \tag{15}$$

Then, in order to determine the attributes of a point $\mathbf{p}$ on the reconstructed implicit surface, we simply have to evaluate the attribute functions in the point $\mathbf{p}$. For this reason, we consider our attribute functions as a procedurally defined solid texture, and of course, all characteristics of solid textures still apply to this procedurally defined solid texture. For example, since both the implicit surface as a 2D manifold in 3D and the solid texture are defined in $\mathbb{R}^3$, deformations applied to the surface and to the texture conserve the geometry-texture coherence and can hence be done without distortions as shown in Figure 5.

5.3 Visualization

To visualize the resulting implicit surfaces, a polygonizer such as the marching cubes algorithm [20, 33] can be used to create a polygonal mesh. Of course, we could also use Crespin's algorithm based on knowledge about initial points [8]. As other possible solutions, particle systems can be used to sample the resulting implicit surface [32, 13] and to polygonize it [9], or a mixed forward and backward warping technique [24] based on knowledge about the unorganized point set can be used. In order to account for global illumination, raytracing can be done and accelerated by taking into consideration the hierarchy of subdomains.

6 Results

All results presented in this section were performed on an Intel Pentium 1.7 GHz with 512 MB of RAM running Linux.

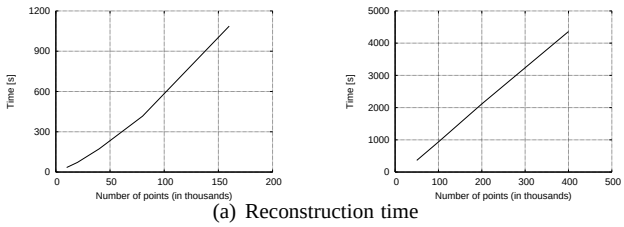
Table 1 presents the processing time depending on the initial number of constraints. In order to confirm the scalability of our reconstruction method, we reduced the initial unorganized point set by randomly choosing #points. We denote #points the number of points (with an additional off-surface points the total number is 2 #points), #leafs the number of leafs of the resulting perfect binary tree, #ppl the equal number of points per leaf, t_{tree} the binary tree creation time in seconds, t_{thin} the data thinning time in seconds, t_{rec} the local reconstruction time in seconds, and t_{poly} the polygonization time in seconds by using Bloomenthal polygonizer [5] with a resolution of 2% of the object's bounding box size. Since the polygonization process consists mainly of the evaluation

Model	#points	#leafs	#ppl	t_{tree}	t_{thin}	t_{rec}	t_{poly}
Isis	10 000	256	60	0.4	89	34	51
	20 000	512	63	1.0	181	72	55
	40 000	1024	66	2.5	373	173	62
	80 000	2048	69	6.0	756	416	70
	160 000	4096	72	14.4	1610	1087	80
Dragon	50 000	1024	81	3.3	390	362	253
	100 000	2048	85	8.6	793	935	285
	200 000	4096	89	18.7	1621	2114	325
	400 000	8192	93	43.2	3354	4360	345

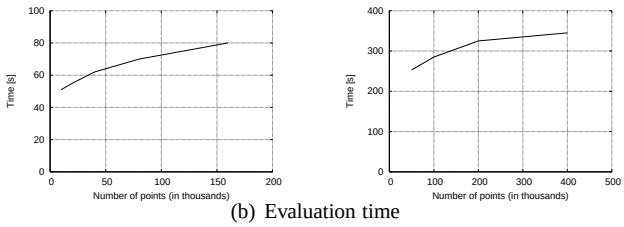
Table 1. Processing time (in seconds) of different 3D models.

of the global reconstruction function, we consider it representantive for the evaluation time. The reconstruction parameters were $q = 5\%$, $T_{leaf} = 100$, and a biharmonic basic function for ϕ.

For the illustration of the experienced linear complexity of the reconstruction time with respect to the number of points, Figure 6(a) shows the graph of the reconstruction time depending on the number of points. Moreover, Figure 6(b) shows the logarithmic complexity of the evaluation time.



(a) Reconstruction time

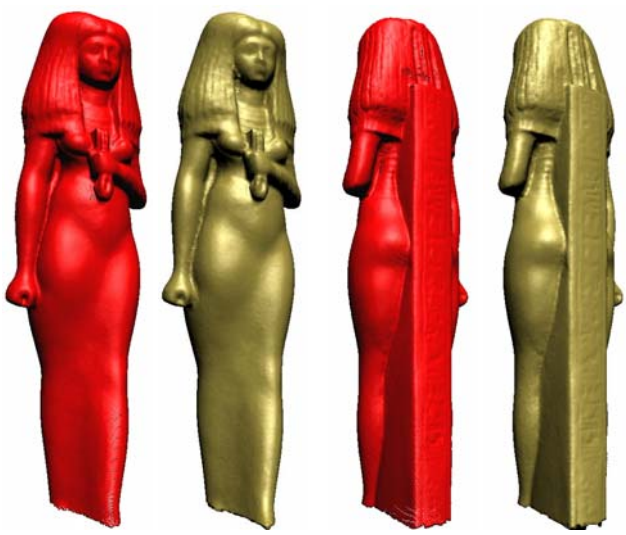


(b) Evaluation time

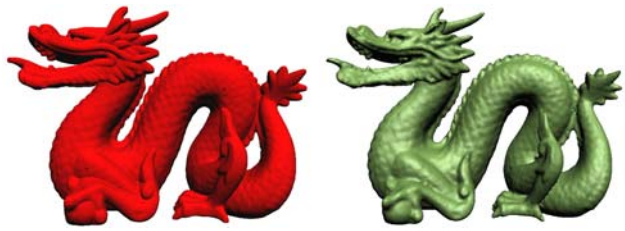
Figure 6. Reconstruction and evaluation time with respect to the number of points. Isis (left) and Dragon (right).

Some visual examples of the reconstruction quality are shown in Figure 7, and we compare a point-based rendering technique using opaque splatting of the unorganized points with the polygonal rendering of the reconstructed implicit surfaces. At the given screen resolution, there are no visible differences between the two methods.

Figure 8 illustrates the influence of the overlapping zone size. If the zone is too small (that is simulated with the small values of q and T_{min}), the implicit function has still the chosen continuity, but the surface appears not pleasant at overlapping zones.

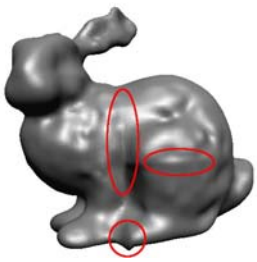


(a) Isis - 160 000 points



(b) Dragon - 200 000 points

Figure 7. Point rendering using splatting (at left, in red) and polygon rendering of the reconstructed surface at the highest resolution.



(a) $q = 1\%$, $T_{min} = 0$

(b) $q = 5\%$, $T_{min} = 5$

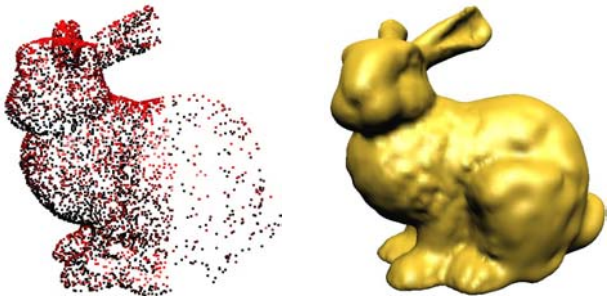
Figure 8. Influence of the overlapping zone size q on the reconstruction quality. $T_{leaf} = 50$.

Table 2 shows the quantitative results of the Stanford Bunny reconstruction with different parameters for T_{leaf} and q. Note that high values for T_{leaf} increase the reconstruction time significantly.

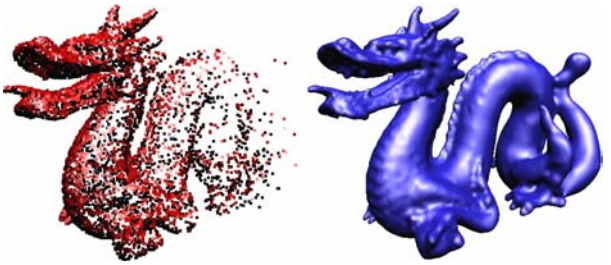
Figure 9 shows that even non-uniform, scattered data can be reconstructed robustly whether the density variation is

Model	T_{leaf}	q	#leafs	#ppl	t_{rec}
Bunny	50	1	1024	46	54
		2	1024	48	59
		5	2048	34	46
		10	4096	30	66
	100	1	512	81	171
		2	512	86	211
		5	1024	58	109
		10	1024	89	474
	200	1	256	152	633
		2	256	162	799
		5	512	106	417

Table 2. Bunny (35 947 points). Reconstruction time (in seconds) with respect to T_{leaf}, q



(a) Non-uniform dataset with hard density variation



(b) Non-uniform dataset with smooth density variation

Figure 9. Reconstruction from dataset of variable point density.

sharp (Figure 9(a)) or smooth (Figure 9(b)). The two models were built from the two original models by simply removing randomly chosen samples from the initial unorganized point sets.

We illustrate the results of the multi-scale reconstruction in two parts. The first one shows only the surface reconstruction, and the second one illustrates the attributes reconstruction as well. To evaluate the error of the low resolution reconstructions, we use the maximal and root-mean-square error of the residuals of the lower resolution point set compared to the point set P

$$e(\mathbf{p}_i) = h_i - f(\mathbf{p}_i)$$
$$e_{max}(P) = \max\{e(\mathbf{p}_i)\}$$
$$e_{rms}(P) = \sqrt{\frac{\sum_{i=1}^{N} e(\mathbf{p}_i)^2}{N}}$$

level	Igea (134 346)		Dragon (437 645)		Santa (75 782)	
	e_{max}	e_{rms}	e_{max}	e_{rms}	e_{max}	e_{rms}
0	2.490	0.447	2.497	0.533	1.660	0.395
1	2.387	0.422	2.975	0.536	1.635	0.330
2	2.341	0.347	2.415	0.415	1.446	0.298
3	2.340	0.309	1.821	0.346	1.397	0.232
4	1.730	0.232	1.716	0.308	1.299	0.170
5	1.493	0.173	1.340	0.248	0.913	0.114
6	1.168	0.125	1.270	0.193	0.660	0.073
7	0.998	0.086	1.146	0.147	0.541	0.046
8	0.680	0.054	1.143	0.110	0.445	0.029
9	0.603	0.035	1.138	0.080	0.256	0.018
10	0.510	0.022	1.113	0.074	0.000	0.000

Table 3. Reconstruction error at different multi-scale levels ($T_{leaf} = 100$, $q = 5\%$).

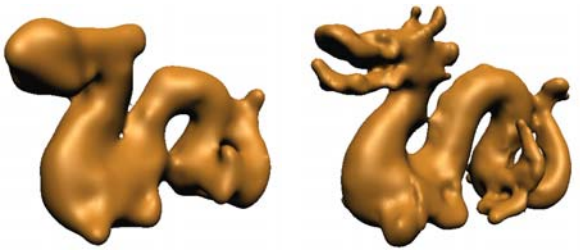
Table 3 shows the two error measures at 10 different levels compared to the highest resolution reconstruction for three models.

Figures 10 and 11 show the visual quality of our multi-scale reconstruction method at different levels. Note that even for a very complex geometry, the low resolution reconstruction maintains the topology well and smoothes out high frequencies.

Recall that the surface and attributes reconstructions can be evaluated at different resolution as illustrated in Figure 12. First, the highest resolution reconstruction from 148 138 points is presented for comparison, then the implicit surface is only reconstructed at level 3 (700 points) with 4 different reconstruction levels for the attributes.

7 Discussion

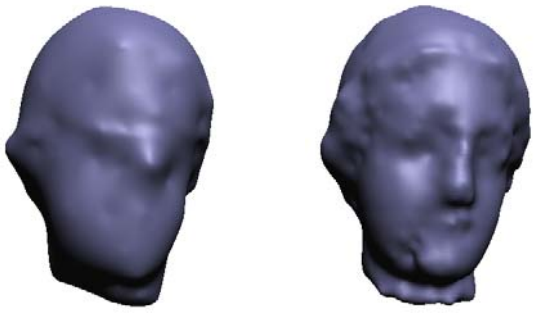
In this section, we evaluate the advantages and drawbacks of our new method compared to the existing multi-scale reconstruction methods for implicit surfaces by Ohtake et al. [23, 22] and to the state-of-the art reconstruction method of Carr et al. [6]. Note that Carr et al. do not explicitly define a multi-scale method, however, they introduce an algorithm to reduce the number of constraints that could be used to establish different resolutions. But similar to the method of Ohtake et al. [23], there will only be a small number of different levels. The multi-level partition of unity implicit method by Ohtake et al. [22] and our method allow to create a significantly higher number of levels that are created adaptively according to local surface characteristics. But note that our method is the only one that enables an adaptive



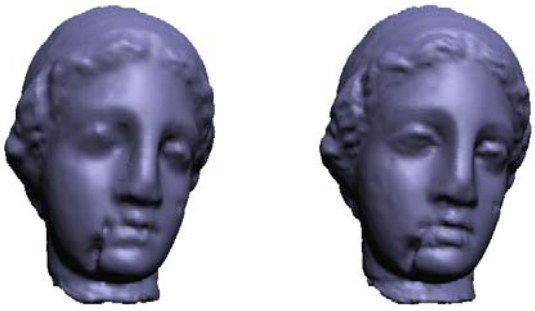
(a) Level 0 (80 points) (b) Level 3 (550 points)

(c) Level 6 (3770 points) (d) Level 9 (25850 points)

Figure 10. Multi-scale representation of the Stanford Dragon (437 645 points).



(a) level 1 (150 points) (b) level 4 (1 042 points)



(c) level 7 (7 150 points). (d) level 10 (49 050 points)

Figure 11. Multi-scale representation of Igea (134 346 points).

evaluation of the defining function of the implicit surface after being reconstructed. This is particularly useful for a view-dependent visualization of the implicit surface, where surface parts being closer to the viewpoint can be visualized more precisely compared to surface parts being farther away from the viewpoint.

Moreover, our reconstruction method enables both the approximation and interpolation of implicit surfaces from large unorganized point sets similar to the method of Carr [6] and [22], whereas the first method of Ohtake et al. [23] can only interpolate unorganized point sets.

Unfortunately, our method and the method of Carr et al. [6] require the introduction of off-surface constraints, and experimental results have shown a rather high reconstruction time compared to the methods of Ohtake et al. [23, 22]. However, they are the only methods enabling the reconstruction of continuous functions for the attributes.

Since in the first method by Ohtake et al. [23] the RBF reconstruction is accelerated using compactly supported radial basis functions (CSRBF), it cannot straightforwardly be used to reconstruct implicit surfaces from very large unorganized point sets due to the too large involved sparse matrix in the linear system to solve. Another drawback of this method is the rather slow evaluation of the defining function of the reconstructed implicit surface at the highest resolution, since the evaluation of the defining functions at all intermediate resolutions is involved.

The multi-level partition of unity method seems to be the only method to be able to reconstruct implicit surfaces while preserving sharp features [22]. Anyway, we are still not convinced that the partition of unity blending between sharp and smooth features of the surface can preserve sharp features without an extensive topology study. Moreover, we believe that the partition of unity blending of three different types of algebraic surfaces involved in this method is not as stable against strongly different topologies between the local reconstructions as by using exclusively local RBF reconstructions as in our new method.

An important advantage of Carr's [6] reconstruction method is, that the involved fast multipole method (FMM) accounts for the constraints more globally compared to simply blending together the local reconstructions. However, the far field expansion required by the FMM has to be done separately for every basis function in every dimension, and is very complex to implement. Moreover, like in the method of Ohtake et al. [23], the involved high computational effort to reconstruct a new implicit surface after small changes in the constraints makes its use difficult in implicit surface modeling applications compared to our method and the method of Ohtake et al. [22].

For the sake of convenience, we summarized the different characteristics of the four discussed reconstruction methods in Table 4.

8 Conclusions

In this paper, we described a new approach to reconstruct large scattered datasets by dividing the global reconstruc-

	Carr et al. [6]	Ohtake et al. [23]	Ohtake et al. [22]	Our method
reconstruction method	RBF	RBF	algebraic	RBF
acceleration technique	FMM	CSRBF	POU	POU
off-surface constraints	√	-	-	√
simple implementation	-	√	√	√
multi-scale	-	√	√	√
high number of levels	-	-	√	√
adaptive reconstruction	-	-	√	√
adaptive evaluation	-	-	-	√
approximation	√	-	√	√
interpolation	√	√	√	√
attributes reconstruction	√	-	-	√
very large point sets	√	-	√	√
fast reconstruction	-	√	√	-
fast evaluation	√	-	√	√
sharp edges	-	-	√	-

Table 4. Comparison of surface reconstruction techniques from unorganized points.

tion domain into smaller local subdomains, solving the reconstruction problems in the local subdomains with variational techniques using radial basis functions, and recursively blending the solutions together using the partition of unity method. A multi-scale reconstruction is obtained by using thinning algorithms to obtain subsets of the data for intermediate resolutions. Starting from these subsets, the local reconstructions are computed with variational techniques using radial basis functions as well.

In contrast to partition of unity implicits [22], where the local reconstructions of all the leaf nodes are blended together using the partition of unity method, we use the partition of unity method also for the inner nodes of the hierarchy. The local reconstructions at the desired precisions are blended together and propagated bottom-up, and the root node contains the global reconstruction function.

Our approach has a nice linear behavior with respect to the size of the dataset. Furthermore, the local reconstruction problems can be solved by various, non-communicating entities due to the independence of the local subdomains. The stability of the reconstruction obtained by the possibility to specify the number of points in the partition of unity blending and by using variational techniques with radial basis functions makes our approach robust against highly, non-uniformly distributed and topologically complex datasets allowing its usage in various application fields.

We applied our new method to the multi-scale reconstruction of implicit surfaces with attributes by reconstructing the defining function of the implicit surface and an attribute function for every attribute. It is particularly beneficial that different resolutions for the surface and the attributes can be selected adaptively for different applications according to local surface characteristics, viewing parameters, provided time, or other criteria.

Our new multi-scale reconstruction method intrigues us in various areas for current and future research. For example, we intend to show the power of our method in further applications, such as repairing 3D polygonal meshes and the reconstruction of missing information in medical data. Moreover, we are currently defining a new point-based rendering technique for our reconstructed implicit surface by using a curvature-driven sampling technique based on particle systems. Lower resolutions of the implicit surface can be used to rapidly spread particles evenly and coarsely over the surface before using the higher resolutions to increase the precision. Using this point-based rendering technique, a completely meshless interactive modeling environment by using points both as modeling and rendering primitive can be defined.

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3–15, January 2003.

[2] M. Alexa, J. Behr, D. Cohen-Or, D. Levin, S. Fleishman, and C. T. Silva. Point set surfaces. In *IEEE Visualization 2001*, pages 21–28. IEEE, 2001.

[3] R. Beatson and G. Newsam. Fast evaluation of radial basis functions. *Computational Mathematics and Applications*, 24(12):7–20, 1992.

[4] R. K. Beatson and W. A. Light. Fast evaluation of radial basis functions: methods for two-dimensional polyharmonic splines. *IMA Journal of Numerical Analysis*, 17(3):343–372, 1997.

[5] J. Bloomenthal. An implicit surface polygonizer. *Graphics Gems IV*, pages 324–349, 1994.

[6] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of ACM SIGGRAPH 2001*, pages 67–76, 2001.

[7] J. C. Carr, R. K. Beatson, B. C. McCallum, W. R. Fright, T. J. McLennan, and T. J. Mitchell. Smooth surface reconstruction from noisy range data. In *Proceedings of ACM Graphite 2003*, pages 119–126, 2003.

[8] B. Crespin. Dynamic triangulation of variational implicit surfaces using incremental delaunay tetrahedralization. In *Proceedings of the 2002 IEEE Symposium on Volume visualization and graphics*, pages 73–80, 2002.

[9] L. H. de Figueiredo, J. Gomes, D. Terzopoulos, and L. Velho. Physically-based methods for polygonization of implicit surfaces. In *Proceedings of Graphics Interface '92*, pages 250–257. CIPS, 1992.

[10] S. Fleishman, M. Alexa, D. Cohen-Or, and C. T. Silva. Progressive point set surfaces. *ACM Transactions on Computer Graphics*, 22(4), 2003.

[11] R. Franke. Scattered data interpolation: Tests of some methods. *Mathematics of Computation*, 38(157):181–200, 1982.

[12] R. Franke and G. Nielson. Smooth interpolation of large sets of scattered data. *International Journal for Numerical Methods in Engineering*, 15(11):1691–1704, 1980.

[13] J. C. Hart, E. Bachta, W. Jarosz, and T. Fleury. Using particles to sample and control more complex implicit surfaces.

(a) Full resolution reconstruction.

(c) Attributes level 3, 7, 11 and 15

Figure 12. Multi-scale reconstruction of Male (148 138 points). (a) Highest resolution. (b) Surface reconstruction at level 3.

In *Proceedings of Shape Modeling International 2002*, pages 129–136, 2002.

[14] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics (Proceedings of ACM SIGGRAPH 92)*, 26(2):71–78, July 1992.

[15] A. Iske. Reconstruction of smooth signals from irregular samples by using radial basis function approximation. In *Proceedings of the 1999 International Workshop on Sampling Theory and Applications*, pages 82–87, 1999.

[16] A. Iske. Scattered data modelling using radial basis functions. In A. Iske, E. Quak, and M. Floater, editors, *Tutorials on Multiresolution in Geometric Modelling*, pages 205–242. Springer, 2002.

[17] N. Kojekine, I. Hagiwara, and V. Savchenko. Software tools using CSRBFs for processing scattered data. *Computers & Graphics*, 27(2):311–319, 2003.

[18] D. Levin. The approximation power of moving least-squares. *Mathematics of Computation*, 67(224), 1998.

[19] D. Levin. Mesh-independent surface interpolation. In *4th International Conference on Curves and Surfaces*, page 46, 1999. Saint-Malo, France.

[20] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics (Proceedings of ACM SIGGRAPH 87)*, 21(4):163–169.

[21] B. S. Morse, T. S. Yoo, P. Rheingans, D. T. Chen, and K. R. Subramanian. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *Proceedings of Shape Modeling International*, 2001.

[22] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level partition of unity implicits. *ACM Transactions on Graphics (TOG)*, 22(3):463–470, 2003.

[23] Y. Ohtake, A. Belyaev, and H.-P. Seidel. Multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. In *Proceedings of Shape Modeling International*, 2003.

[24] P. Reuter, I. Tobor, C. Schlick, and S. Dedieu. Point-based modelling and rendering using radial basis functions. In *Proceedings of ACM Graphite 2003*, pages 111–118, 2003.

[25] V. V. Savchenko, A. Pasko, O. G. Okunev, and T. L. Kunii. Function representation of solids reconstructed from scattered surface points and contours. *Computer Graphics Forum*, 14(4):181–188, 1995.

[26] R. Schaback. Remarks on meshless local construction of surfaces. In R. Cipolla and R. Martin, editors, *The mathematics of surfaces IX*, pages 34–58. Springer, 2000.

[27] R. Sedgewick. *Algorithms in C*. Addison-Wesley, 1988.

[28] G. Turk and J. O'Brien. Variational implicit surfaces. Technical Report GIT-GVU-99-15, Georgia Institute of Technology, 1998.

[29] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, 4:389–396, 1995.

[30] H. Wendland. Fast evaluation of radial basis functions: Methods based on partition of unity. In C. K. Chui, L. L. Schumaker, and J. Stöckler, editors, *Approximation Theory X: Abstract and Classical Analysis*, pages 473–483. Vanderbilt University Press, Nashville, 2002.

[31] H. Wendland. Surface reconstruction from unorganized points. Preprint Gottingen, 2002.

[32] A. P. Witkin and P. S. Heckbert. Using particles to sample and control implicit surfaces. *Proceedings of ACM SIGGRAPH 94*, pages 269–278, 1994.

[33] B. Wyvill, C. McPheeters, and G. Wyvill. Data structure for soft objects. *The Visual Computer*, 2(4):227–234, 1986.