Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization (2003)
G.-P. Bonneau, S. Hahmann, C. D. Hansen (Editors)

Adaptive Smooth Scattered-data Approximation for Large-scale Terrain Visualization

Martin Bertram, Xavier Tricoche, and Hans Hagen

Department of Computer Science, University of Kaiserslautern, Germany
{bertram|tricoche|hagen@informatik.uni-kl.de}

Abstract

We present a fast method that adaptively approximates large-scale functional scattered data sets with hierarchical B-splines. The scheme is memory efficient, easy to implement and produces smooth surfaces. It combines adaptive clustering based on quadtrees with piecewise polynomial least squares approximations. The resulting surface components are locally approximated by a smooth B-spline surface obtained by knot removal. Residuals are computed with respect to this surface approximation, determining the clusters that need to be recursively refined, in order to satisfy a prescribed error bound. We provide numerical results for two terrain data sets, demonstrating that our algorithm works efficiently and accurate for large data sets with highly non-uniform sampling densities.

1. Introduction

Scattered-data fitting is concerned with the global approximation of function values associated with points that are arbitrarily distributed over a compact 2D or 3D domain. In the case of 2D functionals the points p_i are associated with a scalar value f_i, see figure 1. The task consists in computing a surface approximating the function values f_i at the corresponding points in the plane, satisfying a prescribed error bound. Our method provides an efficient construction for adaptive smooth surface approximations. It can easily be extended to higher-dimensional problems.

Data sets provided by modern measurements and numerical simulations become larger and larger (several millions points), which strongly inconveniences the use of a global least squares fit using smooth basis functions. Even triangulated surface approximations are difficult to construct for large-scale non-uniform data, providing only a piecewise linear representation.

The resolution of uniform data sets is mostly adapted to the finest geometric detail that needs to be represented. For large regions of less complexity, there are too many redundant samples. It is therefore useful to reduce the sampling density locally to the level of geometric complexity and to approximate this non-uniform data using adaptive methods. Classical schemes inherited from the research tradition in surface fitting and scattered data approximation producing smooth surfaces prove unsuited for data sets exceeding a few hundred points since they become extremely time-consuming when dealing with larger point sets.

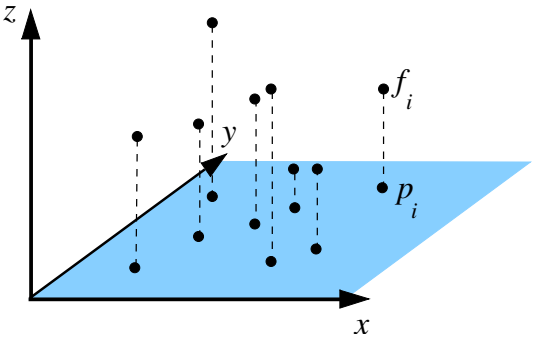


Figure 1: *Scattered data points p_i with associated function values f_i.*

In this paper a new method is proposed that attacks this deficiency by dividing the fitting task in two steps: The first one consists in a quadtree-like clustering of the sample points. It provides subsets that we next locally fit by means of low order Bézier splines. The resulting piecewise continuous surface is then locally approximated by a continuous B-spline surface. This surface is obtained by knot

removal and constitutes our current approximation of the functional. Residuals are estimated and used to control further subdivision of clusters. In that way the method facilitates an adaptive fitting of sparsely distributed data and concentrates refinement on small regions exhibiting large errors. Unlike global least-squares methods, our algorithm localizes the computation at multiple levels of resolution.

The contents of the paper are organized as follows. Related work dealing with scattered data fitting of large data sets is briefly summarized in section 2. The different steps of our algorithm are explained in section 3. Numerical results are proposed for two terrain data sets in section 4. Conclusion and future work are discussed in section 5.

2. Related Work

The topic of scattered data fitting and interpolation has benefited from much research. Many overviews can be found in the literature [8, 17, 18, 21]. Traditional techniques inherited from the approximation theory may be decomposed into two categories: Some of them make use of radial basis functions [15, 7, 9, 18]. Others are based on global spline interpolation or approximation [1, 4, 10, 11, 12, 13].

The shortcoming of most global approaches is that they require the solution of large linear systems which are not always well-conditioned and sparse. Consequently, global fitting methods often become prohibitively time and memory consuming when dealing with data sets whose size exceeds a fairly small number of points (say 500). Therefore new techniques have been designed to reduce the computational complexity to enable the processing of large-scale scattered data sets that contain millions of points.

A method based on multilevel B-splines has been proposed [16]. In a coarse to fine approach the approximation error is used to control the refinement of control lattices over which a C^2 cubic B-spline function is defined. A limitation of this method is the fact that lattices' refinement takes place globally. This is inefficient when accurately reproducing small local features. To overcome this problem, this method has been adapted to local refinement of rectangular regions [22]. The approximated regions can be quite large for complex data sets and user interaction may be required for chosing them.

In [14, 5] a regular triangulation is used and triangular Bézier patches are defined on a subset of all triangles. The global surface is then constructed by ensuring C^1 continuity of the Bézier patches in the remaining triangles. A major drawback of this approach is the lack of hierarchal structure in the surface construction which impose global re-computation if higher accuracy is required in a small region. This technique has been improved recently [20] by the use of a binary triangle tree which allows level-of-detail representation. However, the overall algorithm is quite complex and dependent on an adaptive triangulation of the domain that needs to be maintained to avoid cracks. The resulting surface serves as input for efficient rendering, proving that smooth surface approximations of low polynomial degree are not necessarily less efficient than pure triangle-based methods.

3. Method Description

Our algorithm for adaptive approximation of scattered data is composed of the following steps:

- Adaptive clustering based on quadtree refinement.
- Least squares-fitting of polynomial patches to the data points located in the individual clusters.
- Combining the piecewise polynomials to a B-spline surface with multiple knots at the cluster boundaries. Knot removal is used to combine the fitted patches into a smooth representation.
- Recursive refinement of clusters with local errors above a specified tolerance.

We assume that the scattered data may be non-uniformly distributed over the domain. The sampling density is assumed to be greater in regions of high geometric complexity and smaller in less detailed regions. Our algorithm produces multiple levels of approximation using dyadic knot refinement. In smooth surface regions the refinement terminates when a prescribed error bound is satisfied. The remaining regions are recursively refined and the local approximations smoothly join with the coarser surface components.

Given a set of scattered data,

$$\{(p_i,\, f_i) \mid p_i \in \mathbb{R}^2,\, f_i \in \mathbb{R},\, i = 1,\ldots,n\},$$

where p_i are data points with associated function values f_i, see figure 1, we construct a sequence of approximating functions F_j $(j = 1,2,\cdots)$, minimizing the residuals $\|F_j(p_i) - f_i\|$. Increasing the index j will augment the resolution of F_j. Our algorithm can be easily adapted to approximate higher-dimensional data where $p_i \in \mathbb{R}^m$ and $f_i \in \mathbb{R}^n$.

Every level of resolution is defined by the set

$$L_j \;=\; \{C_j,\, F_j\},$$

where $C_j = C_j^k$ $(k = 1,\cdots,n_j)$ defines a partitioning of the domain, where the individual clusters C_j^k correspond to quadtree nodes at level j, see figure 2.

We start the cluster refinement with a uniform space partitioning C_0 such that every cluster contains enough data points for a polynomial approximation. We use the least-squares method to determine a bilinear or biquadratic polynomial surface approximation for the function values f_i with $p_i \in C_0^k$ for every cluster. The individual patches are combined to a smooth B-spline surface by reducing the multiplicity of inner knots to one, resulting in a C^0 surface in the bilinear case and a C^1 surface in the biquadratic case. This surface is denoted as base approximation F_0.

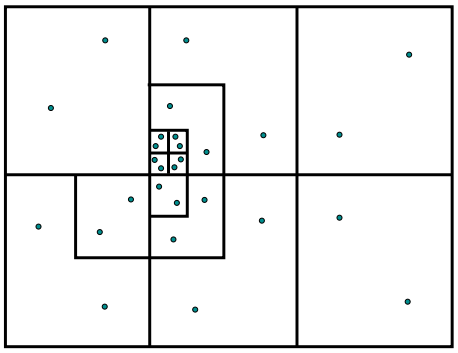


Figure 2: *Adaptive clustering based on quadtree refinement.*

When extremely coarse approximations are required, we start with only one base cluster. We note that coarser representations can be computed more efficiently by coarsening a finer base approximation, since fitting the entire data set is computationally expensive and not necessary for coarse levels of detail. (For this purpose, techniques operating on regular grids, like wavelets or conventional least-squares techniques may be used.)

Then, we uniformly refine the clustering C_0 of the base resolution by splitting every cluster into four rectangular regions of equal size, providing C_j (j=1). For every data point p_i, the associated function value f_i is replaced by the signed distance to the base approximation (orthogonal to the domain),

$$\Delta f_i^j \;=\; f_i - F_{j-1}(p_i). \tag{1}$$

These signed distances are the local residuals of the approximation that need to be reduced in the following fitting steps.

For every cluster, we determine the maximal residual from the L^∞-norm,

$$\varepsilon(C_j^k) \;=\; max_{i:p_i \in C_j^k}\{\|\Delta f_i^j\|\}. \tag{2}$$

The residuals $\varepsilon(C_j^k)$ are compared to a prescribed error bound ε. Again, a polynomial is fit to the data points in every cluster where $\varepsilon(C_j^k) \leq \varepsilon$. This kind of refinement will leave some subtrees of the quadtree empty. The clusters satisfying the error bound are denoted as "idle" and are not stored in the quadtree.

Based on these polynomial patches a detail function ΔF_j is constructed, minimizing the local residuals, such that

$$F_j \;=\; F_{j-1} + \Delta F_j. \tag{3}$$

To obtain ΔF_j (and thus F_j), the individual polynomial patches are merged by knot removal, as described later. The support of ΔF_j extends into all clusters in the 8-neighborhood of fitted clusters, in order to guarantee continuity. Hence, a few "idle" clusters need to be added to the octree, besides the clusters with approximating polynomials.

For further refinement we consider all clusters C_j^k located in the support of ΔF_j. The residuals of points located in these clusters are re-evaluated, according to equation (1) with incremented level index j. The refinement is recursively repeated, until no clusters need to be refined or until a global error bound is satisfied by some L_j.

In the following we describe the fitting and knot-removal procedures used in our algorithm.

3.1. Least-squares Fitting

Considering the points p_i $(i = 1, \cdots, m)$ located in a certain cluster, we need to determine an approximating polynomial,

$$P(s,t) \;=\; \sum_{j=1}^{n} c_j \phi_j(s,t),$$

where the basis functions ϕ_j are products of individual Bernstein polynomials in s and t, for example.

In most cases the number of points is greater than the number of basis functions, i.e. $m > n$, and a linear system for an interpolating surface would be over-determined:

$$\mathbf{Ac} \;=\; \mathbf{f}, \quad a_{ij} = \phi_j(p_i). \tag{4}$$

The residual of this interpolation problem,

$$\sum_{i=1}^{m} \|P(p_i) - f_i\|^2 \tag{5}$$

is minimized by the solution of

$$\mathbf{A^T Ac} \;=\; \mathbf{A^T f} \tag{6}$$

(least-squares fitting [3]). In most cases, the matrix $\mathbf{A^T A}$ is non-singular, providing the coefficients c_j for the best fitting polynomial.

Since $\mathbf{A}$ is not sparse and depends on the points p_i, the cost for this fitting process is dominated by the time complexity $O(m\,n^2)$ for computing $\mathbf{A^T A}$. Since the polynomial degree is fixed (we use bilinear and biquadratic patches), we have an $O(m)$ fitting operation.

Problems occur when the system (4) is under-determined, i.e. $m < n$, or when the matrix $\mathbf{A^T A}$ is singular. The latter is the case, for example, when the data points were downsampled from a regular grid and are mostly collinear, such that there are not enough constraints for determining the coefficients associated with the s- or t-direction.

Other cases exist, where the least-squares residual is small, but the slopes or curvatures of the fitted patch are extremely high, resulting in a poor representation. This happens mostly in clusters containing only few points some of which are close in the domain with great differences in their associated function values. In the "empty" regions of such a

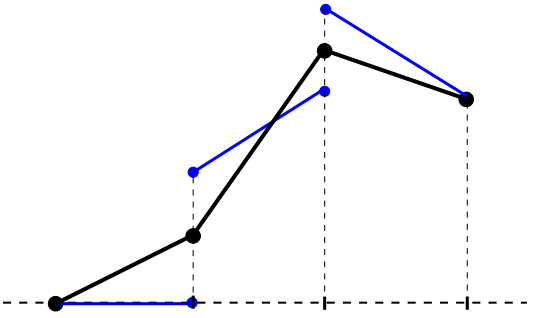


Figure 3: *Knot removal for piecewise linear representation. The left side corresponds to a zero function on an "idle" cluster, while the right side is treated like a domain boundary.*

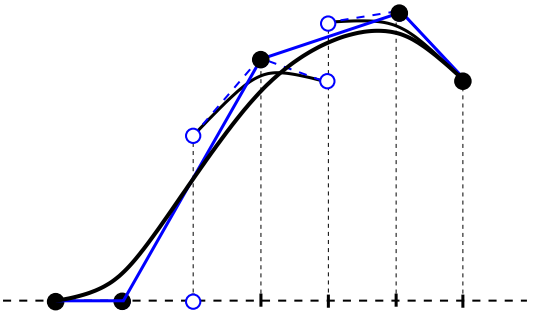


Figure 4: *Knot removal for piecewise quadratic representation. The zero function on the left side is approximated such that the boundary is C^1-continuous.*

cluster the surface can take arbitrarily large values. We detect these cases by comparing the variance of function values f_i with the variance of the Bézier points defining the fitted polynomial. We allow (by a factor of four) greater variance of Bézier points, since these control points are generally not located on the surface.

In all cases, where the least squares fit is not feasible or acceptable, we reduce the polynomial degree by one and solve the least squares system (6), again. We note that at least a constant approximation is feasible, since every cluster with non-zero residual contains at least one point. The fitted polynomial is degree elevated, providing the same number of Bézier points for each patch. We use a bilinear/biquadratic Bézier representation.

3.2. Knot Removal

Assuming that all clusters have an approximating polynomial, this piecewise smooth surface can be represented as a single B-spline patch with multiple inner knots (double and triple knots in the bilinear and biquadratic case, respectively). In this case, the Bernstein polynomials are B-splines

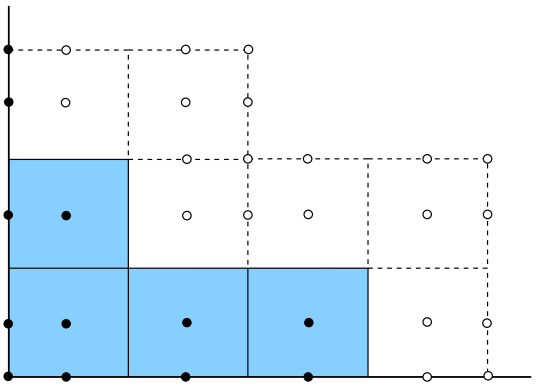


Figure 5: *Support and control points of ΔF_j. The lower left corner defines a domain boundary, requiring extra control points. The white clusters are "idle" and contain only zero control points.*

and thus the Bézier points of the individual patches correspond to de Boor points defining the B-spline surface.

Knot removal [6, 19] can be used to reduce the multiplicity of inner knots to one, such that the new surface is a C^0 or C^1 continuous approximation to the initial piecewise smooth surface. Exploiting the regular structure of the control mesh, knot removal can efficiently be implemented by a least-squares fit computed for each row and column of de Boor points. In our algorithm, however, we want to avoid such a global fitting problem and use local masks for knot removal. This has the advantage that it also works in those cases where data is missing, due to the adaptive refinement.

In the bilinear case, we have four Bézier points (coefficients) interpolating the patch at the cluster corners. At each corner, we take the average of the function values corresponding to the adjacent patches. This process is illustrated for the one-dimensional case in figure 3. If one of the adjacent clusters is "idle", we use the zero function as approximating polynomial before knot removal. After the removal, the residuals of these clusters have changed and need to be re-evaluated. All clusters in the support of ΔF_j are considered for the next level of refinement.

In the biquadratic case, we have 3×3 Bézier points for every cluster. For every row/column of control points, our knot removal procedure simply removes the points located on cluster boundaries, see figure 4. (In Bézier representation, these points would be set to the average of their two neighbors in the row/column.) Now, we have only one de Boor point for every cluster, except on the boundary of the support of ΔF_j. At the boundaries inside the data set's domain, two rows/columns of de Boor points (that need not be stored) are zero, see figure 5. The additional de Boor points on the data set boundary define the boundary curve of the approximation.

We note that the support of ΔF_j does not necessarily define a rectangular region. When evaluating the function, we just use de Boor points inside a smaller rectangular region, defining a B-spline patch inside the larger surface.

3.3. Efficient Evaluation

For computing the individual detail functions F_j, we have approximated the residuals with respect to F_{j-1}. We avoid to evaluate the entire series of detail functions,

$$F_j = F_0 + \sum_{l=1}^{j} \Delta F_l.$$

In order to evaluate the final approximation efficiently, it is desirable to have a single B-Spline representation for every region, using the finest level of detail available. This problem is simply solved by knot insertion on the coarser levels. Due to knot insertion, the number of de Boor points is locally increased without changing the represented surface. The coefficients of F_{j+1} are simply added to the representation of ΔF_j, providing a unique representation of F_j on the support of ΔF_j. Outside this support, the coefficients of the coarser representations, e.g. $F_{j-1}, F_{j-2}, \cdots$ are used for evaluation.

The overall computation time of our algorithm for approximating n scattered data points is $O(n \log n)$, since the number of levels is $O(\log n)$ and the construction of each level L_j requires $O(n)$ operations. For uniformly distributed data, we can start with a base level of fine-resolution, reducing the number of levels to a constant. The memory requirement of our method is $O(n)$, since the maximal number of control points required to represent the finest level (if it was dense) is greater than the sum of control points used on all coarser levels (this number is decreases by four for each level).

4. Numerical Results

We use two terrain data sets to test our method. In both cases the original data is defined over a rectilinear grid. To obtain scattered data sets we apply a down-sampling scheme that randomly removes data points associated with low curvature values (based on discrete curvature estimates). The idea behind this choice is to remove redundant data in smooth regions preserving sharp cusps and edges corresponding to geologic features like mountains, ridges, and canyons. This way, we obtain scattered data with highly non-uniform density providing the input for a challenging approximation problem. Tests are carried out on a PC with AMD Athlon 1100 Mhz processor and 1.5GB RAM.

4.1. Crater Lake Data Set

We first use the Crater Lake data set from USGS. The original data contains about 160,000 points. After down-sampling we obtain a scattered data set with 18,818 points, that is 11.8% of the original, see figure 6. The starting resolution of our cluster grid C_0 is 32×24. All clusters contain a fairly similar number of points before processing. This first example is intended to illustrate the different aspects of our algorithm.

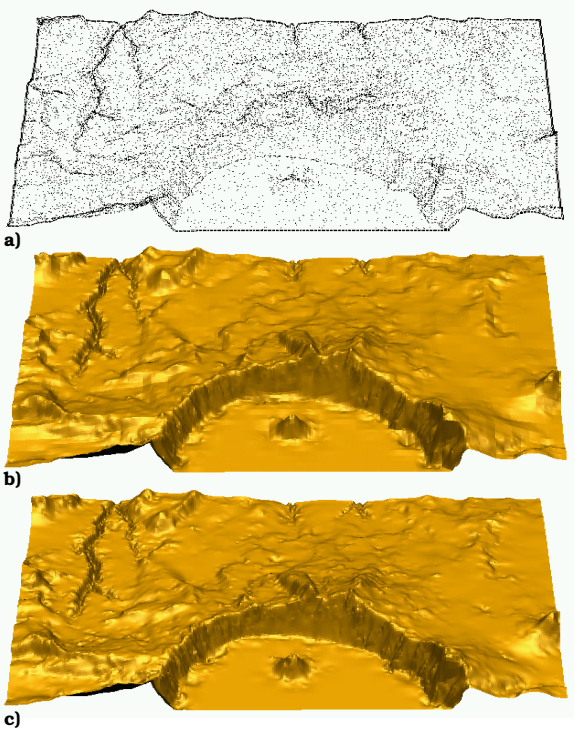


Figure 6: *a) Crater Lake data set, composed of 18,818 points (11.8 percent of its original size). b) Adaptive bilinear approximation. c) Adaptive biquadratic approximation.*

The first fitting applied to the base clusters for the bilinear and biquadratic cases is depicted in figure 7 (left). The discontinuities of both piecewise polynomial surfaces are then eliminated by knot removal, as shown in figure 7 (right).

The continuous surface obtained in that way corresponds to the first iteration of the algorithm. Successive approximation steps for the biquadratic case are illustrated in figure 8. The residuals obtained in successive levels are shown in color plate 1. Gray regions correspond to clusters that satisfy the prescribed error bound and are not further subdivided.

Putting it all together, numerical results based on our non-optimized implementation are enumerated in table 1. The L^2-error computed in each step gives insight into the average approximation quality. Error bound ε is set to 0.5% of the overall amplitude. We observe that the piecewise biquadratic approximation provides more flexibility and thus leads to a better fit (with respect to the L^2 norm) for this data set.

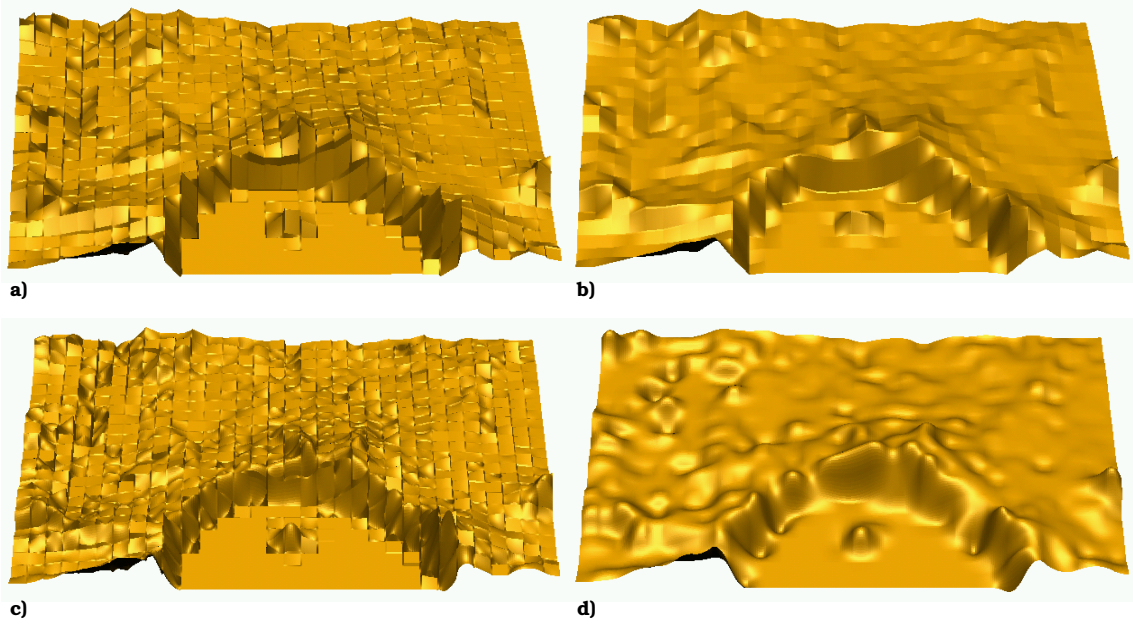


Figure 7: *a) Piecewise bilinear fit of the base level (32 × 24 clusters). b) C^0-continuous approximation after knot removal. c) Piecewise biquadratic fit of the base level. d) C^1-continuous approximation after knot removal.*

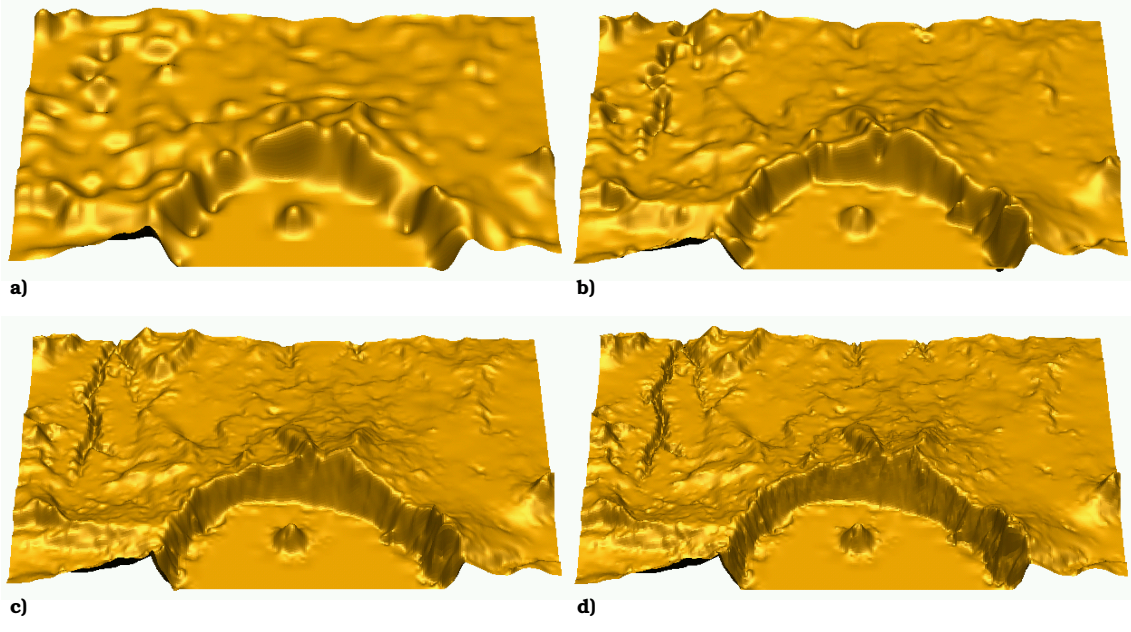


Figure 8: *Approximations at different levels of subdivision (biquadratic case). a-d) Levels 0-3.*

	bilinear fit				biquadratic fit			
step	time/step	total time	error L^2	no. clusters	time/step	total time	error L^2	no. clusters
1	47	47	2.344	768 (100%)	147	147	4.654	768 (100%)
2	70	117	1.408	2,645 (86%)	138	285	2.520	2,757 (90%)
3	92	209	0.899	5,595 (46%)	125	410	1.184	5,983 (49%)
4	179	388	0.621	6,178 (13%)	223	633	0.544	5,966 (12%)
5	576	964	0.452	5,119 (2.6%)	754	1,387	0.326	3,987 (2.0%)
6	2,312	3,276	0.373	2,508 (0.3%)	2,886	4,273	0.266	1,125 (0.1%)

Table 1: *Numerical results for the Crater Lake data set. The tolerance* ε *is* ***0.5%*** *of the overall amplitude. Times are given in ms. Approximation errors are measured in percent of the data set's amplitude. The number of refined clusters is also provided as percentage of clusters in a uniform grid.*

	bilinear fit				biquadratic fit			
step	time/step	total time	error L^2	no. clusters	time/step	total time	error L^2	no. clusters
1	1,576	1,576	6.098	3,072 (100%)	4,172	4,172	13,254	3,072 (100%)
2	1,563	3,139	3,692	10,987 (89%)	4,329	8,501	7.867	11,793 (96%)
3	1,868	5,007	2.071	33,234 (68%)	4,714	13,215	4.327	37,618 (77%)
4	2,702	7,709	1.203	85,435 (43%)	4,590	17,805	1.938	91,894 (47%)
5	4,152	11,861	0.792	163,942 (21%)	5,339	23,144	0.793	175,031 (22%)

Table 2: *Numerical results for Seattle data set. The tolerance* ε *for cluster refinement is* ***1%*** *of the amplitude. Times are given in ms. Approximation errors are measured in percent of the data set's amplitude.*

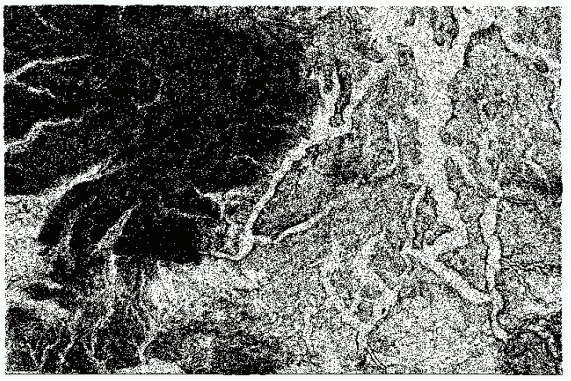


Figure 9: *Seattle data set, composed of 586,970 points.*

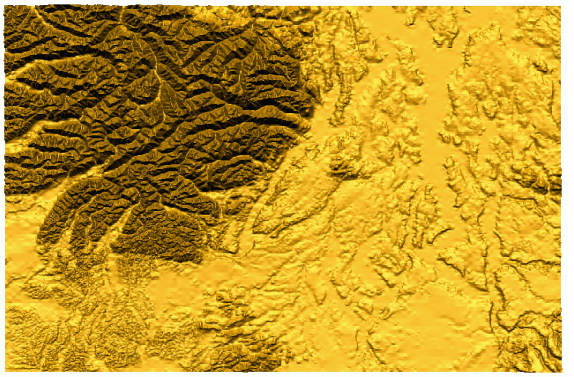


Figure 10: *Adaptive bilinear approximation of Seattle data set.*

4.2. Seattle Data Set

The second data set is much larger than the previous one. It corresponds to the landscape profile around Seattle, WS, provided by USGS. The original data contains about 180 millions points. After down-sampling we obtain a scattered data set with 586,970 points, that is 0.33% of the original, see figure 9. The starting configuration has 64×48 clusters. With a piecewise bilinear fit one obtains the reconstructed surface shown in figure 10. For the biquadratic case the result can be seen in color plate 2. As in the previous example numerical results are summed up in table 2. The error bound ε is set to 1% of the maximal amplitude. In contrast to the results obtained with the Crater Lake data set, we observe slightly better results in the bilinear case than in the biquadratic one. However both results become very close when accuracy increases. An explanation is that uneven terrain like the mountains located in the upper-left part of the data set is better approximated using piecewise bilinear surfaces rather than smooth biquadratic representations.

5. Conclusions

We presented a very efficient and robust adaptive approximation tool for highly non-uniform scattered data. We have

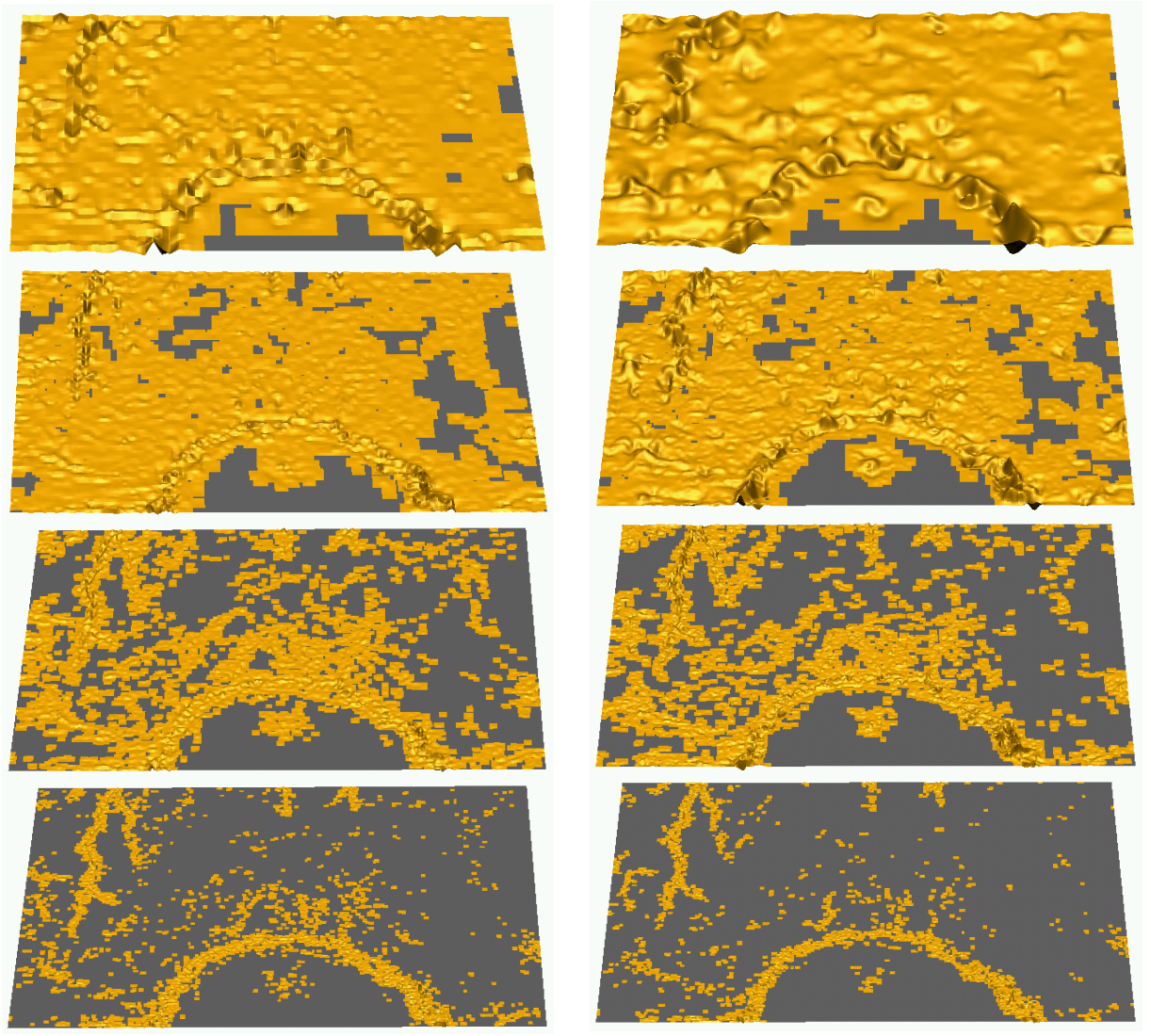
demonstrated that our algorithm provides smooth surface approximations of high quality for large terrain data sets with locally steep gradients defining complex geometry. Our smooth refinable surface representation can be used as a basis for real-time terrain visualization.

Acknowledgements

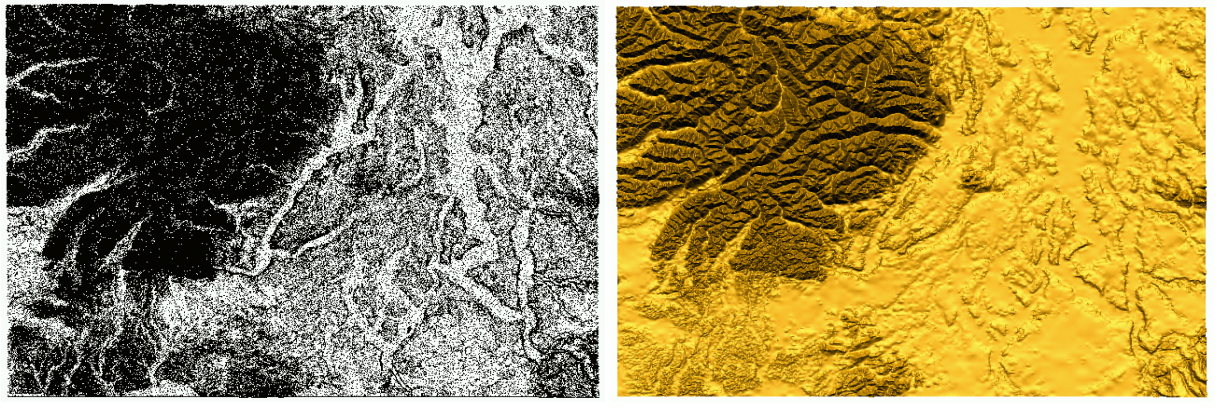
The Crater Lake and Seattle datasets used to test our method were provided by U.S. Geological Survey (USGS). The second data set was downloaded from the web site of the Department of Geological Sciences at the University of Washington in Seattle: http://duff.geology.washington.edu/.

References

1. E. Arge, M. Daehlen, and A. Tveito. Approximation of Scattered Data Using Smooth Grid Functions. In *J. Computational and Applied Mathematics*, **59**:191–205, 1995.
2. R. K. Beatson, W. A. Light, and S. Billings. Fast Solution of the Radial Basis Function Interpolation Equations: Domain Decomposition Methods. In *SIAM J. Scientific Computing*, **22**(5):1717–1740, 2000.
3. W. Boehm and H. Prautzsch, Geometric Concepts for Geometric Design, A.K. Peters, Ltd., Wellesley, Massachusetts, 1994.
4. W. A. Daehmen, R. H. J. Gmelig Meyling, and J. H. M. Ursem. Scattered Data Interpolation by Bivariate C^1-Piecewise Quadratic Functions. In *Approximation Theory and its Applications*, **6**:6–29, 1990.
5. O. Davydov and F. Zeilfelder. Scattered Data Fitting by Direct Extension of Local Polynomials with Bivariate Splines. To appear in *Advances in Comp. Math.*.
6. G. Farin, G. Rein, N. Sapidis, and A. J. Worsey. Fairing Cubic B-Spline Curves. In *Computer Aided Geometric Design*, **4**:91–103, 1987.
7. T. A. Foley. Scattered Data Interpolation and Approximation with Error Bounds. In *Computer Aided Geometric Design*, **3**:163-177, 1986.
8. R. Franke. Scattered Data Interpolation: Test of some Methods. In *Mathematics of Computation*, **38**(157):181–200, 1982.
9. R. Franke., H. Hagen. Least Square Surface Approximation Using Multiquadrics and Parametric Domain Distortion. In *Computer Aided Geometric Design*, **16**:177–196, 1999.
10. R. Franke., G. M. Nielson. Scattered Data Interpolation of Large Sets of Scattered Data. In *Intl. J. Numerical Methods in Engl.*, **15**:1691–1704, 1980.
11. R. H. J. Gmelig Meyling and P. R. Pfluster. Smooth Interpolation to Scattered Data by Bivariate Piecewise Polynomials of Odd Degree. In *Computer Aided Geometric Design*, **7**(5):439–458, 1990.
12. B. F. Gregorsky, B. Hamann, and K. I. Joy. Reconstruction of B-Spline Surfaces from Scattered Data Points. In *Proc. Computer Graphics International 2000*, pp. 163–170, 2000.
13. G. Greiner and K. Hormann. Interpolating and Approximating Scattered 3D Data with Hierarchical Tensor Product Splines. In A. Le Mehauté, C. Rabut, and L. L. Schumaker, *Surface Fitting and Multiresolution Methods*, pp. 163–172, 1996.
14. J. Haber, F. Zeilfelder, O. Davydov, H.-P. Seidel. Smooth Approximation and Rendering of Large Scattered Data Sets. In *Proc. IEEE Visualization 2001*, pp. 341–347, 2001.
15. R. L. Hardy, W. M. Gofert. Least Squares Prediction of Gravity Anomalies, Geoidal Undulations, and Deflections of the Vertical Multiquadrics Harmonic Functions. In *Geophysical Research Letters*, **2**:423–426, 1975.
16. S. Lee, G. Wolberg, and S. Y. Shin. Scattered Data Interpolation with Multilevel B-Splines. In *IEEE Transactions on Visualization and Computer Graphics*, **3**(3):228–244, 1997.
17. S. K. Lodha and R. Franke. Scattered Data Techniques for Surfaces. In H. Hagen, G. M. Nielson, and F. Post, *Proc. Dagstuhl Conf. Scientific Visualization*, pp. 182–222, 1999.
18. M. J. D. Powell. Radial Basis Functions for Multivariate Interpolation. In J. C. Mason and M. G. Cox, *Algorithms for Approximation of Functions and Data*, pp. 143–168, 1987.
19. N. Sapidis and G. Farin. Automatic Fairing Algorithm for B-Spline Curves. In *Computer Aided Design*, **22**(2):121–129, 1990.
20. V. Scheib, J. Haber, M. C. Lin, H.-P. Seidel. Efficient Fitting and Rendering of Large Scattered Data Sets Using Subdivision Surfaces. In *Computer Graphics Forum (Eurographics 2002 Conf. Proc.*, **21**:353–362, 2002.
21. L. L. Schumaker. Fitting Surfaces to Scattered Data. In G. G. Lorentz, C. K. Chui, and L. L. Schumaker, *Approximation Theory II*, pp. 203–268, 1976.
22. W. Zhang, Z. Tang, and J. Li. Adaptive Hierarchical B-Spline Surface Approximation of Large-Scale Scattered Data. In *Proc. Pacific Graphics '98*, pp. 8–16, 1998.



Color Plate 1: *Detail functions ΔF_j (j=1,...,4) for piecewise bilinear (left) and biquadratic (right) approximations. The grey regions are located outside the support of ΔF_j.*



Color Plate 2: *Seattle data set composed of 586,970 points (left) and adaptive biquadratic approximation (right).*