Reconstruction of B-spline Surfaces From Scattered Data Points

Benjamin F. Gregorski, Bernd Hamann, and Kenneth I. Joy

Center for Image Processing and Integrated Computing (CIPIC), Department of Computer Science, University of California, Davis, CA 95616-8562, USA
{gregorsk,hamann,joy}@cs.ucdavis.edu
http://graphics.cs.ucdavis.edu

Abstract

We present a new approach for reconstructing a smooth surface from a set of scattered points in three-dimensional (3D) space. Our algorithm first decomposes a given point set into a quadtree-like data structure known as a strip tree. *The strip tree is used to fit a set of least squares quadratic surfaces to the data points. These quadratic surfaces are then degree-elevated to bi-cubic surfaces and blended together to form a set of B-spline surfaces that approximates the given point set.*

1. Introduction

Digitization devices and scanners generate very large point sets representing complicated geometric models. Data sets typically result from multiple scans and frequently multiple viewpoints. Unfortunately, the resulting massive scattered point sets are not suitable for integration into computer-aided design (CAD) systems. We discuss a method that constructs a set of B-spline surfaces from scattered data points that is usable for further processing in a CAD system.

The basis for our reconstruction is the decomposition of a scattered point set into a 3D strip tree. Our tree structure is an extension of the original strip tree presented by Samet in [10] and [11], and it is similar to the BOXTREE structure of Barequet et al. in [1] Our structure is similar to a quadtree, except that each node in our tree represents a bounding box whose orientation is defined by the best-fit plane approximating the data points inside a bounding box. After each decomposition step, the new tree structure defines a better appoximation to the underlying data points. Decomposition consists of the following steps:

1. A bounding box for all given data points is computed. (The three box axes are parallel to the global coordinate system axes).
2. This bounding box is re-oriented so that its three axes are parallel to the three axes defined by the *principal directions* of the underlying point set, obtained by *principal component analysis (PCA)* . A similar approach for clustering data is used in [4], [5] to generate hierarchies of triangulations for scattered data. This step defines the best-fit (least-squares) plane approximating the data points.
3. The root bounding box is recursively subdivided until the resulting set of bounding boxes, all re-oriented according to their principal directions, approximates the data points by least squares planes, within a user-defined error tolerance.
4. The resulting tree of bounding boxes, the strip tree, is used to fit a set of surfaces approximating the data points inside each box.

Once the scattered points have been decomposed into subsets using the strip tree, we use a bottom-up fitting algorithm to fit a set of surfaces to the points. The tree is used to approximate the data points in each box. Additional points on the boundary faces of the boxes must be used to create the surfaces. The approximation process starts by fitting surfaces to the next-to-last level in the tree. These surfaces are then blended together to form a set of continuous surfaces.

The decomposition method produces a hierarchy of oriented bounding boxes that we can use to construct a set of B-spline surfaces that approximates the scattered points. The construction process utilizes the bounding boxes in place of the scattered point set.

Our paper is structured as follows: In Section 2, we review previously published methods that are related to smooth surface reconstruction. In Section 3, we discuss the

construction of the oriented bounding boxes for a scattered point set. In Section 4, we describe the characteristics of the generalized strip tree. This tree is utilized to progressively fit B-spline surfaces to the scattered point set, which is described in Section 5. Results of our algorithm are provided in Section 6. Conclusions and future work are discussed in Section 7.

2. Related Work

Many different methods exist for reconstructing surfaces from scattered points. Some methods attempt to approximate the surface using a surface based on subdivision or an implicit function. In [9], Hoppe et al. represent the surface to be reconstructed as the zero set of a signed distance function. A contouring algorithm is then used to extract the zero set of the signed distance function that approximates the surface. In [6], Hoppe first constructs an optimized triangular mesh of the data points. The optimized mesh is then used to fit a piecewise smooth subdivision surface that approximates the data. This method can represent sharp "creases" and "darts" in a data set.

Other methods reconstruct one or more B-spline surfaces from the data points. Eck et al. [2] reconstruct a network of B-spline surfaces from given data points. The surface is initially approximated by a dense triangular mesh that is then mapped from a triangular to a quadrilateral mesh topology. A surface spline construction is then used to generate a set of G^1-continuous B-Spline surfaces. Levoy and Krishnamurthy [8] apply an interactive approach to the reconstruction process by having a user interactively specify patch boundaries over an initial dense polygonal mesh. Each of the resulting sections is then parameterized by laying a grid of springs across the polygon mesh. A tensor product B-spline surface is finally fitted to the grid. Greiner and Hormann [3] use hierarchical B-splines to interpolate and approximate scattered data. They first parameterize the data points and then optimize the output surface with respect to a fairness functional.

Our algorithm is of the second type and constructs a set of B-Spline surfaces. We introduce a 3D strip tree for decomposing a given point set. The strip tree decomposes the set of points into a quadtree-like structure that is used to fit B-spline surfaces to the data.

3. Oriented Bounding Boxes

Given a set of points (x_i,y_i,z_i), a bounding box is defined as a box that contains all the points. Typically, a bounding box is a parallelpiped and is oriented with respect to the axes of the global coordinate system. To obtain a tighter bounding box, i.e., one with smaller volume, the bounding box can be oriented by using the three principal directions of the underlying point set. The coordinate system defined by these three principal directions allows us to obtain a bounding box that reflects the orientation of the data points. The orientation process consists of the following steps:

1. Given a set of points (x_i,y_i,z_i), compute the best-fit plane for the point set. The local coordinate frame in 3D space consists of three basis vectors and an origin. We use PCA to determine the basis vectors of the local frame [7].

2. The normal vector of the best-fit plane defines the ordinate direction in the local frame. We measure distances of points to the best-fit plane in ordinate direction. We express all points associated with a box in terms of its local coordinate frame, using the average of the points as local origin.

Figure 2 shows a set of points and its oriented bounding box. The error associated with an oriented bounding box is directly related to the size of the box in normal direction. Considering an oriented bounding box **B** whose side length in normal direction is **L**, all points within **B** are guaranteed to be within distance $\frac{L}{2}$ of the associated best-fit plane. This follows from the fact that the best fit plane associated with an oriented box is defined by the mean of the data points, and the worst-fit vector is determined by PCA.

This orientation algorithm does not work for data sets that do not contain enough points to determine a normal vector, e.g., data sets with one or two points. It also has problems with collinear or nearly collinear point sets one method for addressing this problem is to use the local frame from the node's parent to orient the local frame for the point set.

4. The Generalized Strip Tree

Our strip tree is a quadtree in 3D space whose nodes are oriented bounding boxes. Given a set of points, the root of the tree is a single, properly oriented bounding box that contains all points. A node in the tree is subdivided by first dividing its bounding box into four sub-boxes and then re-orienting these sub-boxes relative to the points inside them. As the strip tree is refined, the bounding boxes of the leaf nodes provide better approximations to the underlying points.[1]

Each node in the strip tree has four children, except the leaves. The children are referred to as the NorthWest (NW), NorthEast (NE), SouthWest (SW), and SouthEast (SE) children. Each node has at most one West, East, North, and

[1] We note that the resulting tree is similar to the BOXTREE of Barequet et al. However, we generate this tree in a top-down fashion, using principal directions of point subsets.

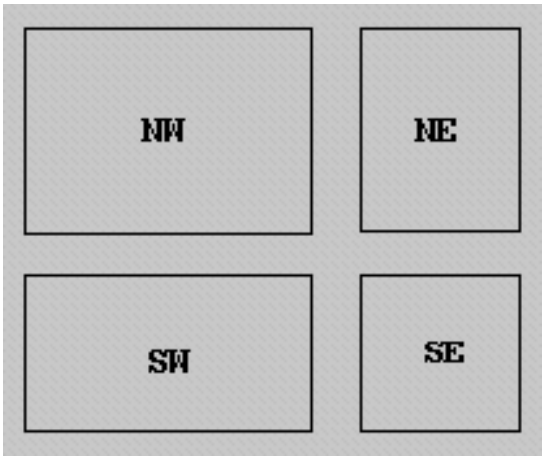

Figure 1. Arrangement of nodes used for patch blending.

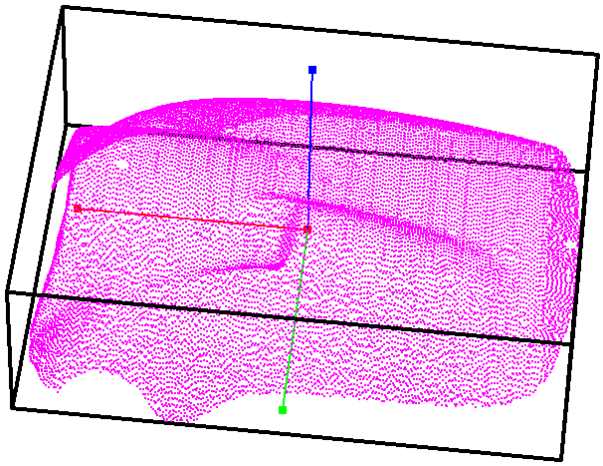

Figure 2. Root bounding box for scattered point set.

South neighbor. In our implementation, the strip tree is a *threaded tree*, which allows us to easily draw different levels of the tree and move between them. The arrangement of a node's children is shown in Figure 1. The decomposition of a box consists of four steps:

1. First, we determine a "subdivision point" within the bounding box around which to form the four sub-boxes. This subdivision point is defined as a given data point closest to the center of the bounding box. This point is used to divide the bounding box into four sub-boxes, all having parallel faces.

2. Once the four sub-boxes are determined, the data points contained in the original box are distributed among the four sub-boxes.

3. The four sub-boxes are re-oriented using the procedure outlined in Section 2.

4. The *box points* for the new, re-oriented sub-boxes are computed. (Box points are points that are not necessarily part of the given data set, but are required for later surface fitting.) The box points are either the four midpoints of the four box edges parallel to the normal vector of the best-fit plane or four original points closest to the midpoints of these edges. (Provided these are close enough to a box edge in normal direction). Figure 4 shows the box points of the leaf nodes for the strip tree shown in Figure 3. (The box points are drawn in black.)

The subdivision process is illustrated in Figures 2-5.

Figure 2 shows the root bounding box, and Figure 3 shows the strip tree after the root node has been subdivided and the bounding boxes have been re-oriented. Figure 4 shows the box points along with the bounding boxes for the four nodes of the strip tree. In this example, the box points are the four original points, considering each sub-node, that are closest to the edges of the bounding boxes in the direction of the best-fit normal vectors. (The normal vectors in these pictures are shown in blue.) Figure 5 shows the strip tree after four subdivision steps.

A bounding box is decomposed only when all of its four children can be successfully re-oriented. In cases where one or more of the new child nodes contain a degenerate set of points, as discussed in Section 4, the decomposition process fails and the bounding box is not subdivided.

5. Fitting Surfaces

Once the strip tree is decomposed so that the leaf nodes of the tree all have an error below a prespecified error threshold, the strip tree approximation is used to construct surfaces that approximate the data points in each box. To construct a least-squares B-spline surface to the points associated with a node, the node must first be subdivided. The approximation process for a node in the next-to-last level begins with a single bi-quadratic B-spline patch fitted to the box points of the node's children. The 3x3 control points required for a bi-quadratic B-spline patch are obtained from the four corner box points and averages of interior points. As an example, Figure 4 shows four leaf nodes and their 16 box points. This set of 4x4 points is reduced to a set of 3x3 points by keeping the four corner points, averaging pairs of control points in the middle of the outside edges, and

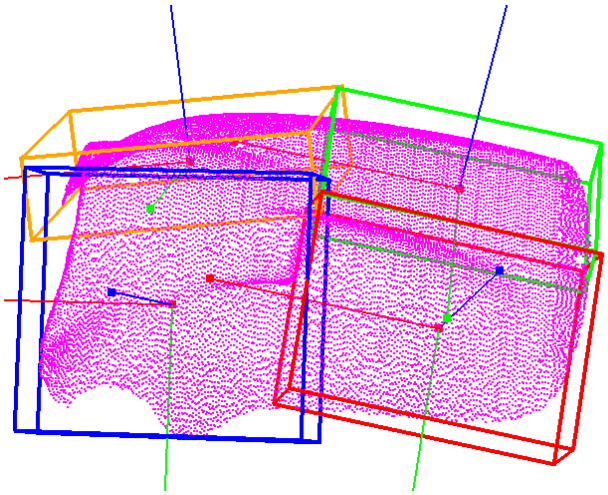

Figure 3. Strip tree after one subdivision step.

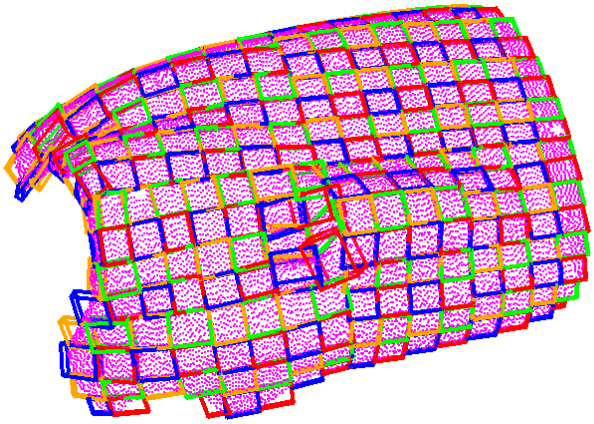

Figure 5. Strip tree after four subdivision steps consisting of 16x16 boxes.

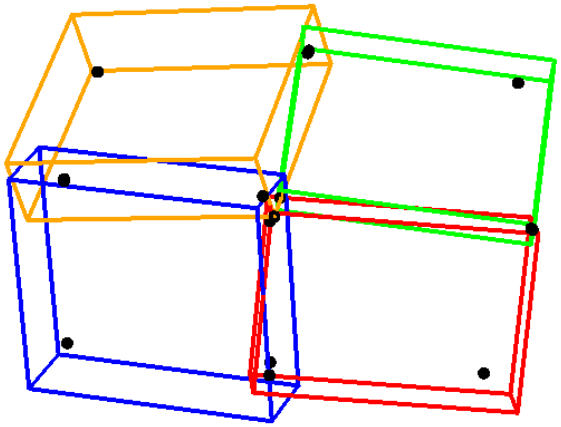

Figure 4. Box points for strip tree shown in Figure 3.

averaging the four control points in the interior. A least-squares bi-quadratic B-spline patch is fitted through the set of 3x3 control points. This surface is then degree-elevated to a bi-cubic B-spline patch. After the fitting process, the resulting surfaces are blended together to form a new set of C^1-continuous B-spline surfaces. The input to the blending process consists of four bi-cubic B-spline patches, following the arrangement of patches shown in Figure 1.

The blending process consists of three steps:

1. We force two surfaces to become C^0-continuous by requiring the last row/column of control points of the first surface and the first row/column of control points of the second surface to be equal. This is done averaging the corresponding points from each patch. At the point where four surfaces meet, the average of the four corner points is used.

2. The boundary curves of the surfaces are then made C^1-continuous by using the box points of the strip tree nodes to approximate the derivative of the data in the North/South and East/West directions. (These correspond to the u and v parametric directions of the B-spline surfaces.) The derivatives are approximated using central differences wherever possible and forward differences along the boundaries. The boundary control points of the surfaces are then adjusted acordingly.

3. Finally, the cross boundary derivatives or twist-vectors are made C^1-continuous by first computing the average of the twist vectors for the surfaces to be blended. The interior control points are then adjusted so that the twist of the surface is equal to this average.

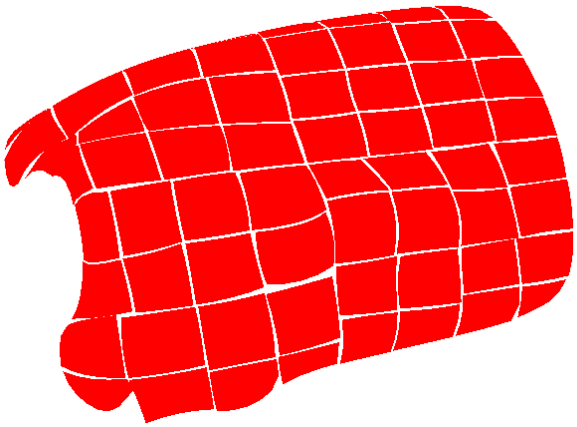

Figure 6. Initial surfaces for strip tree shown in Figure 5.

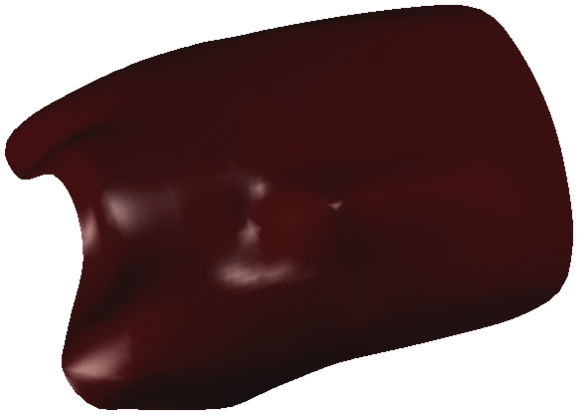

Figure 7. Final surfaces for strip tree shown in Figure 5

6. Error Calculations

The error in the resulting network of surfaces is a function of the size of the oriented bounding boxes in the direction normal to the surface and the error introduced by the blending process. It is given by the formula:

$$E_{STNode} = BoxSize_{NormalDirection} + max(E_{blending})$$

Given a single node in the strip tree with an oriented bounding box and an associated approximation surface, the error is the sum of the size of the bounding box in the normal direction and the error introduced by the blending process. The normal direction is the normal vector of the best-fit plane used to form the oriented bounding box as discussed in Section 3. The error introduced by the blending process is computed as the distance by which the new control points deviate from their original positions. The maximum over all of these values is used. The overall error for the approximation is the maximum error over all of the strip tree nodes that have a valid approximation surface.

7. Results

Figure 6 shows the inital set of independent surfaces created for the strip tree shown in Figure 5. Figure 7 shows the final B-Spline surface (Gouraud shaded) obtained by blending. The dataset used in these pictures is a skidoo dataset obtained from the web site of Hugues Hoppe (http://www.research.microsoft.com/˜ hoppe). The final approximation error for this model using 64 surfaces is about 2.7. Where the original x, y, and z coordinates of the data

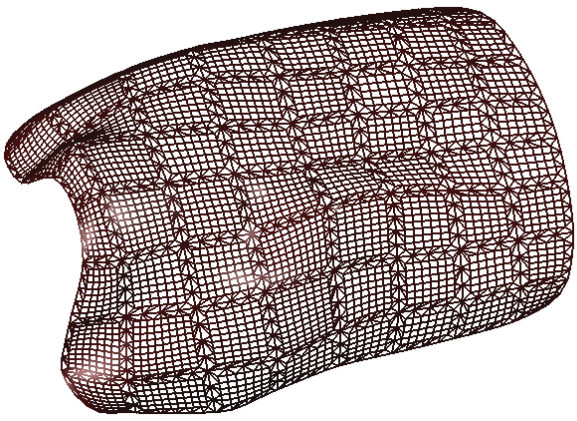

Figure 8. A wireframe rendering of Figure7.

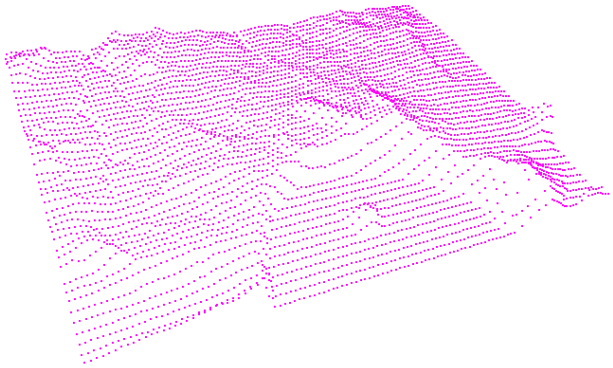

Figure 9. The data points for Crater Lake.

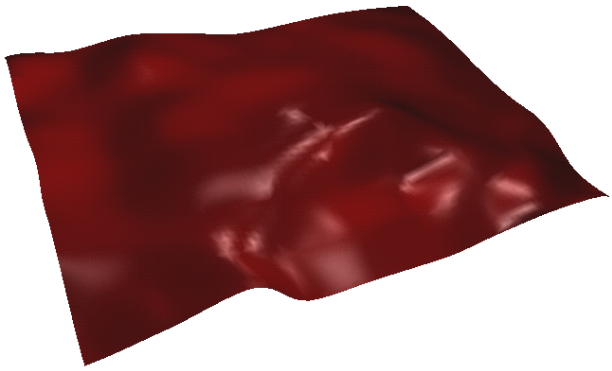

Figure 11. Final Surfaces for Figure 10.

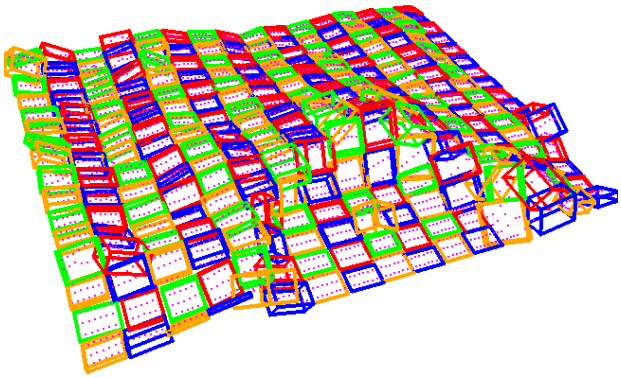

Figure 10. Strip Tree with 16x16 nodes.

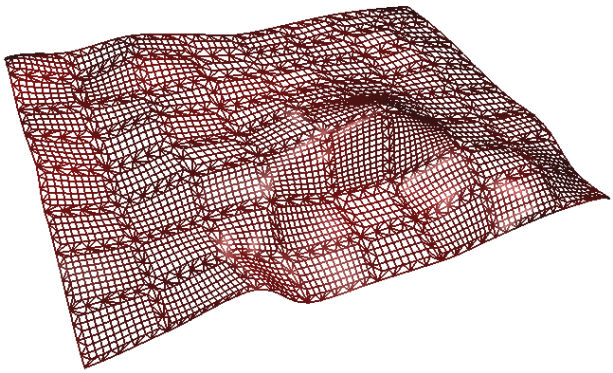

Figure 12. Wireframe rendering of Figure 11.

points vary in the intervals [-8.1 8.35], [-12.5 10.6], [-4.9 3.3] respectively. This means that the original data points are no more than 2.7 units away from the approximation. The majority of the error in the approximation is derived from the error in the blending process. This is because the blending process currently does not deal effectively with highly curves boundaries.

Figures 9-12 show the datapoints, strip tree, and final renderings for the crater lake dataset from the USGS. The final approximation error for this model using 64 surfaces is about 15. Where the original x, y, and z coordinates of the data points vary in the intervals [-51.7 , 51.6666], [-69.9 , 69.9], [-11.9 , 18.8] respectively. Figures 13-15 show the crater lake data set approximated with a (32x32) striptree resulting in 256 surfaces. The final approximation error is about 8.4.

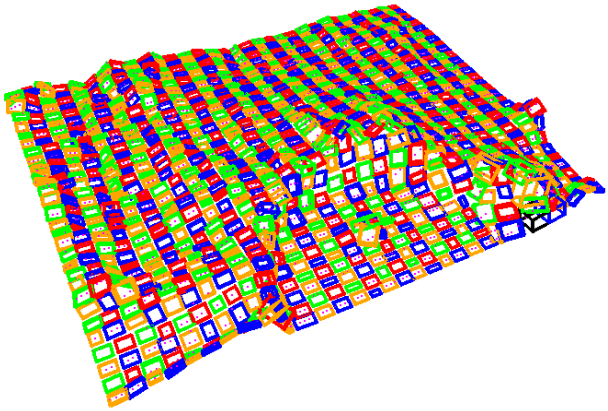

Figure 13. Strip Tree with 32x32 nodes.

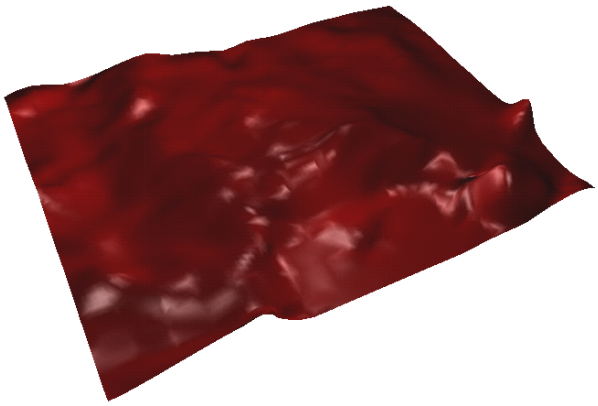

Figure 14. Final Surfaces for Figure 13.

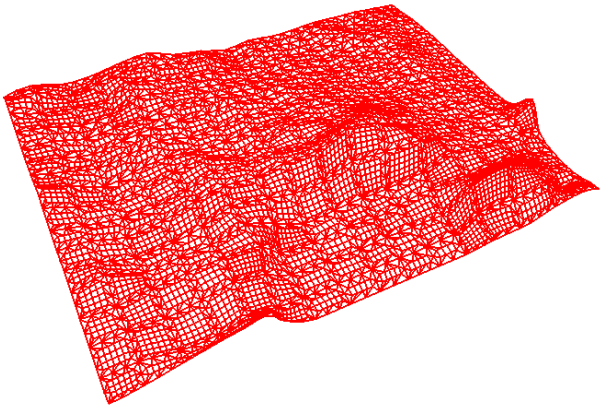

Figure 15. Wireframe rendering for strip tree in Figure 14.

8. Conclusions and Future work

We have presented a new method for reconstructing surfaces from scattered points. Our algorithm is based on a generalization of the strip tree used to approximate the given points initially. This initial approximation is used to construct a set of surfaces that approximates the given data points. Our algorithm works well on scattered data that represent smooth surfaces, and where smooth surfaces are desired as output. It does not work well for surfaces that are self-intersecting, or "twisting surfaces" The limitation is given by the structure of the oriented bounding boxes. When the oriented bounding boxes do not correspond to a single sheet of the underlying surface, the algorithm may produce erroneous results for the leaves of the tree.

At present, our algorithm subdivides a given point set in a uniform manner. The strip tree needs to be enhanced to allow for a non-uniform subdivision of the data in regions with more data points or more complicated, highly curved behavior. This will allow us to use more surfaces in areas with a higher sampling rate or higher curvature variation, and to approximate subtle features more adaptively. Furthermore the fitting and blending processes need to be improved for regions with higher curvature. The fitting process can be improved by using a weighted fitting process to better approximate sharper features. The blending process can be improved by finding better ways to approximate the derivatives of the data and to determine the final twist vectors. Sharper features could also be better approximated by relaxing the continuity constraints in certain regions. For example, regions with sharp creases, darts, or cliffs could be made C^0-continuous but not C^1-continuous. This would decrease the error of the approximation and at the same time lead to a better representation of the data.

Lastly the blending process needs to be improved to deal with holes in the dataset and datasets with highly curved boundaries. Holes in the dataset could be distinguished by letting the user specify a "hole size" that determines whether two data points lie across a hole. In these situation, patches that lie across a hole would not be be blended together. Curved boundaries could be dealt with by only blending patches whose edges are within a certain distance of each other.

9. Acknowledgements

This work was supported by the National Science Foundation under contract ACI 9624034 (CAREER Award), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the Office of Naval Research under contract N00014-97-1-0222; the Army Re-

search Office under contract ARO 36598-MA-RIP; the NASA Ames Research Center through an NRA award under contract NAG2-1216; the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159; and the North Atlantic Treaty Organization (NATO) under contract CRG.971628 awarded to the University of California, Davis. We also acknowledge the support of ALSTOM Schilling Robotics, Chevron, Silicon Graphics, Inc. and ST Microelectronics, Inc. We thank the members of the Visualization Thrust at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis.

References

[1] G. Barequet, B. Chazelle, L. J. Guibas, J. S. B. Mitchell, and A. Tal. BOXTREE: A hierarchical representation for surfaces in 3D. *Computer Graphics Forum*, 15(3):387–396, 1996.

[2] M. Eck and H. Hoppe. Automatic reconstruction of b-spline surfaces of arbitrary topological type. In *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 325–334. ACM SIGGRAPH, Addison Wesley, 1996.

[3] G. Greiner and K. Hormann. Interpolating and approximating scattered 3D-data with hierarchical tensor product splines. In A. L. Mehaute, C. Rabut, and L. L. Schumaker, editors, *Surface Fitting and Multiresolutional Methods*, pages 163–172. Vanderbilt University Press, 1996.

[4] B. Heckel, A. E. Uva, and B. Hamann. Clustering-based generation of hierarchical surface models. In *Proceedings IEEE Visualization '98 – Late Breaking Hot Topics*, 1998.

[5] B. Heckel, A. E. Uva, B. Hamann, and K. Joy. Surface reconstruction using adaptive clustering methods. In *IEEE Transactions on Visualization and Computer Graphics*, 2000.

[6] H. Hoppe. *Surface reconstruction From Unorganized Points.* PhD thesis, Department of Computer Science and Engineering, University of Washington, 1994.

[7] J. E. Jackson. *A User's Guide to Principal Components*. A Wiley-Interscience Publication, John Wiley and Sonc, Inc., New York, 1991.

[8] V. Krishnamurthy and M. Levoy. Fitting smooth surfaces to dense polygon meshes. In *SIGGRAPH 96 Conference Proceedings*. ACM SIGGRAPH, Addison Wesley, 1996.

[9] H. H. T. D. T. D. J. McDonald and W. Stuetzle. Surface reconstruction from unorganized points. In *SIGGRAPH 92 Conference Proceedings*, Annual Conference Series, pages 71–78. ACM SIGGRAPH, Addison Wesley, 1992.

[10] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, Reading, Massachusetts, 1990.

[11] H. Samet. *Introduction to Spatial Data Structures*. Computer Graphics, Image Processing, and GIS-Addison-Wesley, Reading, Massachusetts, 1990.