2004-06-01

Empirical Analysis of Computational and Accuracy Tradeoffs Using Compactly Supported Radial Basis Functions for Surface Reconstruction

Weiming Liu
tanmin@gmail.com

Bryan S. Morse
morse@byu.edu

See next page for additional authors

Authors
Weiming Liu, Bryan S. Morse, and Lauralea Otis

Empirical Analysis of Computational and Accuracy Tradeoffs Using Compactly Supported Radial Basis Functions for Surface Reconstruction

Bryan Morse, Weiming Liu, and Lauralea Otis
Department of Computer Science, Brigham Young University
morse@cs.byu.edu

Abstract

Implicit surfaces can be constructed from scattered surface points using radial basis functions (RBFs) to interpolate the surface's embedding function. Many researchers have used thin-plate spline RBFs for this because of their desirable smoothness properties. Others have used compactly supported RBFs, leading to a sparse matrix solution with lower computational complexity and better conditioning. However, the limited radius of support introduces a free parameter that leads to varying solutions as well as varying computational requirements: a larger radius of support leads to smoother and more accurate solutions but requires more computation. This paper presents an empirical analysis of this radius of support. The results using compactly supported RBFs are compared for varying model sizes and radii of support, exploring the relationship between data density and the accuracy of the interpolated surface.

1. Introduction and Background

A number of techniques have emerged using radial basis functions (RBFs) to interpolate implicit surfaces from scattered surface points and some number of non-surface constraints [7, 9, 8, 4, 1, 2, 6]. These methods basically take the same approach: known points on the surface define where the implicit surface's embedding function must have a value of zero, known off-surface constraints define where the embedding function has nonzero values, and these (point,value) targets are then interpolated using RBFs. Though they differ in various ways (the RBFs used, the means of defining the non-surface constraints, and the tolerance of fitting the points), they all share this key idea: rather than explicitly interpolating the surface, *they interpolate the embedding function implicitly defining the surface*.

Fundamental applications of this idea generally use thin-plate spline RBFs [3] so as to produce the smoothest interpolation possible. However, the direct formulation of this requires the solving of a large, full, generally ill-conditioned system of equations and quickly becomes computationally impractical for large problems.

Various methods have been used to accelerate this, including the use of compactly supported basis functions [4,
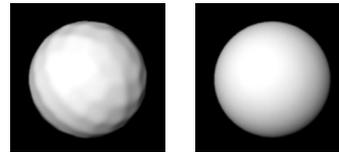

Figure 1. Surfaces using compactly supported RBFs with insufficient data density (left) and sufficient data density (right)

6], approximating a large set of constraints by a well-selected subset [1, 12], and partitioning the domain [5]. All of these trade off accuracy in the reconstructed surface for computational efficiency, recognizing that infinitely precise computation of the surface is already infeasible given noisy data and the limitations of finite-precision arithmetic.

The use of compactly supported RBFs (for example [10], as used in [4]) does not reduce the size of the system of equations but leads to a sparser and better-conditioned matrix to solve. This reduces the storage requirements and computational complexity of the problem [4]. However, this comes at the following costs:

- The solutions, while typically qualitatively similar, are not generally as accurate as thin-plate splines, and
- The radius of support is a free parameter, leading to differing solutions as the radius changes (Fig. 1).

Wendland [11] has shown that compactly supported RBFs of sufficiently high order are qualitatively comparable to thin-plate splines for sufficiently high data density (the number of points typically within the radius of support). The reconstructed surface is well-conditioned with respect to changes in the radius of support, transitioning smoothly to more accurate solutions as the radius of support (and corresponding data density) increases.

Thus, the free radius-of-support parameter is not simply a source of uncertainty in the solution but *a desirable control parameter for trading off between speed and accuracy*.

But what is this nature of this tradeoff? As one varies the radius of support, what can one reasonably predict about the effect on the resulting computational efficiency and accuracy? This paper presents the results of an empirical analysis of this radius of support and the accompanying tradeoffs in speed and accuracy.

2. Implementation

For these experiments, we follow the compactly supported approach outlined in [4], which is based on the general approach of [9] and only briefly summarized here.

We begin with a set of points known to lie on the desired implicit surface and constrain the interpolated embedding function to have a value of zero at these points. Using the method of [9], we also place non-zero constraints at a fixed offset in the direction of the known or desired normals at these surface points. We thus have a set of constraints $(\overline{c}_i, h_i)$ such that $h_i = 0$ for all $\overline{c}_i$ on the surface and $h_i = 1$ for all $\overline{c}_i$ at a fixed offset from that surface. We then interpolate an embedding function f from these constraints such that $f(\overline{c}_i) = h_i$.

We obtain this interpolation using an RBF $\phi(r)$ by defining the embedding function f as a weighted sum of these basis functions centered at each of the constraints:

$$f(\overline{x}) = \sum_{i=1}^{n} d_i \phi(\|\overline{x} - \overline{c}_i\|) \tag{1}$$

where $\overline{c}_i$ is the position of the constraint and d_i is the weight of the radial basis function positioned at that point.[1]

To solve for the set of weights d_i that satisfy the known constraints $f(\overline{c}_i) = h_i$, we substitute each constraint $(\overline{c}_i, h_i)$ into Eq. 1:

$$f(\overline{c}_i) = \sum_{j=1}^{n} d_j \phi(\|\overline{c}_i - \overline{c}_j\|) = h_i \tag{2}$$

Eq. 2 thus provides a system of equations for solving for the weights d_j, which are then substituted back into Eq. 1 to provide the embedding function.

For thin-plate radial basis functions, the matrix created by this system of equations has zero values only along the diagonal. For compactly supported RBFs, the matrix has zero elements for all i, j such that $\|x_i - x_j\| > a$, where a is the radius of support.

3. Experiments and Results

In order to be able to evaluate the accuracy of the solutions, we generated several analytically defined 2- and 3-D models to serve as "gold standards". For the 2-D cases, these included a circle, a square, and an ellipse, all with uniformly distributed constraints. For 3-D, we created a sphere with uniformly distributed constraints. We then interpolated these shapes using compactly supported RBFs of varying support, measuring the average number of points within the radius of support and the time to solve for the RBF weights.

---

1   For some RBFs, including the thin-plate spline RBFs, an additional polynomial may also be required.

We then used the interpolated shapes to measure the average error (distance between the curves in the direction of the true curve's normal) at various points on the shape. For the 2-D shapes, these points were generated uniformly at a density much greater than the original constraints. For the 3-D shapes, these points were generated randomly but again with a much greater density than the original constraints. As with [11], we consider the models not only in terms of radius of support and total number of constraints but in terms of the effective *data density*, the number of points typically within the radius of support.

3.1. Timing

As the radius of support varies, the timing should also vary accordingly. Figure 2 shows the time required to solve the system of equations for varying radii of support and varying model sizes for a 2-D circle. As expected, the time required to perform the interpolation increases as the radius of support (and thus the number of non-zero elements of the matrix) increases. The linear relations on this log-log plot suggest a power relationship $O(p^k)$ where p is the number of support points and k ranges from 0.7 for smaller models to around 2.0 (1.9 to 2.1) for larger models.

3.2. Accuracy

As one increases the radius of support, one would also expect to reduce the error in the interpolation. Figs. 3–6 show the average error for reconstruction of the circle, square, ellipse, and sphere models respectively. For each, we measure the average error for varying radii of support and for varying model sizes.

The overlapping results in Fig. 3 agree with the intuition that the primary independent variable is the data density, independent of the actual model size or radius of support. In other words, that the accuracy of the interpolation depends on the number of points used to interpolate each unknown
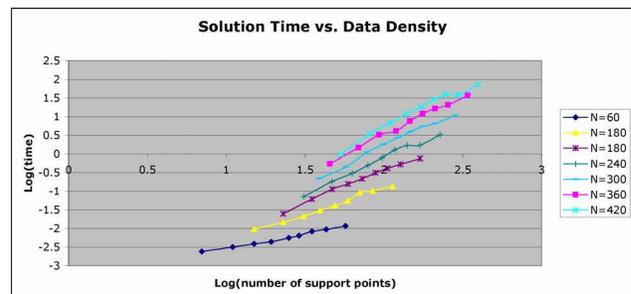

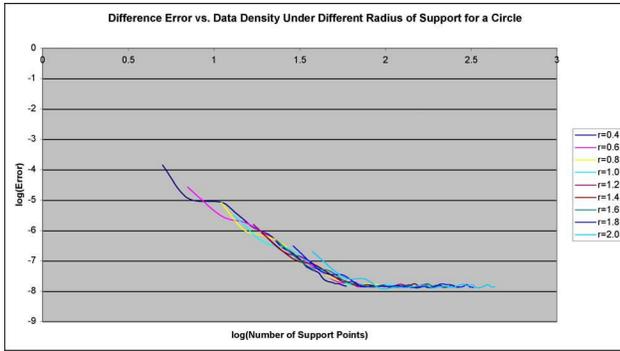**Figure 2. Timing comparisons for varying radii of support and varying model sizes for a reconstructed circle**

Figure 3. Average error vs. data density for a circle with varying numbers of points
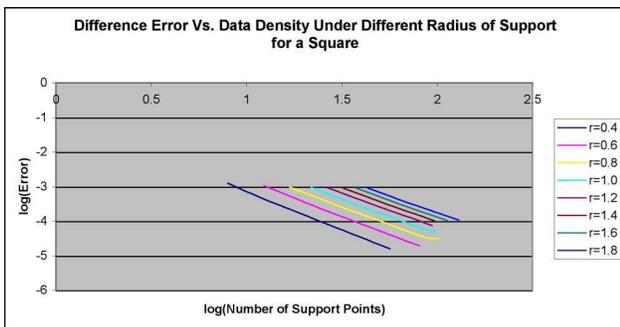

Figure 5. Average error vs. of data density for an ellipse with varying numbers of points

Figure 4. Average error vs. data density for a square with varying numbers of points

Figure 6. Average error vs. of data density for a sphere with varying numbers of points

Figure 7. Thin-plate (dotted curve) and compactly supported (dashed curve) fitting at a high-curvature point

point, not the actual radius of support or other (more distant) points in the model.

These results also demonstrate a linear relationship when plotted on a log-log scale, again suggesting a (negative) power relationship between the data density and the average error, approximately $O(p^{-2})$ until asymptotically tailing off as the limits of finite-precision numerics are reached.

However, the results for the square (Fig. 4) tell a different story. The error again reduces quadratically with the data density, but the error rates are different for different absolute radii of support. Specifically, for comparable data densities, smaller radii of support produce less error.

This result is best explained by the sharp corners of the square. Because they are not affected by as many points on the other side of the high-curvature point, compactly supported RBFs are able to more accurately interpolate the square near a corner. This is illustrated in a simple 1-D case in Figure 7, where a compactly-supported RBF of relatively small radius interpolates near the high-curvature point more closely than a thin-plate spline RBF, though obviously performing less well in smoother areas.

This behavior is also present, though less pronounced, in the case of the ellipse, whose curvature varies less extremely than the square. Th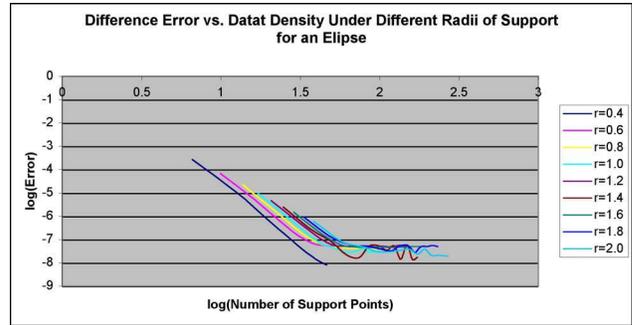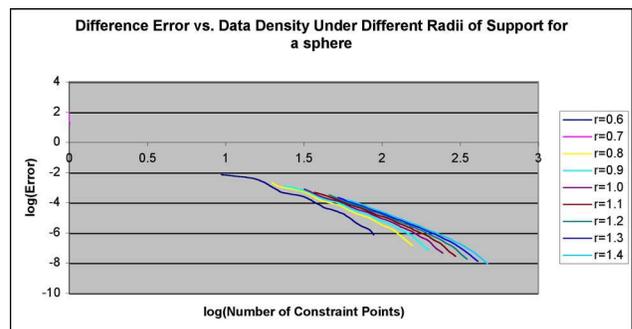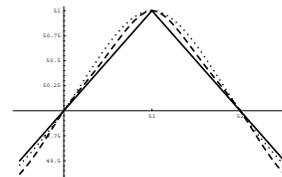e error is primarily driven by the data density, but given comparable data densities, smaller radii of support perform better than larger ones because of their ability to better handle higher-curvature areas.

The results for the 3-D sphere are similar qualitatively to those for the 2-D circle.

3.3. Accuracy as a Function of Shape Curvature

To test this idea that the accuracy depends not only on the data density but on the curvature of the model, we used the ellipse to plot the accuracy at each point as a function of the local curvature (Figure 8). For comparison, we repeated this experiment with thin-plate splines.
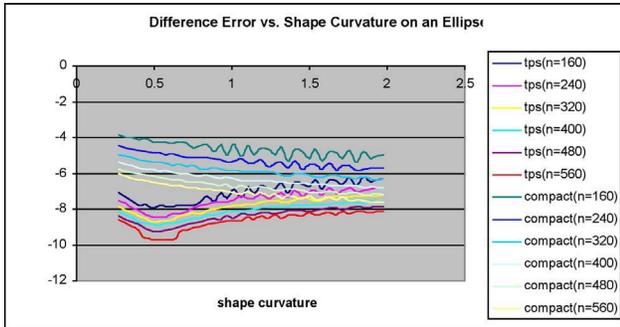
Figure 8. Average error vs. shape curvature for an ellipse with varying numbers of points

As intuition suggests, thin-plate splines are more accurate with greater data density but less accurate with higher curvature. Compactly supported RBFs likewise perform better for higher data densities, but unlike thin-plate splines, compactly supported RBFs perform better in higher-curvature segments and more poorly in smoother segments. This agrees with intuition that the limited-region nature of compactly supported RBFs should perform well in smaller-feature areas and more poorly in broader, smoother areas. Perhaps counterintuitive is that compactly supported RBFs eventually outperform the thin-plate RBFs for higher-curvature segments, though this was hinted at in Figs 4, 5, and 7.

4. Conclusions

The results of these experiments suggest the following:

- The error accrued through using compactly supported radial basis functions for curve or surface reconstruction diminishes steadily as the radius of support increases, though this improvement asymptotically diminishes near the limits of finite-precision numerics.[2]
- The relationship between the error and the radius of support is driven primarily by the scale-independent data density rather than the absolute radius of support. However, this is also affected to a lesser degree by the relationship between the radius of support and the feature size (curvature) of the shape.
- The relationship between the error and the data density appears to follow a power law with an exponent of approximately -2.
- The relationship between the computational time and the data density appears, for a fixed number of total points, to also follow a power law relationship, with an exponent of approximately 2 for larger models.

---

[2] For larger radii, the error may again increase as the data density becomes so high that the matrix becomes poorly conditioned, but we experienced this only for radii that more than encompass the entire model, which defeats the purpose of using compactly supported RBFs.

Taken together, these show that as the number of points in the model (and hence the data density) increases, one can commensurately reduce the radius of support in order to maintain the same data density. This results in comparable error while "buying back" some of the computational time required for the larger model. This agrees with anecdotal evidence suggested in earlier work [4], which tried to maintain a relatively constant data density as the size of the model increased.

References

[1] J. C. Carr, T. J. Mitchell, R. K. Beatson, J. B. Cherrie, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3D objects with radial basis. In *SIGGRAPH 2001 Proceedings*, Computer Graphics Proceedings, Annual Conference Series. ACM SIGGRAPH, ACM Press, August 2001.

[2] H. Dinh, G. Turk, and G. Slabaugh. Reconstructing surfaces by volumetric regularization using radial basis functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, October 2002.

[3] J. Duchon. Sur l'erruer d'interpolation des fonctions de plusieurs variables par les d^m splines. *R.A.I.R.O Analyse numerique*, 12(4):325–334, 1978.

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

[5] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level partition of unity implicits. In *Proceedings 2003 SIGGRAPH*, Computer Graphics Proceedings, Annual Conference Series. ACM SIGGRAPH, ACM Press, 2003.

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

[7] V. V. Savchenko, A. 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.

[8] G. Turk and J. F. O'Brien. Shape transformation using variational implicit surfaces. In *SIGGRAPH '99 Proceedings*, Computer Graphics Proceedings, Annual Conference Series. ACM SIGGRAPH, ACM Press, 1999.

[9] G. Turk and J. F. O'Brien. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics*, 21(4):855–873, October 2002.

[10] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *AICM*, 4:389–396, 1995.

[11] H. Wendland. Error estimates for interpolation by compactly supported radial basis functions of minimal degree, 1997.

[12] G. Yngve and G. Turk. Robust creation of implicit surfaces from polygonal meshes. *IEEE Trans. Vizualization and Computer Graphics*, 8(4):346–359, October-December 1999.