Efficient Reconstruction of Large Scattered Geometric Datasets using the Partition of Unity and Radial Basis Functions

Ireneusz Tobor
LaBRI - INRIA Futurs
Université Bordeaux 1
France
tobor@labri.fr

Patrick Reuter
LaBRI - INRIA Futurs
Université Bordeaux 1
France
preuter@labri.fr

Christophe Schlick
LaBRI - INRIA Futurs
Université Bordeaux 1
France
schlick@labri.fr

ABSTRACT

We present a new scheme for the reconstruction of large geometric data. It is based on the well-known *radial basis function* model combined with an adaptive spatial and functional subdivision associated with a family of functions forming a *partition of unity*. This combination offers robust and efficient solution to a great variety of 2D and 3D reconstruction problems, such as the reconstruction of implicit curves or surfaces with attributes starting from unorganized point sets, image or mesh repairing, shape morphing or shape deformation, etc. After having presented the theoretical background, the paper mainly focuses on implementation details and issues, as well as on applications and experimental results.

Keywords: radial basis functions, partition of unity, surface reconstruction, implicit modeling

1 MOTIVATION

In many applications fields, real-world datasets are often provided as a non-uniform, unorganized set of a large amount of discrete data. The main (and maybe the most difficult) problem to solve, in order to exploit these scattered data, is to efficiently and precisely reconstruct a continuous function starting from this dataset. In the field of geometric modeling, for instance, this problem has become of major importance due to the rapid development of 3D range scanners that acquire 3D geometries as an unstructured set of points.

Reconstructing a continuous function starting from unorganized data sets has been intensively studied over the last decades. Generally, the techniques can be divided into two major categories. The first category tries to generate a parametric function that interpolates or approximates the initial dataset. Piecewise linear approximation is the easiest and the most popular technique in this category. The second category interpolates or approximates the dataset by building a family of implicit real-valued scalar functions where the reconstructed domain is defined as the zero-set of them.

We will focus in this paper on the reconstruction of implicit surfaces without loss of generality. About 20 years ago, in an extensive survey, Franke [Frank82] identified radial basis functions (RBFs) as one of the most accurate and stable methods to solve scattered data interpolation problems. The pionneering work to interpolating surfaces using RBFs starting from unorganized point sets can be attributed to Savchenko et al. [Savch95] and Turk and O'Brien [Turk98]. 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 surfaces from large point sets consisting of more than several thousands of points.

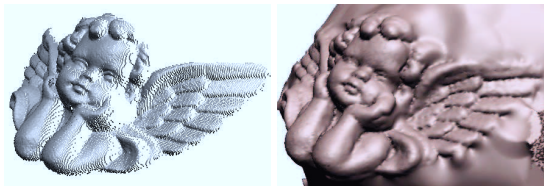


Figure 1: Example of surface reconstruction from unorganized points.

To overcome this problem, by using Wendland's compactly supported RBFs [Wendl95], Morse et al. [Morse01] showed how to reconstruct implicit surfaces from larger datasets since the involved linear system becomes sparse. This algorithm was further improved by Kojekine et al. [Kojek03] 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 non-uniform datasets where the density of the samples may vary significantly over the dataset. A multi-scale approach such as the one proposed by Ohtake et al. [Ohtak03b] overcomes this limitation, but it is not feasible for the approximation of noisy data. Anyway, for all techniques using compactly supported radial basis functions, the number of points that can be processed is still limited since the techniques remain global in nature.

A different approach to interpolate large point sets has

Journal of WSCG, Vol.12, No.1-3., ISSN 1213-6972
WSCG'2004, February 2-6, 2004, Plzen, Czech Republic

been proposed by Carr et al. [Carr01] based on a fast evaluation of RBF technique by Beatson et al. [Beats97, Beats92] using fast multipole methods. Unfortunately, the far field expansion has to be done 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 theorical survey [Wendl02a] and a more practical sketch [Wendl02b] on which we have based our approach.

Two other methods to reconstruct implicit surfaces without using radial basis functions have drawn much attention recently. In the first one, called Point Set Surfaces, presented by Alexa et al. [Alexa01, Alexa03] an implicit surface is reconstructed using a projection operator based on the method of moving least squares. The resulting implicit surfaces 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.

In the second one, called MPU implicits, developped recently by Ohtake et al. [Ohtak03a], the partition of unity method [Frank80] 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.

We combine in this paper two well-known methods in order to obtain a new reconstruction scheme for large datasets. RBFs are used to solve a set of small local problems 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 as the data does not need to be completely stored in memory, but can be accessed sequentially from disk.

Note that our presentation focuses more on geometric data, because it is the application field where there are the larger available datasets. But our technique is valid for any dimension discrete dataset. This is not the case for alternative reconstruction techniques such as Point Set Surfaces [Alexa01, Alexa03] or MPU [Ohtak03a], which are totally driven by the specific properties of 3D surface reconstruction. As an example of multidimensional reconstruction we present a technique that combines the reconstruction of geometric and colorimetric discrete data to generate an implicit surface with solid texture.

The paper is oganized as follows: Section 2 provides the theoretical background of our technique. In Section 3, we present our implementation and the practical choices to obtain an easy, efficient, and robust reconstruction algorithm. Section 4 presents some 3D applications and experimentals results, and in Section 5 we conclude and indicate some directions to future work.

2 THEORY

2.1 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 $f : \mathbb{R}^d \to \mathbb{R}$ with

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

In order to obtain a radial basis function (RBF) reconstruction of the point set P, a function f satisfying the equation

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

has to be found. We denote here $\|\mathbf{p}_i, \mathbf{p}_j\|$ the Euclidean distance, ω_i the weights, $\phi : \mathbb{R} \to \mathbb{R}$ a basis function, and π a polynomial of degree m depending on the choise of ϕ. $\pi(\mathbf{p}) = \sum c_i \pi_i(\mathbf{p})$ with $\{\pi_\alpha\}_{\alpha=1}^{Q}$ a basis in the d-dimensional null space containing all real-valued polynomials in d variables and of order at most m, hence $Q = \binom{m+d}{d}$.

The basis function ϕ has to be conditionally positive definite [Iske02], and some popular choices proposed in the literature are shown below:

$$\text{biharmonic} \quad \phi(r) = r \quad \text{with } \pi \text{ of degree 1} \tag{3}$$

$$\text{pseudo-cubic} \quad \phi(r) = r^3 \quad \text{with } \pi \text{ of degree 1} \tag{4}$$

$$\text{triharmonic} \quad \phi(r) = r^3 \quad \text{with } \pi \text{ of degree 2} \tag{5}$$

As we have an under-determined system with $N + Q$ unknowns (ω and $\mathbf{c}$) and N equations, so-called natural additional constraints for the coefficients ω are added, so that

$$\sum_i \omega_i c_1 = \sum_i \omega_i c_2 = \ldots = \sum_i \omega_i c_Q = 0. \tag{6}$$

The equations (1), (2), and (6) determine the following linear system:

$$A\mathbf{x} = \mathbf{b} \tag{7}$$

$$A = \begin{bmatrix} \Phi & P^T \\ P & 0 \end{bmatrix} \tag{8}$$

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

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

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

$$\mathbf{b} = [h_1, h_2, \ldots, h_N, \underbrace{0, 0, \ldots, 0}_{Q \text{ times}}]^T \tag{9}$$

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

If an approximation rather than an interpolation is required, one solution is to modify the diagonal of the matrix $\Phi \leftarrow \Phi - 8k\pi\rho\mathbb{I}$ (see Carr [Carr01, Carr03]), where the parameter ρ controls the fitting tolerance (i.e. the result is getting smoother when ρ is increased).

2.2 Partition of unity method

The main idea of the partition of unity (POU) approach is to divide the global domain of interest into smaller domains where the problem can be solved locally. More formally, the global difficult problem P is decomposed into several smaller local problems P_i and their local solutions S_i are combined together using the weighting coefficients of S_i that act as smooth "glueing" functions to obtain a global solution S.

Consider a global domain Ω and divide it into M "slightly" 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 $\text{supp}(w_i) \subseteq \Omega_i$ 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 solution is then defined as a combination of the local functions weighted by the partition functions w_i.

$$F(\mathbf{p}) = \sum_{i=1}^{M} f_i(\mathbf{p}) w_i(\mathbf{p}). \tag{10}$$

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

$$w_i(\mathbf{p}) \quad = \quad \frac{W_i(\mathbf{p})}{\sum_j W_j(\mathbf{p})}. \tag{11}$$

Any function W_i is appropriated, but to guarantee the continuity of the global interpolation function F, it has to be continuous at the boundary of the regions Ω_i.

2.3 Complexity analysis

The solution of the linear system (7) of size N requires $O(N^3)$ floating point operations and $O(N^2)$ core-memory cells. Thus it is clear that direct methods are not suitable for a number of constraints greater than several thousands.

In the partition of unity approach, Ω is divided into M subdomains. With a "good", quasi-uniform distribution, every Ω_i contains N/M constraints in average. The new solution requires $O(M(N/M)^3)$ operations and $O((N/M)^2)$ cells. As N/M can be considered as a constant, the reconstruction complexity is in $O(N)$, hence it is linear with respect to the number of constraints.

Another benefit using the POU method is in the evaluation of the interpolating function (2). The global RBF approach requires $O(N)$ operations for one single evaluation. Using the POU approach, two steps are required: first, finding all regions containing the point to evaluate, and second, evaluating the radial basis functions of the small local regions. Thus, the number of operations required is $O(M+N/M)$. Using an appropriate partitioning and data structure for fast neighbor searching like octree or kd-tree, we can reduce the first step to $O(\log N)$ or even $O(1)$, and with N/M considered as a constant, the evaluation complexity is $O(1)$.

3 RECONSTRUCTION SCHEME

3.1 Partitioning and local reconstruction

According to the equations (10) and (11) two families of functions have to be built: the weighting functions W_i and the local reconstruction functions f_i. Our reconstruction algorithm requires two steps: a space partition step that determines a set of overlapping domains $\{\Omega_i\}$ with associated weighting functions $\{W_i\}$, and a reconstruction step that computes the set of local functions $\{f_i\}$.

As we strive for an optimal reconstruction time, we have to obtain a quasi-uniform repartition of the points in the domains Ω_i. The recursive algorithm 1 describes an adaptive subdivision of the domain Ω_i, so that each subdomain contains between T_{min} and T_{max} points.

Algorithm 1 Partition(P, Ω_i)

Require: points P, domain Ω_i
Ensure: set of domains $\{\Omega_j\}$
  compute n number of points P
  if $n > T_{max}$ **then**
    subdivide Ω_i into overlapping $\Omega_i^1, \ldots, \Omega_i^k$
    Partition(P, Ω_i^1)
    ...
    Partition(P, Ω_i^m)
  else if $n < T_{min}$ **then**
    while $n \notin [T_{min}, T_{max}]$ **do**
      if $n < T_{min}$ **then**
        enlarge Ω_i
      else if $n > T_{max}$ **then**
        reduce Ω_i
      end if
    end while
    domain OK, add Ω_i to $\{\Omega_j\}$
  else
    domain OK, add Ω_i to $\{\Omega_j\}$
  end if

If we call this function with the bounding domain Ω of the point set P, the result is a set $\{\Omega_1, \ldots, \Omega_M\}$ where $\forall \Omega_i \; T_{min} \leq \text{Card}(P_i) \leq T_{max}$.

Starting from the domain Ω_j we divide it recursively into m overlapping domains $\Omega_j^1 \ldots \Omega_j^m$. When the number of points P_j^n in Ω_j^n domain is higher than T_{max} we continue with the recursive subdivision. If the number of points is smaller than T_{max}, the recursion is terminated and the current domain is added to the $\{\Omega_i\}$ set. However, if a domain Ω_i does not contain enough points, the local interpolation f_i can lead to unexpected results. Hence, when the number of points is smaller than T_{min}, we have to adjust Ω_i. The adjusting scheme is an iterative process when Ω_i is enlarged and reduced until the number of points P_i is between T_{min} and T_{max} as explained in the following subsection.

The second step is quite simply: for every domain Ω_i in the $\{\Omega_i\}$ set, a RBF reconstruction f_i is computed from the point set P_i as shown in section 2.1.

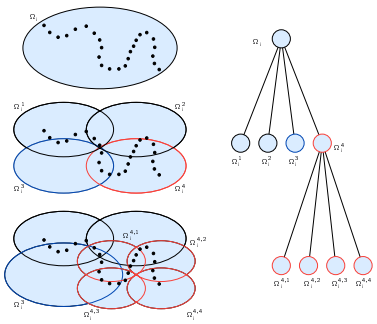


Figure 2: Space subdivision (Ω_i and Ω_i^4) and domain enlarging (Ω_i^3) in order to adaptively balance the number of points per domain.

Figure 2, shows a hierarchy of domains, created by our partitioning algorithm. The final $\{\Omega_i\}$ set contains all the leafs of the tree.

The T_{max} parameter controls time and stability. Increasing its value improves the local reconstruction at the cost of evaluation time. For the surface reconstruction, $T_{max} \in [120,200]$ is a reasonable threshold. We use also $T_{min} > 30$ that leads to a reconstruction without visible artifacts.

3.2 Operations on domains

The domains Ω_i can be of any shape, in practice, we use two simple and convex objects: axis-aligned ellipsoids of center C and axis A (Figure 3(a)), and axis-aligned bounding boxes defined from the two opposite corners S and T (Figure 3(a))

In order to enlarge or reduce a domain we simply scale it by a factor while keeping the domain center as shown Figure 3(b).

Figure 3(c) shows a subdivision step. In this stage a domain Ω is divided in eight (in the 3D case) equal-sized, overlapping subdomains based on a classical octree decomposition.

The table formalizes such operations according to S, T, C and A parameters. We denote the new parameters of the transformed domain with S', T', C' and A'.

	ellipsoid Ω_e	box Ω_b
enlarge or reduce	$A' = kA$ $C' = C$	$S' = S - \frac{(k-1)(T-S)}{2}$ $T' = T + \frac{(k-1)(T-S)}{2}$
subdivide	$A' = \frac{\sqrt{3}}{2} dA$ $C' = C - \frac{A}{2}$	$S' = S - \frac{(d-1)(T-S)}{4}$ $T' = \frac{S+T}{2} + \frac{(d-1)(T-S)}{4}$

The cofficient k denotes a *scale factor* in the iterative process of the enlarge/reduce scheme in order to adapt the domain size to the number of points. In practice, we take $k = 1.05$ for enlarge and $k = 0.98$ for reduce. The coefficient d is an *overlap factor*. A larger value for d yields larger overlapping zones and larger subdomains and hence a more stable reconstruction, but also increases the total number of domains and the reconstruction time ($d = 1.1$ is a good compromise).

3.3 Weighting function

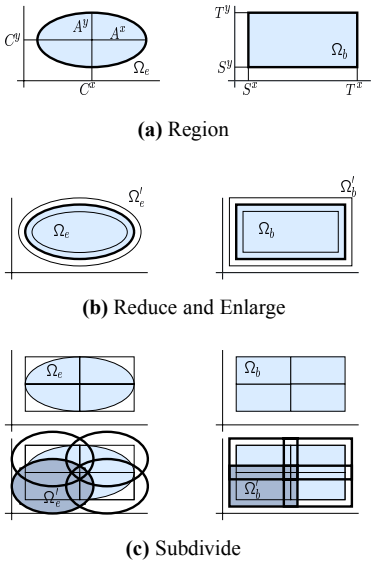
The choice of the weighting functions W_i determines the continuity between the local solutions f_i and the continuity of the global reconstruction function F. Our weighting functions W_i are defined as the composition of a distance function $D_i : \mathbb{R}^n \to [0,1]$, where $D_i(\mathbf{p}) = 1$ at the boundary of Ω_i and a decay function $V : [0,1] \to [0,1]$: $W_i(\mathbf{p}) = V \circ D_i(\mathbf{p})$.

We propose two formulations for the distance function D_i; D_i^b for a box and D_i^e for an ellipsoid:

$$D_i^b(\mathbf{p}) = 1 - \prod_{r \in x,y,z} \frac{4(\mathbf{p}^{(r)} - S^{(r)})(T^{(r)} - \mathbf{p}^{(r)})}{(T^{(r)} - S^{(r)})^2}$$

$$D_i^e(\mathbf{p}) = \sum_{r \in x,y,z} \frac{(\mathbf{p}^{(r)} - C^{(r)})^2}{(A^{(r)})^2}$$

Depending on the choice of the decay function V, a more or less smooth weighting function W_i is created with the desired continuity over the domain Ω_i. We suggest to use one of the following formulations for V that were chosen

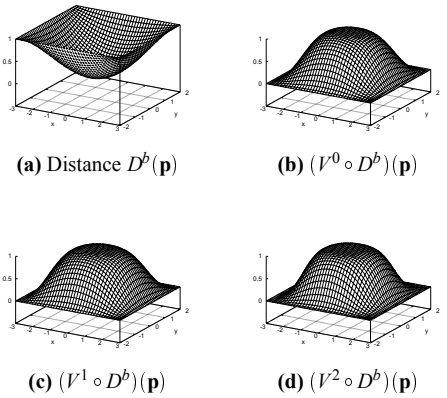


(a) Region

(b) Reduce and Enlarge

(c) Subdivide

Figure 3: Operations on the ellipsoid and box regions.



(a) Distance $D^b(\mathbf{p})$ **(b)** $(V^0 \circ D^b)(\mathbf{p})$

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

Figure 4: Distance function and weighting function for a rectangular region.

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.).

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

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

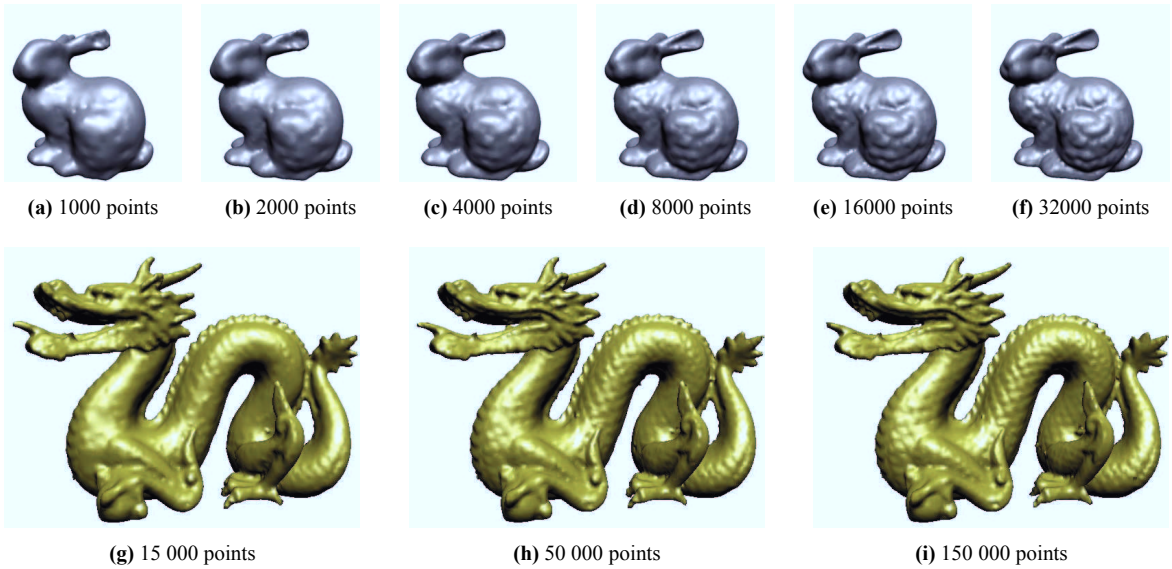
$$\mathbb{C}^2 : \quad V^2(d) = -6d^5 + 15d^4 - 10d^3 + 1$$

Figure 4 shows the geometric interpretation of these formules for a 2D rectangular domain.

3.4 Reconstruction constraints

When the set of values $\{h_1, \ldots h_k\}$ of the conditions (1) are distinct, we can solve various reconstruction problems using the technique described so far. However, when reconstructing implicit surfaces from a set of N distinct points $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$, we want to find a function satisfying

$$\forall i \; F(\mathbf{p}_i) = 0, \tag{12}$$



(a) 1000 points **(b)** 2000 points **(c)** 4000 points **(d)** 8000 points **(e)** 16000 points **(f)** 32000 points

(g) 15 000 points **(h)** 50 000 points **(i)** 150 000 points

Figure 5: Reconstruction of the Stanford Bunny and Dragon from different initial point sets.

and the conditions (1) are not sufficient because $h_1 = h_2 = \ldots = h_N = 0$.

Unfortunately, with these conditions, the system (7) has the trivial solution, the constant function $F(\mathbf{p}) = 0$, which is not useful. One common solution [Turk98, Carr01] is to use *off-surface points*: the constraints (12) are completed by an additional set of points $P' = \{\mathbf{p}'_1, \ldots, \mathbf{p}'_l\}$ where $F(\mathbf{p}'_i) = h_i \neq 0$. These off-surface points $\mathbf{p}'_i$ can be computed starting from the initial points $\mathbf{p}_j$ and moving them along the normal vector: $\mathbf{p}'_i = \mathbf{p}_j + k_j \mathbf{n}_j$. The normal is usually obtained during data acquisition, however, when the normal is not available, it can be estimated from neighboring points [Hoppe92].

Based on the common convention that $F(\mathbf{p}) > 0$ inside and $F(\mathbf{p}) < 0$ outside the surface, Turk and O'Brien [Turk98] add one off-surface point at the exterior of the surface ($k_j > 0, h_i = -1$) on every initial point. Carr et al. [Carr01] propose to add two new points on both sides of the surface ($k_j > 0, h_i = -1$ or $k_j < 0, h_i = -1$) for a subset of the initial points.

In practice, we found that taking a translation value k_i as 1% of the length of the bounding box is often, but not always, sufficient. Carr et al. [Carr01] give a simple condition to reconstruct surfaces without auto-intersections: for every off-surface point $\mathbf{p}'_i = \mathbf{p}_j + k_j \mathbf{n}_j$, the nearest surface point has to be the point $\mathbf{p}_j$ that it is derived of.

4 APPLICATIONS AND RESULTS

All results presented in this section were performed on an Intel Pentium 1.7 GHz with 512 MB of RAM running Linux. To solve the linear systems, we used the linear solver from the GNU Scientific Library package [GNU] based on LU-decomposition.

To visualise the resulting implicit surfaces, we used a polygonizer such as [Loren87, Bloom94] to create a polygonal mesh. We can even imagine a more topologically stable algorithm based on knowlegde about the initial points [Cresp02]. As other possible solutions, we can mention the oriented particles method [Witki94, Lomba95] and a mixed forward and backward warping method [Reute03].

Our first application is the reconstruction of implicit surfaces starting from unorganized point sets. Table 1 presents the processing times depending on the initial number of constraints.

We denote #P the number of points (with two additional off-surface points the total number of constraints is 3#P), M the number of regions, t_{rec} the geometry reconstruction time in seconds, t_{poly} the polygonization time in seconds using Bloomenthal polygonizer [Bloom94] with a resolution of 2% of object's bounding box size, and e_{RMS} the RMS error of the reconstructed surface.

Model	#P	M	t_{rec}	e_{RMS}	t_{poly}
Bunny	1 000	50	6	0.19	126
	2 000	157	17	0.17	182
	4 000	223	24	0.13	134
	8 000	676	73	0.10	218
	16 000	946	106	0.07	170
	32 000	2 577	329	0.04	288
Buddha	25 000	1 814	143	0.15	445
	50 000	3 553	369	0.12	701
	100 000	7 431	717	0.10	1 020
	150 000	10 126	1 216	0.08	1 400
	300 000	22 285	2 652	0.02	2 354
	500 000	33 745	5 254	0.005	4 064

Table 1: Processing time and error of different models.

The initial models Bunny, Happy Buddha and Dragon borrowed from Stanford 3D Scanning Repository [Lar] were downsampled and the complete process consisting of reconstruction and polygonizing was applied. At the same time we computed the RMS error of the downsampled constraints set compared to the initial point set. We can confirm the linear complexity of our reconstruction process, whereas usual polygonal reconstruction (based on Delaunay triangulation for instance) is $O(N \log N)$ at best. Note that the global error decreases proportionally to the number of points as it is expected. Figures 1 and 5

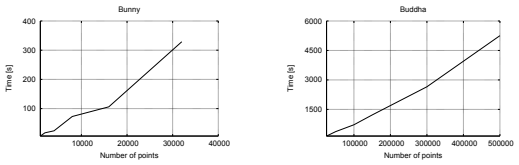


Figure 6: Reconstruction time in function of number of points. Bunny and Buddha. See also Table 1

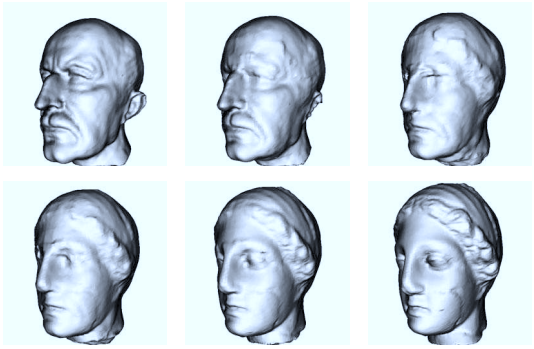


Figure 7: Linear morphing between Max Planck head and Igea.

show the examples of the reconstruction quality.

The influence of the two thresholds T_{min} and T_{max} is shown in Table 2. The same 8000 points model of the Dragon [Lar] was processed with different parameters. Visual results have shown that increasing T_{max} above a threshold does not influence the reconstruction quality but only increases the processing time, whereas a too small T_{min} can lead to visible artefacts.

T_{min}	T_{max}	M	t_{rec}	t_{poly}
20	50	1474	23	93
50	100	638	82	135
50	200	434	104	143
50	300	251	226	187
50	400	195	364	237
100	300	251	283	253
100	400	195	451	309

Table 2: Influence of T_{min} and T_{max}.

All characteristics of implicit surfaces that are reconstructed using our implementation still apply, such as function-based shape modelling [Pasko95]. As an example, we show in Figure 7 the linear morphing from the Max Planck head to the Igea head.

The second application is the possibility to combine geometric reconstruction with the reconstruction of additional surface attributes such as color channels, reflectance, and others. Such attributes can be considered as an intersection of the surface and a 3D procedural solid texture, where each attribute is reconstructed separately. Note that the number of constraints to reconstruct the implicit surface can be different from the number of constraints to reconstruct the attributes (Figure 8). As said, this is a unique feature of our RBF/POU reconstruction technique compared to alternative technique such as Point Set Surfaces or MPU that can only handle geometric constraints.

A nice characteristic is, that the same deformation function applied to the implicit surface and the attributes conserves the geometry and texture coherence. As an example Figure 9 shows a Chameleon model twisted along the z axis.

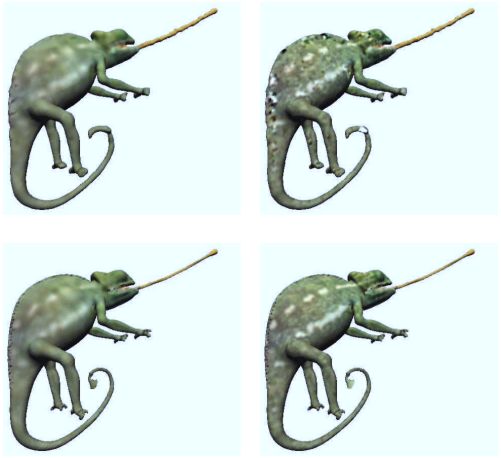


Figure 8: Independent reconstruction of geometry and texture. First row: surface reconstruction from 10 000 points, second row: from 100 000 points. First column: texture reconstruction from 10 000 points, second column: from 100 000 points.

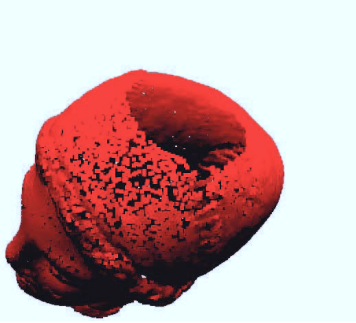
Table 3 presents the reconstruction time for the geometry (t_{geo}) and the texture (t_{tex}) in seconds, and also shows the memory usage (in MB) during reconstruction. As we can see, the memory peak has a linear complexity what confirms a theoretically limited memory due to local reconstruction stages.

Model	#P	M	t_{geo}	t_{tex}	Mem
King	2 500	271	9	2	7
	10 000	809	49	10	10
	40 000	2 595	222	55	22
Chameleon	25 000	2 228	109	39	17
	50 000	3 671	272	77	28
	75 000	7 203	352	148	42
	100 000	9 129	586	183	58

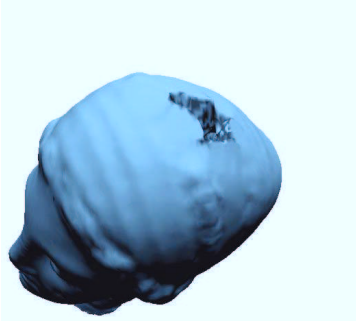
Table 3: Geometry and texture reconstruction. Memory peak.

Mesh repairing is another common use of 3D reconstruction. As partition of unity is a purely local method, meshes with very large holes may be incorrectly repaired, so we propose a simple semi-automatic process where the user has to specify additional subdomains that include the hole boundary. Figure 10 shows that it is sufficient to manually add one single region containing the boundary of the hole to repair the mesh as expected.

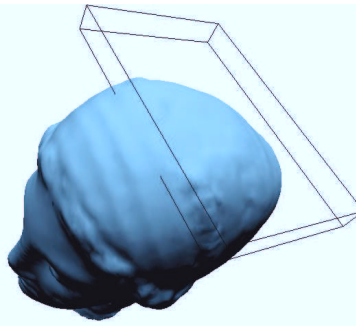
The following results show additional features of the surface reconstruction. The difference between interpolation and approximation is shown at Figure 11. The same 10 000 points model of the Stanford Bunny was reconstructed with a different fitting tolerance ρ. Finally, Figure 12 shows that non-uniform, scattered data can be very



(a) Initial point set from incomplete mesh



(b) Incomplete reconstruction. The hole is too large.



(c) Correct reconstruction with user interaction

Figure 10: Mesh repairing.

Figure 9: Twisted chameleon - simultaneous transformation of reconstructed geometry and texture.

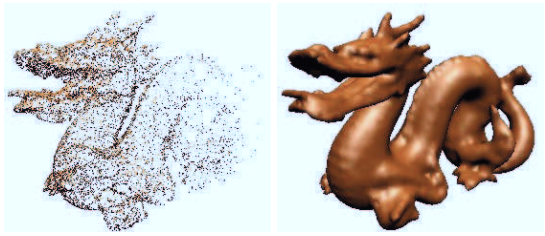
(a) Interpolation.

(b) Approximation. Fitting tolerance $\rho = 10^{-6}$

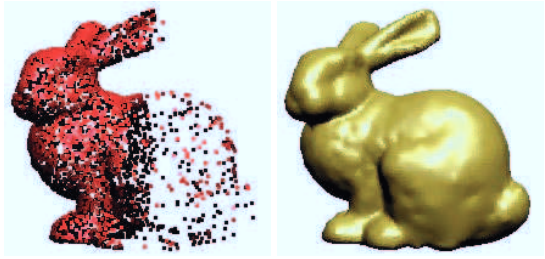
Figure 11: Interpolation and approximation of point set.

robustly reconstructed whether the density variation is smooth (Figure 12(a)) or sharp (Figure 12(b)).

Another interesting application domain is volume reconstruction for medical images. Medical images are often acquired as a set of slices distanced of some millimeters. A difficult problem is to connect the slices by reconstructing the missing information between. We obtained convincing results (not presented here) simply by considering the raw data as a sampling of a 3D signal function and then applying the reconstruction scheme on it.



(a) Non-uniform dataset with smooth density variation



(b) Non-uniform dataset with sharp density variation

Figure 12: Variable point density test.

5 CONCLUSION AND FUTURE WORKS

We described a new approach to reconstruct large geometric datasets by dividing the global reconstruction domain into smaller local subdomains, solving the reconstruction problems in the local subdomains using radial basis functions with global support, and combining the solutions together using the partition of unity method. Our approach has a nice behaviour 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. Moreover, the stability of the reconstruction using radial basis functions makes our approach robust against highly, non-unifomly distributed and topologically complex datasets allowing its usage in various application fields.

We showed the quality of our approach on a variety of examples in different domains, and the quantitative re-

sults confirmed our expectation of the linear complexity behaviour. We think that the simplicity of the described process combined with the practical implementation issues given in this paper makes our approach highly accessible.

Our new approach intrigues us in various areas for current and future resarch. For example, the hierarchy of regions has useful information only into its leafs. We are currently investigating how the recursive domain decomposition method can be exploited in order to define a multiresolution representation that can be used not only for progressive reconstruction, but also for level-of-detail evaluation and visualization.

Moreover, we are currently exploiting the locality of our reconstruction scheme to define a point-based modeling environment in order to improve previous work by Turk et al. [Turk02] and ourselves [Reute03]. Not only the locality of the reconstruction process, but also the constant evaluation time of the reconstruction function makes our new approach attractive for interactive modelling applications. Herein, we are also using our approach to create 3D procedural textures from the attributes of the non-uniformly distributed points.

REFERENCES

[Alexa01] Marc Alexa, Johannes Behr, Daniel Cohen-Or, David Levin, Shachar Fleishman, and Claudio T. Silva. Point set surfaces. In *IEEE Visualization 2001*, pages 21–28. IEEE, October 2001.

[Alexa03] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3–15, January 2003.

[Beats92] R. Beatson and G. Newsam. Fast evaluation of radial basis functions. *Computational Mathematics and Applications*, 24(12):7–20, 1992.

[Beats97] 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.

[Bloom94] Jules Bloomenthal. An implicit surface polygonizer. *Graphics Gems IV*, pages 324–349, 1994.

[Carr01] Jonathan C. Carr, Richard K. Beatson, Jon B. Cherrie, Tim J. Mitchell, W. Richard Fright, Bruce C. McCallum, and Tim R. Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of ACM SIGGRAPH 2001*, pages 67–76, 2001.

[Carr03] 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 Graphite 2003*, pages 119–126, 2003.

[Cresp02] Benot 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.

[Frank80] Richard Franke and Greg Nielson. Smooth interpolation of large sets of scattered data. *International Journal for Numerical Methods in Engineering*, 15(11):1691–1704, 1980.

[Frank82] Richard Franke. Scattered data interpolation: Tests of some methods. *Mathematics of Computation*, 38(157):181–200, 1982.

[GNU] GNU Scientific Library. `http://www.gnu.org/software/gsl/`.

[Hoppe92] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics (Proceedings of ACM SIGGRAPH 92)*, 26(2):71–78, July 1992.

[Iske02] Armin Iske. Scattered data modelling using radial basis functions. In Armin Iske, Ewald Quak, and Michael Floater, editors, *Tutorials on Multiresolution in Geometric Modelling*, pages 205–242. Springer, 2002.

[Kojek03] Nikita Kojekine, Ichiro Hagiwara, and Vladimir Savchenko. Software tools using CSRBFs for processing scattered data. *Computers & Graphics*, 27(2):311–319, 2003.

[Lar] Georgia Large Geometric Models Archive. `http://www.cc.gatech.edu/projects/large_models/`.

[Lomba95] Jean-Christophe Lombardo and Claude Puech. Oriented particles: A tool for shape memory objects modelling. In *Graphics Interface '95*, pages 255–262, 1995.

[Loren87] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics (Proceedings of ACM SIGGRAPH 87)*, 21(4):163–169, 1987.

[Morse01] Bryan S. Morse, Terry S. Yoo, Penny Rheingans, David 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.

[Ohtak03a] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Transactions on Graphics (TOG)*, 22(3):463–470, 2003.

[Ohtak03b] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. Multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. In *Proceedings of Shape Modeling International*, 2003.

[Pasko95] Alexander Pasko, Valery Adzhiev, Alexei Sourin, and Vladimir V. Savchenko. Function representation in geometric modelling: concept, implementation and applications. *The Visual Computer*, 11(8):429–446, 1995.

[Reute03] Patrick Reuter, Ireneusz Tobor, Christophe Schlick, and Sebastien Dedieu. Point-based modelling and rendering using radial basis functions. In *Proceedings of Graphite 2003*, pages 111–118, 2003.

[Savch95] Vladimir V. Savchenko, Alexander Pasko, Oleg G. Okunev, and Tosiyasu L. Kunii. Function representation of solids reconstructed from scattered surface points and contours. *Computer Graphics Forum*, 14(4):181–188, 1995.

[Turk98] Greg Turk and James O'Brien. Variational implicit surfaces. Technical Report GIT-GVU-99-15, Georgia Institute of Technology, 1998.

[Turk02] Greg Turk and James F. O'brien. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics*, 21(4):855–873, 2002.

[Wendl95] Holger Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, 4:389–396, 1995.

[Wendl02a] Holger 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.

[Wendl02b] Holger Wendland. Surface reconstruction from unorganized points. Preprint Gottingen, 2002.

[Witki94] Andrew P. Witkin and Paul S. Heckbert. Using particles to sample and control implicit surfaces. *Proceedings of ACM SIGGRAPH 94*, pages 269–278, 1994.