Abstract—The level set method can implement a wide variety of shape modeling operations (e.g. offsetting, skeletonization, morphing, blending, smoothing, sharpening, embossing, denoising, sculpting, growing, texturing and fitting) simply by specifying a corresponding speed function that controls the growth of an evolving voxel isosurface. The problem is that the basic level set method is implemented on a fixed resolution grid, which limits the utility of these shape modeling operations. We instead represent surfaces with a collection of radial basis function dipole pairs, and derive the motion of these dipoles to implement a surface propagation similar to the level set method but on a smooth, arbitrary resolution model. We demonstrate the utility of this approach with new level set methods for surface fitting, blending and center redistribution for RBF dipole models.

Keywords—implicit surface; level set; radial basis function; particle system; surface fitting

1. INTRODUCTION

The level set method evolves a surface by growing it in the direction of its normals, where the rate of this growth is controlled by a “speed” function defined over the surface. Though it was originally designed for numerical PDE applications [1], it has also been shown to be a single tool that can support a wide array of shape modeling tools, including:

- offsetting [2],
- skeletonization [3],
- morphing [4], [5],
- blending, smoothing, sharpening, morphology and embossing [6],
- surface denoising [7],
- sculpting [8],
- growing [9],
- geometric texturing [10], and
- fitting [11], [12], [13], [14].

The level set method relies on the representation of shape as an isosurface of a 3-D rectilinear grid of scalar voxel values. Solving a PDE over the grid links the desired surface propagation speed to the changes in voxel values needed to move the isosurface so. This representation is numerically robust and represents free-form shapes to the detail allowed by the grid resolution, but it also limits the usefulness of this approach for geometric modeling. The fixed resolution of the grid causes blocky artifacts in fine shape features. Increasing the effective grid resolution can reduce these artifacts, but make the method either more expensive or more complicated to process. The fixed resolution of the grid also limits the method’s application on smooth, man-made objects especially for computer-aided geometric design and production image synthesis.

This paper adapts the level set method to work on smooth surfaces defined by dipoles of radial basis functions. Given the desired surface motion specified by a speed function, we derive the motion of the RBF dipoles needed to obtain the corresponding implicit surface motion. The main advantage of this RBF dipole level surface formulation is that it can represent features at a variety of scales, whereas the classical level set method operates at a fixed voxel grid resolution. Voxel isosurfaces can be blocky and creased, even when a piecewise cubic interpolation method is used, whereas RBF surfaces are analytic ($C^\infty$).

The new RBF dipole level set method is implemented on top of the particle system modeling interface of Witkin and Heckbert [15], and uses a surface constrained particle system to sample the speed function across the entire implicit surface. The classical level set method uses the narrow band method [16] to restrict computation to fixed-resolution voxels neighboring the isosurface, whereas our approach distributes particles directly on the isosurface in densities sensitive to curvature and speed function variation.

A long standing challenge of the RBF surface representation has been the number and location of radial basis functions needed to represent a shape. Existing approaches [17], [18] distribute RBF centers in a space octree about the shape, inserting and deleting centers based on their effect on shape fidelity. The incorporation of surface particles into our RBF dipole level set method allows the particle population dynamics to manage RBF dipoles over the isosurface. In addition to inserting dipoles where they are needed and deleting dipoles where they are redundant, this formulation can also move dipoles to where they can be more effective.

Sec. II reviews previous work, showing the novelty of dipole RBF propagation not only as a surface modeling and fitting method but also an alternative to the level set method. Sec. III reviews the formulation of radial
basis function surfaces and their dipole representation. Sec. IV derives the motion of a RBF dipole surface propagating along its normals governed by a speed function, through differential geometry derivations of the motion of the normals needed to reorient the dipoles. Sec. VI demonstrates the method by using it to compactly fit RBF surfaces to point data and to the blending and smoothing of RBF models of surfaces.

2. Previous Work

2.1 Implicit Surface Fitting

Data-centric implicit formulations (including radial basis functions) arose specifically for the purpose of reconstructing surfaces from scattered point data. Shepard’s method [19], [20] fit point data by solving for the scales of functions of the distance to the data points, and this approach led to modern surface reconstruction techniques based on radial basis functions [21], [18] and partitions of unity [22], and like techniques based on distances to tangent planes [23] most popularly moving least squares [24]. Shen et al. [25] similarly used the potential of polygons to implicitize and repair large meshes. These techniques produce an implicit model tied to the input data and reconstruction method, whereas the formulation we describe evolves these surfaces into place according to a speed function.

McMahon and Franke [17] studied the problem of knot (center) selection for surface reconstruction, which strives to represent these surfaces using a smaller number of “knot” points rather than the entire collection of data points. For surface meshes, this problem is simplification [26]. McMahon and Franke proposed a simple k-means clustering of the data points, which targeted knot distribution but not necessarily reconstruction quality. Since the coefficients of most data-dependent basis function reconstructions are determined by solving a linear system, interpolation quality cannot be differentiated by the parameters, which hinders gradient descent and other quality optimizations. Hence reconstruction quality is commonly improved by organizing knots in a cluster hierarchy (sometimes simply an octree over a grid) such that knots can be subdivided in regions of high approximation error [18]. The proposed RBF surface evolution can actually move knots to improve surface approximation, though the motion occurs in the normal direction with slight tangential motion occurring due to the evolution of surface normal directions.

The parameter motion and primitive adaptivity used by the proposed RBF surface evolution works similarly to another one proposed for metaballs [27]. An RBF active contour model [28] evolved the motion of the centers of a single-center RBF surface according to a minimal energy model for fitting and registering slices of scanned objects, which expanded and cited an earlier manuscript version of this paper [29], and has been further expanded for compact-support RBF’s [30] and anisotropic RBF’s [31]. A recent topology optimization application [32] solves for the RBF coefficients of fixed centers necessary for a desired level set propagation whereas we solve for dipole center motion by finding the normal direction. They detect and locally over-smooth shocks whereas we use particle population dynamics to handle such cases.

2.2 The Level Set Method

The image processing, computer vision and medical imaging communities often employ active contours in 2-D and active surfaces in 3-D for segmentation [33]. They categorize these active contours and surfaces into two families: parametric, based on the Lagrangian dynamics of snakes [34], and geometric, based on the Eulerian level set method [1]. Radial basis function level set methods support the free-form topology of implicit surfaces and Eulerian methods, but with the smaller storage and computation requirements of surface-based Lagrangian methods.

The performance of the Eulerian level set method can be improved from space-order to surface-order by the fast marching method (for speed functions that never change sign) and the narrow band method [35]. The storage needed can be decreased by hierarchical representations (and accompanying integration methods) with quadtrees [36], [37] and octrees [38], [39], and by sophisticated storage and compression schemes including paging and vector quantization [40], RLE encoding [41] and out-of-core processing [42]. These techniques increase the effective resolution of the level set method to better adapt to fine surface features, but the resulting output is nevertheless constrained to grid resolution.

Propagating surface meshes would better preserve fine features for surface modeling, though Osher and Sethian [35] make strong arguments against their use due to topology changes and numerical instability near shocks. Lawrence and Funkhouser [9] managed to propagate a dynamic polygonal mesh [43] but had to limit their demonstration to simple cases that avoided shocks and topology changes. Semi-Lagrangian methods [44], [45] can robustly propagate meshed 2-D curves across topology changes by tracing vertices on a Lagrangian path through time, but using the Eulerian grid to compute function values along their path. Similar methods were used to propagate isolated water and air particles to improve the precision of an Eulerian air-water interface [46], [47]. Clever smoothing and resampling during each surface mesh propagation step can robustly propagate shocks and collision detection and a repolygonization response can manage topoggy changes [48], [49].

3. Background

A radial basis function (RBF) is a radially symmetric real function \(C(||x - x_i||)\) about a “center” point \(x_i\). Given a collection of center points \(x^*\) and desired function values \(F^*\), RBFs can be used to define the function

\[
F(x) = n \cdot x + d + \sum_{i=1}^{n} a_i C(||x_i - x||) \tag{1}
\]
where \( \mathbf{n} \cdot \mathbf{x} + d \) is a plane equation for regularization and \( a_i \) are unknown scales of the radial basis functions. These \( n+4 \) unknowns can be found by solving the linear system

\[
\begin{bmatrix}
C_{ij} & x_j^T & 1 \\
x_j & 0 & 0 \\
1 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
a_i \\
a \\
d
\end{bmatrix}
= \begin{bmatrix}
F_i \\
F_r \\
0
\end{bmatrix}
\tag{2}
\]

where \( C_{ij} = C(||x_i - x_j||) \). In 2-D the RBF \( C(r) = r^2 \log r^2 \) results in a thin plate spline that minimizes the bending energy of the interpolant, and its 3-D analog \( C(r) = r^3 \) minimizes the bending energy of the field (but not necessarily that of the isosurface \( F^{-1}(0) \)).

Radial-basis-function representations must have at least one center off the isosurface to avoid trivial (constant) solutions to (2). One strategy is to place single RBF center set to an “inside” value in the middle of an object, but it is sometimes difficult to determine the middle of a contorted object. Another strategy is to surround the object with RBF centers set to an “outside” value, but these values can sometimes interfere with the shape of the RBF surface.

We want to avoid a global specification of inside and outside. As the surface evolves, its topology and extent may change, and any global inside or outside constraints may become invalid or interfere with the propagation. We thus use RBF dipoles \([50]\) consisting of pairs of centers at \( x_i \pm \epsilon \mathbf{n}_i \) with values \( \pm \epsilon \), where \( \mathbf{n}_i \) is the desired surface normal at \( x_i \). In addition to overcoming the previous problems, RBF dipoles have the additional benefit of exerting control over the local orientation of the RBF surface.

The RBF dipole surface model not only interpolates surface points, it also specifies the desired surface normal at these surface points. While the surface normal of the resulting implicith surface interpolated from these RBF dipoles may differ from the intended dipole normal direction (the offset direction between the two RBF centers of the dipole), these differences are in practice not significant. We thus propagate the RBF dipole model by moving each dipole in its desired normal direction. However, this motion will update the surface and different growth rates should cause some portions of the surface to change orientation. Hence we must update the orientation of the dipole, as derived in the next section.

4. RADIAL BASIS FUNCTION PROPAGATION

The propagation of data-centric implicit formulations designed for surface reconstruction are problematic because the parameters for such models are defined by either linear system solution or an iterative process on the data points. Thus there is no closed-form derivation of the necessary motion of the surface parameters because one cannot find the gradient of the solution of a linear system. For such systems, the data points themselves can be propagated in the surface normal direction. Here we examine the specific case of radial basis functions, and in particular their arrangement as dipoles slightly inside and outside the surface they model.

Recall the level set method, expressed using the notation of Witkin and Heckbert \([15]\),

\[
\dot{F}(x, t) = -s(x, t)(||F_x(x, t)||,
\tag{3}
\]

where \( s \) is the scalar speed function, \( F_x \) is the gradient of \( F \) with respect to the vector \( x \) and \( \dot{F} \) indicates the time derivative of \( F \). Essentially, the implicit surface function \( F \) changes value at each location \( x \) in space as the negative product of the speed function times its gradient magnitude. For dipole level sets, we seek to derive from this formulation the motion of the RBF dipoles necessary to change the function \( F \) appropriately.

We perform a “level set” propagation of the RBF surface (1) by moving each RBF dipole by its speed function along its surface normal direction, and solving (2) to get the new surface. We thus need to derive a formula to propagate the surface normal, so that we can align the dipole to it at each step of the level set propagation. Hence we derive the change in normal direction over time, \( \dot{\mathbf{n}} \).

We will use two projection operators. The first operator \( \mathbf{m}\mathbf{m}^T \) returns the component of a vector operand that lies in the direction of the (unit) surface normal \( \mathbf{n} \). The second operator \( \mathbf{I} - \mathbf{m}\mathbf{m}^T \) returns the component of a vector operand that lies in the tangent plane of the surface, where \( \mathbf{I} \) is the \( 3 \times 3 \) identity matrix.

The derivative of a vector in a unit vector field is perpendicular to the original vector,

\[
\mathbf{n} \cdot \mathbf{n} = 1 \Rightarrow d(\mathbf{n} \cdot \mathbf{n})/dt = 0 \Rightarrow \dot{\mathbf{n}} \cdot \mathbf{n} = 0
\tag{4}
\]

We can thus write the change in normal direction over time as

\[
\dot{\mathbf{n}} = \mathbf{T}\dot{\mathbf{n}} = T\frac{d}{dt}\frac{F_x}{||F_x||}
\tag{5}
\]

\[
= T \frac{||F_x||}{||F_x||^2} F_x - T \frac{d||F_x||/dt}{||F_x||^2} F_x
\tag{6}
\]

\[
= T \frac{F_x}{||F_x||}.
\tag{7}
\]

Substituting \( \dot{F}_x \) using the gradient of (3), we get

\[
\dot{\mathbf{n}} = \frac{1}{||F_x||}(-s||F_x||)\mathbf{x}
\tag{8}
\]

\[
= -Ts_x - s \frac{T F_{xx} \mathbf{n}}{||F_x||}.
\tag{9}
\]

The computation of \( s_x \), the speed gradient, depends on the application. In the case of level-set morphing \([5]\) and scattered point fitting \([13]\), \( s \) is a distance to some target object so its gradient \( s_x \) is a unit vector in the direction of the closest point in the target. Other speed functions, especially those that depend on the geometry of the implicit surface of \( F \), have more complicated gradients.
For example, the speed function for the surface propagation under mean curvature is

\[ s = - \text{div} \mathbf{n} = - \text{div} \frac{F_x}{|F_x|} \]

\[ = - \frac{1}{|F_x|} (\text{tr} F_{xx} - \mathbf{n}^T F_{xx} \mathbf{n}) \]

\[ = - \frac{\text{tr}(TF_{xx})}{|F_x|}, \quad (12) \]

where \( F_{xx} \) denotes the Hessian of \( F \) and \( \text{tr} \cdot \) returns the trace of a matrix.

We can thus compute \( \dot{F}_x \) for (7) by combining the level set propagation equation (3) with (12),

\[ \dot{F}_x = (-||F_x||s)_x \]

\[ = \text{tr}(TF_{xx})_x \]

\[ = \text{tr}(F_{xx})_x - n_x F_{xx} n - n^T F_{xx} n_x - n^T F_{xxx} n \]

\[ = (\text{tr}(F_{xx})_x - n^T F_{xxx} n) - 2n_x F_{xx} n \]

\[ = (\text{tr}(F_{xx})_x - n^T F_{xxx} n) - \frac{2}{|F_x|} TF_{xx} F_{xxx} n, \]

\[ \quad (13) \]

where the last equation evaluates the gradient of the normal as the matrix

\[ n_x = \frac{TF_{xx}}{|F_x|} \quad (14) \]

Thus the time derivative of the normal is

\[ \dot{n} = T \left( \frac{\text{tr}(F_{xx})_x - n^T F_{xxx} n}{|F_x|} - \frac{2TF_{xx} F_{xxx} n}{|F_x|^2} \right). \]

\[ \quad (15) \]

The notation \( F_{xxx} \) is a rank-3 tensor of third derivatives, and can be evaluated as a vector of matrices

\[ F_{xxx} = ((F_{xx})_x)_x, (F_{xx})_y, (F_{xx})_z. \]

\[ \quad (16) \]

where e.g. \( (F_{xx})_y \) is the partial derivative of the Hessian wrt \( y \). The tensor-vector product in (15) may then be decomposed and computed as the weighted sum of matrices

\[ F_{xxx} \mathbf{n} = n_x (F_{xx})_x + n_y (F_{xx})_y + n_z (F_{xx})_z, \]

\[ \quad (17) \]

where \( \mathbf{n} = (n_x, n_y, n_z)^T \).

In 3-D we use the radial basis function \( C^i(x) = ||x - x_i||^3 \) for some fixed center location \( x_i \) of no consequence to differentiation. If we let \( r = ||x - x_i|| \) and the gradient formula \((r^n)_x = nr^{n-2}x\), then the radial basis function \( C^i \) differentiates as

\[ C_x = 3rx_i, \]

\[ C_{xx} = 3rI + \frac{3}{r} xx^T, \]

\[ C_{xxx} = \frac{3}{r} (I_{ij} x_i + I_{ik} x_k + I_{jk} x_j) - \frac{3}{r^3} xx^T xx^T, \]

\[ \quad (18) \]

\[ \quad (19) \]

\[ \quad (20) \]

where the last equation is a rank-3 tensor in Einstein notation, which translates into the vector-of-matrices form

\[ C_{xxx} = \frac{3}{r^3} \begin{bmatrix} 3x & y & z \\ y & x & z \\ z & x & y \end{bmatrix}, \]

\[ - \frac{3}{r^3} \begin{bmatrix} x & xx^T & y & xx^T & z & xx^T \end{bmatrix}. \]

\[ \quad (21) \]

The Laplacian of a radial power function is

\[ \text{Lap} r^n = n(n + 1)r^{n-2} \]

\[ \quad (22) \]

Hence the gradient of the Laplacian term in (15) evaluates to

\[ \text{tr}(C_{xx})_x = (12r)_x = \frac{12}{r} x. \]

\[ \quad (23) \]

The derivatives of the basis functions easily combine to form the derivatives of the RBF system. For example, the RBF surface gradient is simply the sum of basis gradients

\[ F_x = \mathbf{n} + \sum_i a_i C_x (x - x_i). \]

\[ \quad (24) \]

5. ADAPTIVITY

The benefit of RBF dipole propagation versus the uniform grid of the ordinary level set is that the frequency of dipole sampling can vary over the surface, sampling fine features more accurately and smooth regions more coarsely, focusing computation where it is needed most.

To achieve this benefit, some kind of dipole adaptivity method must be implemented to redistribute RBF dipoles appropriately to accurately sample the speed function as it varies across across the surface, and to accurately alter the surface as a result.

![Fig. 1. A sphere morphing to a teapot can become unbounded (a), which is corrected with the adaptive insertion of dipoles (b). Gray disks indicate the floater particles that sample the speed function, yellow cones indicate existing dipoles, and gray cones indicate the adaptively inserted dipoles.](image-url)
Witkin and Heckbert [15] developed a dynamical system to control particle population dynamics so their floater particles would distribute themselves evenly and also adaptively across an implicit surface. Our RBF dipole propagation is implemented on top of such a system, using floater particles to sample the speed function, and benefits from this population dynamics model to sample the speed function uniformly across the surface. Surface particle sampling can also be made adaptive to better sample areas of high surface curvature [51], but for surface propagation, we are more interested in variation of the speed function than of surface geometry. While curvature adaption can be easily extended to speed variance adaption, the speed functions we used in our examples did not vary abruptly enough to require such measures.

Assuming the floater particles adequately sample the speed function, the next concern is adapting the number and placement of RBF dipoles to control the approximation and propagation of the implicit surface. When the surface expands, portions between dipoles can grow arbitrarily large, even unbounded. The population dynamics model ensures these expanses are sampled by floater particles, but without a dipole such areas are difficult to control. We overcome this problem by tracking the distance from each floater particle to its nearest RBF dipole. When this distance exceeds a threshold, then a new dipole is inserted at that particle position. This differs from the constraint insertion of Morse et al. [28] which added a new constraint (e.g. dipole) between pairs of constraints when they grew too far apart.

This RBF dipole level set method can only converge to a shape within the limits of its flexibility of the RBF dipole model. Static RBF surface reconstruction methods often begin with few centers and adaptively add more where the error is greatest [18], and we can extend this concept to the dynamic surfaces of level set propagation. In the dynamic level set case, a floater particle between dipoles might remain still even though it reports a non-zero speed value at that surface position. When a floater particle’s actual speed differs significantly from its measured speed (from the speed function), we insert a new RBF dipole at its position to provide additional control to that portion of the surface propagation.

For example, Fig. 2 uses the distance to a target cone mesh as a speed function for an RBF dipole surface propagation. Starting the dipoles on the vertices of the cone mesh yields a sphere, whose floater particles report a non-zero speed function, even though the RBF dipoles are at locations whose speed function is zero. Inserting dipoles at floater particles whose actual v. measured speed difference allows the sphere to evolve into the desired cone shape.

Our dipoles propagate only in their dipole-normal direction, which means they do not tend to traverse the surface tangentially. Adding dipole repulsion forces could allow dipoles to traverse the surface, especially in directions that improve the propagation, but in our experiments we have not yet found such dipole motion to be necessary.
Fig. 4. A “fertility” sculpture intended to appear smooth, represented as a scanned dense mesh of 5K vertices (a) and as an implicit surface (b) using 320 RBF dipoles (c) fit using the dipole RBF level set method. A plane was fit and intersected to create the flat edge along the base.

6. RESULTS

We demonstrate the surface propagation of dipole RBF surfaces on three examples: as a better alternative to a triangle mesh, as a scattered point interpolation method, and as a bulge-free blending method. The goal of these examples is to illustrate the visual and quantitative quality of the resulting shape model. We did not document the running time which was executed interactive over several minutes in our unoptimized research prototype programmable particle system.

A propagating surface can be fit to a set of scanned points or mesh vertices \( Y \) using the speed function

\[
s(x^i) = (y - x^i) \cdot n^i,
\]

where \( y \in Y \) is the closest scattered data point to the particle \( x^i \). Because the level set method evolves a moving surface, the current surface normal \( n^i \) at particle \( x^i \) is used to allow particles to converge to the space between data particles. Because we use the normal of the propagating surface, this speed function does not need the normals defined at the data points that other approaches require.

Figure 3 illustrates three representations of the Stanford bunny using a similar number of primitives. As a control example, a 692 vertex simplified mesh was constructed [26] and an RBF dipoles was placed at each vertex oriented by the vertex normal averaged from its adjacent faces, yielding the result (a) and error (b). The feature preservation and selective vertex density of mesh simplification adaptively sampled areas near e.g. the nose but overall the reconstruction quality has isolated problems, especially in the ear. Allowing the RBF dipoles to flow under the level set method yielded a better approximation (c) and error (d), even with slightly fewer dipoles. The reconstruction is also visually superior to a mesh with a similar number of vertices (e), and (f) shows that the positions of the propagated dipoles differ significantly from these vertex positions.

Fig. 4 shows a meshed reconstruction of a “fertility” sculpture (a) reconstructed more smoothly using an RBF dipole fit (b) with centers displayed in (c). Assuming one needs only three values for a mesh vertex position (ignoring edges and faces) and five values for an oriented

Fig. 5. A racket modeled with RBF dipoles on a torus and cylinder (a) is propagated using a mean-curvature speed function to blend and smooth the surface (b), especially the handle end (c-f).
RBF dipole (three spatial and two orientation coordinates), the mesh requires 15,000 values whereas the RBF dipole representation requires 1,600 values, about 11% of that of the mesh. The dipole distribution (c) reveals that the population dynamics detected that additional detail was needed on the left side of the base, though the subtle geometric detail of the hairline and the chisel marked texture would require significantly more RBF dipoles to capture. The base was manually intersected with a plane, and the concave region where the mother is seated on the base has significant negative Gaussian curvature which challenges the RBF’s curvature minimizing goal and yields some oscillation. Overall, the RBF dipole representation yields a significantly more compact representation of smooth objects, though sharp feature preservation remains challenging.

Fig. 5 demonstrates the propagation of RBF dipoles under mean curvature. A racket was modeled by fitting an RBF dipoles surface to a torus and cylinder, and the mean curvature propagation smoothly blends one end of the cylinder into the torus, and rounds the other end. Since the flow does not generate a bulge as it blends, it serves as an alternative to the convolution surfaces previously devised to avoid blend bulges [52]. Some shrinkage is evident, so mean curvature flow is often combined with a positive (outward) constant flow to retain the original volume.

Table 1 indicates the running time for each propagation time step of the RBF dipoles surface evolution. These numbers were averaged over five runs, on a 3GHZ 2GB single-core Intel Pentium D CPU. The “RBF Time” indicates time for solving RBF system, whereas “Tot. Time” adds in population dynamics and other overhead. The direct solver (an LU factorization from the TNT library) is more accurate whereas the iterative conjugate gradient solver is faster, especially as it is started with the values from the previous propagation time step, but is less accurate, terminating when the squared residual falls below a global threshold (0.05). The iterative solver is much faster, except not as much on the racket example likely because the mean curvature creates some propagation oscillation that requires many iterations to settle.

### Table 1

<table>
<thead>
<tr>
<th>Shape</th>
<th>Dipoles</th>
<th>RBF Time</th>
<th>Tot. Time</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fertility</td>
<td>330</td>
<td>182 ms</td>
<td>199 ms</td>
<td>18MB</td>
</tr>
<tr>
<td>Bunny</td>
<td>687</td>
<td>1,379 ms</td>
<td>1,441 ms</td>
<td>24MB</td>
</tr>
<tr>
<td>Racket</td>
<td>415</td>
<td>778 ms</td>
<td>851 ms</td>
<td>18MB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Shape</th>
<th>Iters.</th>
<th>CG Solver</th>
<th>Tot. Time</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fertility</td>
<td>20</td>
<td>69 ms</td>
<td>83 ms</td>
<td>22MB</td>
</tr>
<tr>
<td>Bunny</td>
<td>90</td>
<td>582 ms</td>
<td>643 ms</td>
<td>39MB</td>
</tr>
<tr>
<td>Racket</td>
<td>70</td>
<td>502 ms</td>
<td>571 ms</td>
<td>24MB</td>
</tr>
</tbody>
</table>

7. **Conclusions**

We have derived the equations of motion for dipole RBF implicit surface evolving under a speed function sampled on its surface. This enables level set algorithms to avoid a costly voxel representation and use more compact and efficient implicit surfaces based on dipole RBF’s instead. We propagate RBF surfaces by moving dipoles under a normal flow. The propagation of RBF dipoles is more efficient than the propagation of general implicits because it does not require a linear system solution to connect the desired surface motion to the corresponding parametric motion, though the RBF system itself still must be solved to find its unknown coefficients.

The result is that the level set formulation where problem solutions are specified as surface evolution speed functions may now be extended beyond voxel grid isosurface representations to include smooth implicit surface representation, particularly dipole RBF surfaces. We demonstrated several examples for fitting implicit surfaces to existing point and meshed shapes. For man-made objects consisting of smooth surfaces, implicit models are a more compact representation than are polygon meshes or parametric patch networks, and we demonstrated how RBF surface flow helps us construct these implicit models.

This investigation has stimulated several ideas for further research. First, the optimization of RBF models remains primitive, and we do not yet have a way of sensibly determining which RBF centers can be deleted, or how to move RBF centers tangentially to improve an implicit surface approximation. Second, each propagation step of RBF dipole surfaces does not require a linear system solution for the propagation but does require the solution of a new RBF system. As an RBF system evolves, perhaps there is an efficient way to find a new solution given the previous solution, as is done in topological optimization [32]. Third, there may be likewise more efficient ways to solve the linear system for propagation since each step’s solution is near its previous solution.

**Acknowledgments**

Supported in part by the NSF and DARPA under the CARGO grants #0310446 and #0310705, and the NSF ITR grant #0121288. Thanks to Matei Stoila for details in the RBF normal propagation and mean curvature system itself still must be solved to find its unknown coefficients.

**References**


