Modeling Physical Fields for Interrogative Visualization*

Chandrajit L. Bajaj

Department of Computer Sciences
Purdue University
West Lafayette, Indiana

http://www.cs.purdue.edu/research/shastra/shastra.html
bajaj@cs.purdue.edu

Abstract

Interrogative visualization refers to the process of interactive computer graphics display and accurate quantitative querying of physical data. Quantitative querying includes search for metric, combinatorial and topological information. To support this paradigm, we build uniform, compact, co-registered representations (spline models) of multiple physical data fields over the same domain.

Dense, unstructured volumetric scalar fields are approximated by C^1-continuous trivariate polynomial spline functions of low degree. Additionally, these spline functions are also used to model scattered scalar fields sampled over a manifold surface in the volume. Using both implicit polynomial spline surfaces and trivariate polynomial spline functions allows for model representations of both manifold and associated scalar fields over the same spatial decomposition. Quantitative querying is made efficient by utilizing various search structures over the modeled field data.

1 Introduction

In many scientific applications one or more physical quantities are measured (or computed, in the case of a simulation) at a large number of points scattered in space (*volumetric data*), or on the surface of an object (*manifold data*). Here, I provide a brief review of some piecewise polynomial approximation algorithms used for the construction of spline models of both scattered volumetric and manifold [2, 3, 7]. I give some details of tensor-product Bernstein-Bézier (BB) polynomial constructions used both as implicit surface splines, as well as trivariate polynomial function splines, and defined over a single spatial octree decomposition [2]. Tensor-product B-splines or Box-splines could also be used [16].

Scattered data, as opposed to data associated with the nodes of a regular grid, pose challenging visualization problems. Standard techniques for direct volume visualization, isosurface extraction etc. rely on the existence of a grid of known topology [37]. One common approach to the modeling of scattered data in any dimension is based on triangulating the data points, simplifying the triangulation to reduce the size of the model, and then constructing a suitable interpolant on each triangle (or higher-dimensional simplex) of the reduced model, additionally approximating the original data [1, 29, 30]. The initial triangulation and post simplification can become quite expensive. Alternatively, spatial decomposition techniques can be used to approximate a *dense* set of multiple volumetric scattered scalar values with C^1-continuous spline functions of

*Supported in part by NSF grant CCR 92-22467, AFOSR grant F49620-94-1-0080, ONR grant N00014-94-1-0370.

low degree. The approach is incremental and adaptive, with the spatial decomposition being refined where needed to achieve a desired error bounded approximation. Such trivariate polynomial spline approximation schemes for scattered volumetric and manifold scalar field data, built over 3D incremental Delaunay triangulations are reported in [7, 3]. Here I report on a similar scheme for volumetric data using tensor-product BB polynomial splines over octree decompositions of a cuboid enclosing the data. Additionally, I review the modeling of manifolds and associated scalar field data using both implicit tensor product BB polynomial spline surfaces and trivariate tensor product BB polynomial spline functions over the same spatial octree decomposition [2].

The main goals achieved by such an adaptive modeling technique are: (i) compact adaptive representation of domains and associated physical fields and (ii) easy visualization, and interactive querying of physical fields. Additionally, multiple physical fields for which different data samples are available, can be modeled by splines defined over the same spatial decomposition, allowing for easy co-registration of the data. Incremental regular decompositions, such as octrees, provide a natural search structure as well as a hierarchy of approximating spline models. Hierarchical spline representations provide interrogative visualization at multiple approximation levels[20]. Seed cell pre-computations coupled with interval search trees provide fast multiple query times, such as for isocontouring [8]. Isocontouring and local normal projection methods with color plots can be used to display fields over a manifold. Parallel implementations of such mphysical field odeling and visualizations are also easy [6]. Availability of a compact model for multiple, co-registered scalar and vector fields (and for the geometry of the manifold) allows accurate quantitative queries of combinatorial, metric and topological properties of the fields. The C^1 spline model additionally supports computation of smooth navigation paths in the field data.

An overview of the volumetric data reconstruction algorithm is presented in Section 2. In Section 3 we show how the manifold data reconstruction problem can be reduced to a volumetric data modeling problem. Details on the mathematics involved are given in Section 4. Several examples and results are shown and discussed in Section 5. All proofs of the lemmas and theorems can be found in [2].

2 Modeling of volumetric data

There is a large body of literature on modeling of volumetric data. Reviews of prior work in the field can be found in [1, 5, 21, 30, 31]. We first describe how to use tensor-product polynomial splines to model dense, scattered volumetric data. For the sake of simplicity, we describe our method for the reconstruction of a single, scalar field. Multiple fields can be simultaneously represented by replacing the scalar weights that define the spline model with *vectors* of weights.

Our problem can be formally stated as follows:

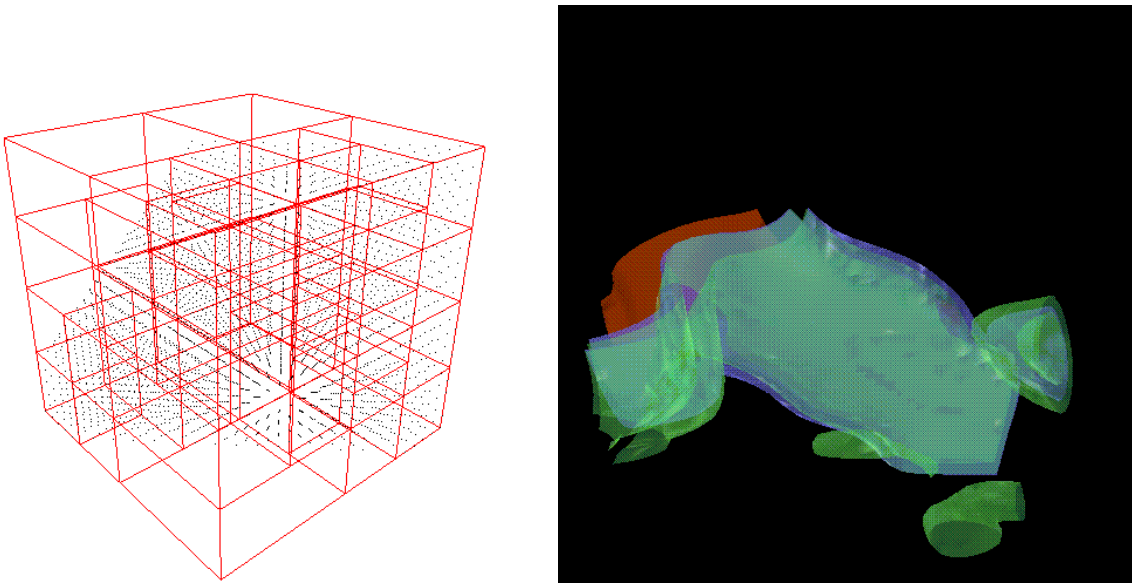
Definition 2.1 *Given a set of points $P = \{p_i\}_{i=1}^N \subset \mathbf{R}^3$ and associated data values $V = \{v_i\}_{i=1}^N$, build a trivariate, C^1-continuous, piecewise polynomial function $W(x, y, z)$ that approximates the data within a given error bound ϵ, i.e. such that $|W(p_i) - v_i| < \epsilon$, for $i = 1, \ldots, N$.*

The technique starts with the construction of a cuboid (box) containing the data set, and an initial coarse partition of it into smaller boxes. For each vertex of the grid, we compute a local trilinear *approximant*, based on nearby data points. These approximants define a function value and an associated derivative jet at each vertex of the grid. In each box we compute a tricubic (or triquadratic) polynomial that interpolates the vertex data, and such that adjacent functions join with C^1 continuity. The error of this approximation at the i-th data point is given by the difference between v_i and the value of the spline at p_i. If this error is too large, we proceed to a local refinement, by splitting the box into eight equal sub-boxes. The process continues until the specified error condition is satisfied.

Inside each box, the polynomial is represented in Bernstein-Bézier form (BB-form). For $(x, y, z) \in D = [a_1, a_2] \times [b_1, b_2] \times [c_1, c_2]$, a BB-form (m, n, q)-polynomial is defined as

$$W(x, y, z) = \sum_{i=0}^{m} \sum_{j=0}^{n} \sum_{k=0}^{q} w_{ijk} B_i^m(u(x)) B_j^n(v(y)) B_k^q(w(z))$$

where $B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$,, $u(x) = \frac{x-a_1}{a_2-a_1}$, $v(y) = \frac{y-b_1}{b_2-b_1}$ and $w(z) = \frac{z-c_1}{c_2-c_1}$. A tricubic BB polynomial



(a) (b)

FIGURE 1: Example of reconstruction from volumetric data. (a) Data points and final octree. (b) Isosurfaces extracted from the piecewise polynomial spline model.

(see Figure 3) is defined by 64 coefficients (or *weights*), associated with the vertices of a regular, $4 \times 4 \times 4$ grid.

Let $D = [\alpha_1, \alpha_2] \times [\beta_1, \beta_2] \times [\gamma_1, \gamma_2]$ be a cuboid containing the data set P. The outline of the algorithm is as follows:

Begin Algorithm

1. **Initial Partition.** Construct a partition of D. That is choose a_i, b_j, c_k so that

$$\begin{array}{ccccccccccc} \alpha_1 & = & a_0 & < & a_1 & < & \ldots & < & a_{\ell_1} & = & \alpha_2 \\ \beta_1 & = & b_0 & < & b_1 & < & \ldots & < & b_{\ell_2} & = & \beta_2 \\ \gamma_1 & = & c_0 & < & c_1 & < & \ldots & < & c_{\ell_3} & = & \gamma_2 \end{array}$$

   and partition D by $D = \bigcup D_{ijk}$, with

$$D_{ijk} = [a_{i-1}, a_i] \times [b_{j-1}, b_j] \times [c_{k-1}, c_k], \text{ and}$$

$$i = 1, \ldots, \ell_1, j = 1, \ldots, \ell_2, k = 1, \ldots, \ell_3.$$

   Mark all elements and vertices in the mesh as *new*.

2. **Local Fitting.** For every *new* vertex q in the mesh, compute a local tri-linear approximant $l(x, y, z)$, by least-squares fitting of the data in a neighborhood of q.

   Compute the following *derivative jet* by evaluating at q the function $l(x, y, z)$ and its derivatives:

$$l, \frac{\partial l}{\partial x}, \frac{\partial l}{\partial y}, \frac{\partial l}{\partial z}, \frac{\partial^2 l}{\partial x\, \partial y}, \frac{\partial^2 l}{\partial x\, \partial z}, \frac{\partial^2 l}{\partial y\, \partial z}, \frac{\partial^3 l}{\partial x\, \partial y\, \partial z}.$$

   The vertex is then marked as *old*.

3. C^1 **interpolation.** For each *new* element D_{ijk} in the mesh, compute a tricubic BB-polynomial $W_{ijk}(x, y, z)$ that interpolates the derivative jet at all its eight vertices, and such that the piecewise polynomial $W(x, y, z)$ is C^1 continuous over all D. The element is then marked as *old*.

   Details on how to compute the coefficients of W_{ijk} are given in Section 4. We show there how to create additional *degrees of freedom*, by splitting each box into eight sub-boxes, to be used to achieve a better local fitting to the data. An alternative, tri-quadratic interpolant is also described.

4. **Adaptive step.** For each element for which a local interpolant has been computed in the previous step, compute an *approximation error*, as the maximum algebraic distance between the function and the data value at data points, $e = \max_{p_\ell \in D_{ijk}}(|W_{ijk}(p) - v_\ell|)$. If $e > \epsilon$ then subdivide D_{ijk} into eight sub-boxes. Mark the sub-boxes and their vertices as *new*. Then return to step 2.

End Algorithm

Notice that the adaptive refinement of the subdivision can be represented as an octree search structure (see e.g. [33]). In the following we refer to the *level* of a box with the following meaning: boxes at level 1 are those belonging to the coarser subdivision. Boxes at level $i + 1$ have been obtained subdividing a box at a level i into eight sub-boxes.

Two adjacent boxes share the common face in one of the following two fashions:

1. The two faces coincide completely.

2. One contains the other as a proper subset.

In step 4, eight new sub-boxes and seven new vertices are created (one for each face plus one in the middle of the original box). For the vertex shared by all sub-boxes, we always compute the derivative jet from the local approximant as described in step 2. For the six "face" vertices, there are two possible cases, depending on whether the box adjacent along that face has also been split or not. In the first case, the vertex has no data associated with it, and new data is computed in step 2 of the algorithm. In the second case, the "face" vertex is shared with an adjacent box, for which the C^1 interpolant has already been computed. In this case the derivative jet is computed evaluating this interpolant, to guarantee the global C^1 continuity. Figure 1 shows an example of reconstruction from volumetric data.

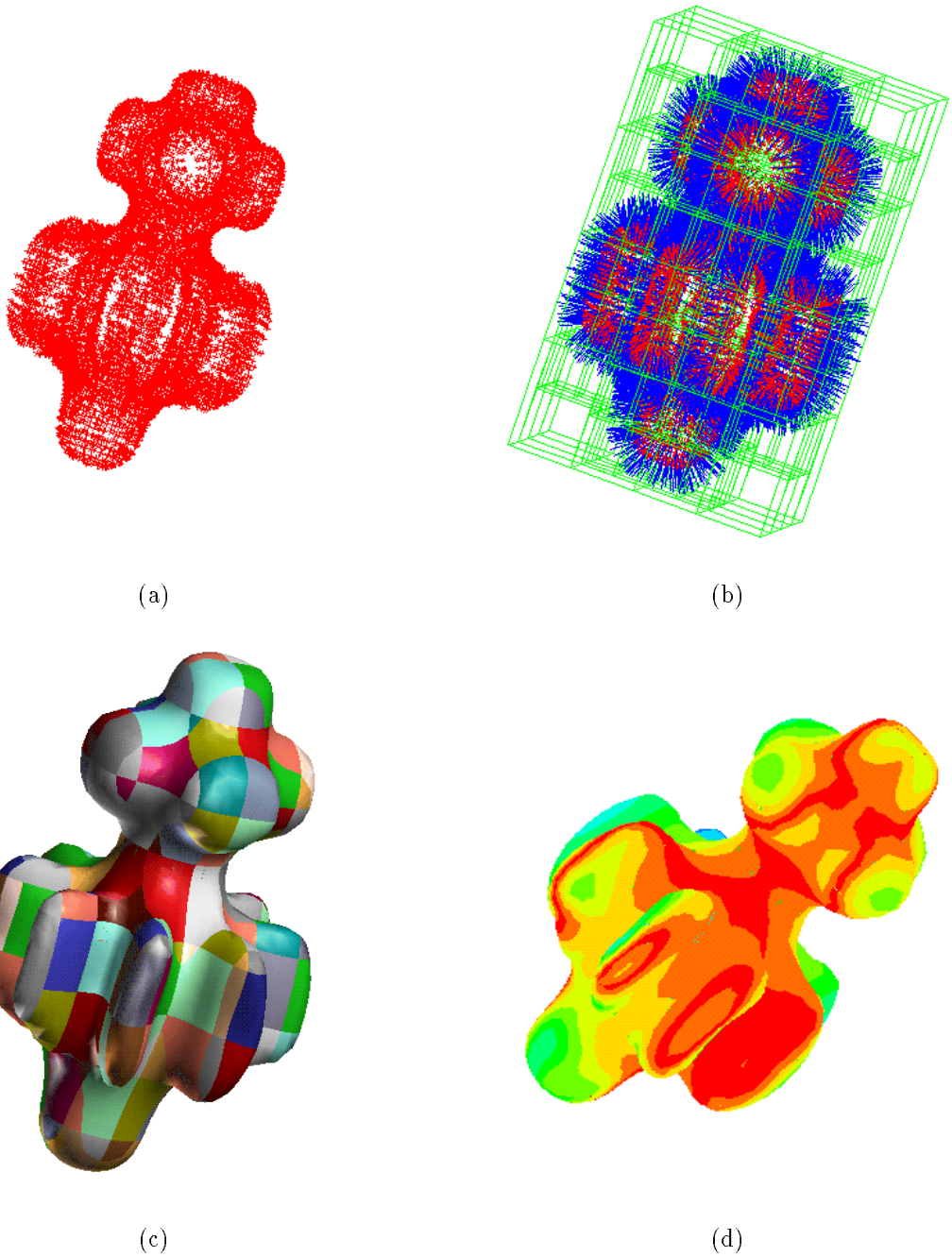
3 Modeling of Manifold Data

We consider the problem of reconstructing both the surface of an object and a scalar field on it from dense, scattered data. We assume that both the surface and the field are continuous and have continuous first-order derivatives. Since data measurements are subject to error, we do not try to exactly interpolate the data but rather approximate it within a given tolerance. The problem may be formally stated as follows:

Definition 3.1 *Given a set of dense scattered points $P = \{p_i\}_{i=1}^N \subset \mathbf{R}^3$ on an unknown manifold $\mathcal{M}$, and associated data values $V = \{v_i\}_{i=1}^N$, construct a smooth surface $\mathcal{S} : S(x, y, z) = 0$, such that $\mathcal{S}$ approximates P within a given error bound ϵ_S, and a smooth function $\mathcal{F} : F(x, y, z)$, such that $\mathcal{F}$ approximates the data V associated with P within a given error bound ϵ_F.*

The problem of reconstructing the approximation $\mathcal{S}$ to the unknown manifold $\mathcal{M}$ has attracted the interest of many authors (for a review, see [11]). Several papers focus on building a piecewise-linear surface [12, 13, 18, 25, 36]. Other methods use parametric or functional surface patches for either local or global interpolation [9, 22, 24, 27, 32, 34]. A few papers (see [3, 14, 15, 23, 28, 35]) use implicit surface patches. In this paper, we use an implicitly defined tensor-product polynomial spline surface to approximate the unknown surface $\mathcal{M}$.

The problem of interpolating or approximating data defined over a *given* manifold in $\mathbf{R}^3$ is commonly referred to as *modeling 3D scattered manifold data* or the *function-on-surface* problem [30]. Prior work on this subject includes [1, 9, 10, 19, 21, 31].



(a) (b)

(c) (d)

FIGURE 2: Reconstruction of a surface and an associated scalar field from scattered data: (a) Input points. (b) Orientation of normals and octree subdivision. (c) Polynomial Spline approximation. (d) Reconstructed scalar field.

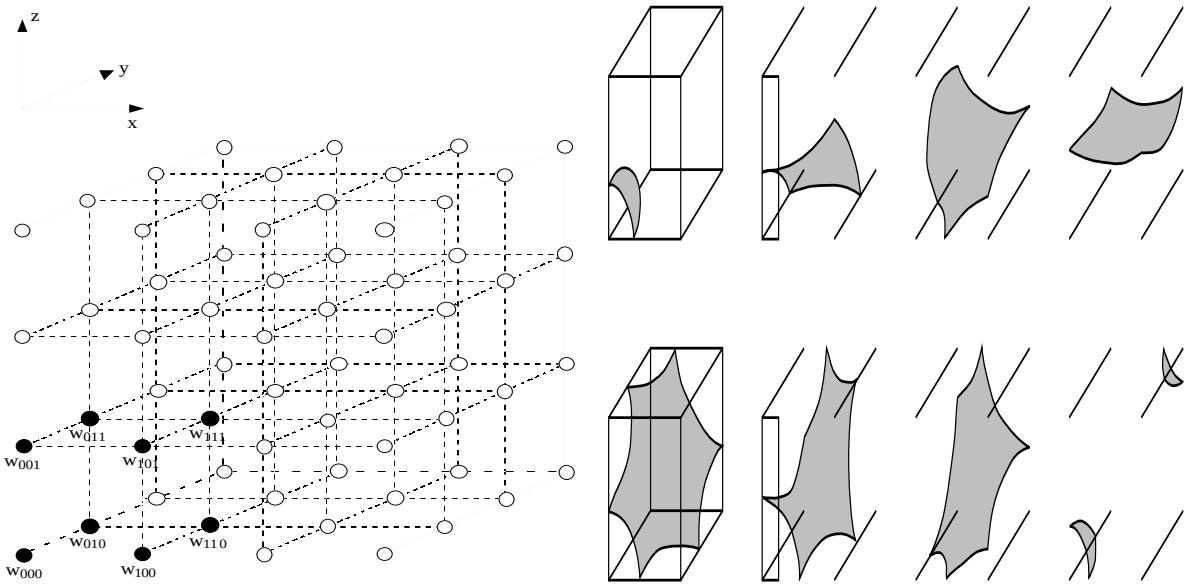


FIGURE 3: (left) The sixty-four weights defining a tricubic polynomial in BB form. The filled dots correspond to weights that are determined by the derivative jet at a vertex. (right) The eight admissible cases for the signed-distance function δ. The sign of δ on the sixty-four control points is uniform in each region delimited by the shaded surface, and changes across it.

3.1 From surface data to volume data: The signed-distance function

If $\mathcal{M}$ is a connected and orientable surface in $\mathbf{R}^3$, then it is possible to define (in all $\mathbf{R}^3$) a function $\delta(q)$, called signed-distance, by

$$\delta(q) := sign \cdot dist(q, \mathcal{M})$$

where $dist(q, \mathcal{M})$ denotes the Hausdorff distance from the point q to the surface $\mathcal{M}$ and the *sign* is chosen so that $\delta(q)$ is positive when q is outside $\mathcal{M}$, and negative when q lies inside it. Then $\delta(q) = 0$ will recover the surface $\mathcal{M}$.

When only discrete data on a surface is available, an approximate signed-distance can be defined in some appropriate way (see e.g. [3, 25, 28]). Given the signed-distance function, one can approximate it with a piecewise polynomial function $S(x, y, z)$ (in a suitable domain containing P), and then extract the zero-contour of S.

Using the signed-distance function to reconstruct a surface from scattered data points has been considered by several authors.

Moore and Warren [28] use a tetrahedral decomposition of the space, and reconstruct the surface by implicit barycentric Bernstein-Bézier patches. For each tetrahedron, they compute the associated surface patch by least-squares fitting of the data points and samples of a "local" signed-distance function. They partition the tetrahedron into a regular mesh of tetrahedra, and compute the distance of each vertex of the mesh to the closest data point. If the data points are dense enough, then the tetrahedra of the mesh containing at least one point form a "layer" that partitions the mesh into two components (one on each side of the layer), to which two opposite signs can be assigned arbitrarily. This sign is associated to the distance values computed for each vertex. This dense-sampling assumption can be too restrictive in some practical cases.

Hoppe et al. [25] use a more global approach to correctly orient the approximated manifold. First, for each data point p_i, they compute a best fit plane, and the associated normal $\hat{n}_i$, based on best-fit of k neighbor points (the *k-neighborhood* of p_i). The problem is now assigning a consistent orientation to these planes. They build the *Riemannian Graph*, $RG(P)$ over P (two points $p_i, p_j \in P$ are connected by an

edge in $RG(P)$ iff either p_i is in the k-neighborhood of p_j or p_j is in the k-neighborhood of p_i). Each edge (i,j) is assigned the weight $1 - |\hat{n}_i \cdot \hat{n}_j|$, and a minimum spanning tree is computed. Intuitively, this tree connects points that have close-to-parallel associated planes. They then orient the plane associated with the point with the largest z-value so that its normal points toward the positive z-direction, and propagate this orientation to other points traversing the minimum spanning tree. Their method continues with the construction of a regular subdivision of a cuboid containing the data points into boxes. The value of δ is computed at all vertices of the subdivision as the signed-distance of the vertex from the oriented plane associated with the closest point in P. An algorithm similar to *marching cubes* is then used to construct a piecewise-linear approximation of the zero contour of δ. In two subsequent steps, described in [24, 26], the constructed mesh is optimized (i.e., the number of triangles is reduced while the distance of the mesh from the data points is kept small) and then a smooth surface is built on it. While this approach gives very convincing results, and allows for smooth objects with sharp features to be correctly reconstructed, the computational time required by the optimization and smoothing steps can be significant.

Bajaj et al. [3] propose the use of α-shapes [17] to build a piecewise-linear approximation of the surface being reconstructed. The α-shape is a sub-complex of the Delaunay triangulation of the set of points P. This approach has the advantage of being based on a sound mathematical definition of the *shape* of a set of points. The piecewise-linear approximation is then used to compute the value of the signed-distance function at any point. A piecewise polynomial approximation is subsequently built on a Delaunay 3D triangulation of a domain containing P.

3.2 Implicit tenosr-product polynomial spline reconstruction of the manifold

The manifold data modeling algorithm, is reduced to the problem of volumetric data modeling by constructing the approximate signed-distance function. In our implementation we have used both the normal propagation approach [25] and the weighted α-shapes [17]. We then use the adaptive algorithm described in Section 2 to approximate the signed-distance function, whose value is known at data points (where it is obviously zero) and can be computed (approximately) at any other point. Once the signed-distance function has been approximated to the given level of accuracy, its zero contour is used as an implicit representation of the unknown manifold. At the same time, a least squares approximation of the field defined over the manifold can be computed. In Figure 2 the phases of the reconstruction process are shown. The advantage of using implicit tensor-product splines is that we can construct both the manifold and the associated scalar field over the same domain.

Different from the volumetric data modeling case, an important issue here is guaranteeing that the resulting patches are smooth and single-sheeted. For this, a theorem similar to Theorems 3.2 and 3.3 in [4] for a tetrahedron can be established for a tensor-product scheme:

Theorem 3.1 *Let $P_{ijk} = P \cap D_{ijk}$. If there exists a plane $\pi(x,y,z) = 0$ such that all points of P_{ijk} lie within a distance ϵ from the plane, for a sufficiently small ϵ, then the zero contour of the local fitting $S_{ijk}(x,y,z)$ is smooth, single sheeted in D_{ijk} and lies within some distance, depending only on ϵ and the degree of S_{ijk}, from the plane $\pi(x,y,z) = 0$.*

In [32] it is has been pointed out that if all the weights increase or decrease monotonically along one of the coordinate directions, then straight lines parallel to that direction intersect the surface patch at most once. A less constrained condition is the following:

Theorem 3.2 *Given a BB-form (m,n,q)-polynomial, if there exists an integer l $(0 < l < m)$ such that*

$$w_{ijk} \leq 0,\ i = 0,\ldots,l-1;\ j = 0,\ldots,n;\ k = 0,\ldots,q$$
$$w_{ijk} \geq 0,\ i = l+1,\ldots,m;\ j = 0,\ldots,n;\ k = 0,\ldots,q$$

and there is at least one strict inequality in each set of inequalities, then straight lines parallel to the x-direction intersect the surface patch exactly once. Similar conclusions hold for the y and z-direction intersections.

These theorems can be used as follows: the volumetric data approximation algorithm adaptively subdivides boxes based on the local fitting error. When dealing with manifold data, one adds the constraint that

the conditions stated in Theorem 3.2 are satisfied. In each box, after the weights have been computed, we check whether the weights of the signs have one of the eight topologies depicted in Figure 3. If this is true, then the patch is non-singular and single-valued along each of the principal axes directions. We subsequently compute the approximation error (w.r.t. the data points), to decide whether to split the box or not. If the weight-signs test fails, the box is split and the approximation process is repeated in each sub-box.

4 C^1 Interpolation by tri-cubic or tri-quadratic polynomials

In the overview of our reconstruction algorithm in Section 2 we mentioned that at each vertex q of the octree-like mesh we compute a tri-linear approximant $l(x, y, z)$ by least-squares fitting of the data in a neighborhood of the vertex. For example, one could take the k-closest points to q, or take all the points contained in a box or sphere centered around q. The number of points to consider, or the size of the box or sphere, depend on the particular data set. We then compute the following *derivative jet*,

$$l, \frac{\partial l}{\partial x}, \frac{\partial l}{\partial y}, \frac{\partial l}{\partial z}, \frac{\partial^2 l}{\partial x\,\partial y}, \frac{\partial^2 l}{\partial x\,\partial z}, \frac{\partial^2 l}{\partial y\,\partial z}, \frac{\partial^3 l}{\partial x\,\partial y\,\partial z}. \tag{1}$$

and build an interpolant in each element of the mesh. This Section shows a way of constructing tri-cubic and tri-quadratic interpolants over a box, given the above derivative jet at all its eight vertices, so that the composite function is C^1 continuous. All proofs of the lemmas and theorems stated in this Section can be found in [2]. The following Lemma tells us how to compute the coefficients around a vertex from the derivative jet there.

Lemma 4.1 *Let*

$$W(x, y, z) = \sum_{i=0}^{m} \sum_{j=0}^{n} \sum_{k=0}^{q} w_{ijk} B_i^m(u) B_j^n(v) B_k^q(w),$$

for $m > 0,\ n > 0,\ q > 0$, be a BB-form polynomial on the box $D = [a_1, a_2] \times [b_1, b_2] \times [c_1, c_2]$. Then W interpolates the derivative jet (1) at the vertex (a_1, b_1, c_1) if and only if

$$w_{000} = l \tag{2}$$

$$w_{100} = w_{000} + \frac{a_2 - a_1}{m} \frac{\partial l}{\partial x} \tag{3}$$

$$w_{010} = w_{000} + \frac{b_2 - b_1}{n} \frac{\partial l}{\partial y} \tag{4}$$

$$w_{001} = w_{000} + \frac{c_2 - c_1}{q} \frac{\partial l}{\partial z} \tag{5}$$

$$w_{110} = w_{010} + w_{100} - w_{000} + \frac{(a_2 - a_1)(b_2 - b_1)}{mn} \frac{\partial^2 l}{\partial x \partial y} \tag{6}$$

$$w_{101} = w_{100} + w_{001} - w_{000} + \frac{(a_2 - a_1)(c_2 - c_1)}{mq} \frac{\partial^2 l}{\partial x \partial z} \tag{7}$$

$$w_{011} = w_{010} + w_{001} - w_{000} + \frac{(b_2 - b_1)(c_2 - c_1)}{nq} \frac{\partial^2 l}{\partial y \partial z} \tag{8}$$

$$w_{111} = w_{110} + w_{101} + w_{011} - w_{001} - w_{010} - w_{100} + w_{000} + \frac{(a_2 - a_1)(b_2 - b_1)(c_2 - c_1)}{mnq} \frac{\partial^3 l}{\partial x \partial y \partial z} \tag{9}$$

Similar conclusions hold for the other 7 vertices of the box D.

Tri-cubic interpolation. We can use tri-cubic polynomials (see Figure 3) to interpolate the derivative jet at all eight vertices and satisfy the C^1 continuity constraints, as stated by the following

Theorem 4.1 *If two tri-cubic BB-form polynomials W_1 and W_2 interpolate the derivative jet (1) at the common four vertices of two adjacent boxes D_1 and D_2, then W_1 and W_2 are C^1 continuous on the common face of D_1 and D_2.*

As the last Theorem states, if a volume consists of boxes, and if a tri-cubic BB polynomial on each box is constructed from the derivative jet at the vertices by formulae (2)—(9), then the composite function is C^1 continuous. The proof of the theorem above also shows that $m = n = q = 3$ is the minimal degree for forming C^1 piecewise functions, since there is no degree of freedom left. If a lower degree polynomial is used or some degrees of freedom are required, one has to subdivide each box into smaller sub-boxes.

Creating degrees of freedom by using tri-cubic interpolation. As mentioned in Section 2, the purpose of creating degrees of freedom is to achieve a better approximation to the data. For a given box D with derivative jets on its eight vertices, the interpolating tri-cubic W_D is uniquely defined as described above. To create some degrees of freedom, we subdivide the box D into eight equally-sized sub-boxes. The derivative jet at the center c of the box D is free. This means we can freely choose eight weights around c (the value of the other 19 is determined by the C^1 continuity conditions). We can determine the values to assign to the eight weights by least-squares fitting of data values within the box. Of course, this subdivision process can be repeated if necessary. The composite function is always C^1 continuous.

Tri-quadratic interpolation. C^1 interpolation of the derivative jet can also be achieved by a tri-quadratic piecewise polynomial. However, this requires splitting each box in eight sub-boxes to create additional degrees of freedom in the choice of coefficients.

The following theorem guarantees that by subdivision, the tri-quadratic interpolants over the eight sub-boxes exist uniquely and the composite function is C^1 continuous.

Theorem 4.2 *Let $D = [a_1, a_2] \times [b_1, b_2] \times [c_1, c_2]$ be a given box. At each of its eight vertices P_{rst}, we are given the derivative jet (1). If we subdivide D into eight sub-boxes D_{rst}, then there exists uniquely one piecewise polynomial function W on D such that*

a. $W_{rst} = W|_{D_{rst}}$ is a (2,2,2)-polynomial, that interpolates the derivative jet at P_{rst}.

b. W is C^1 continuous on D.

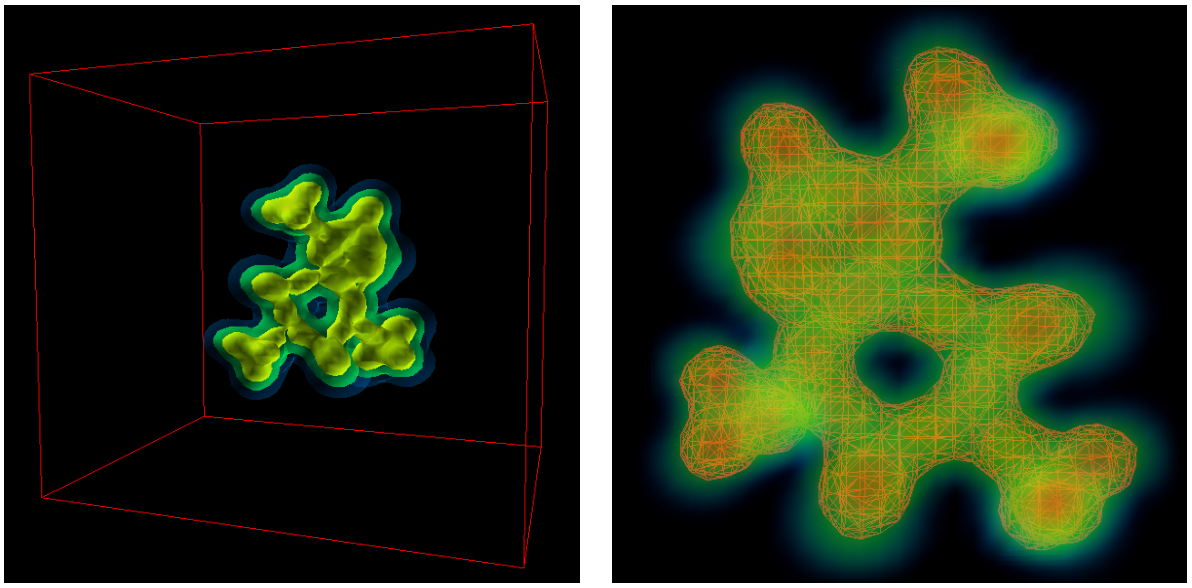
c. If W' is defined in the same way over an adjacent box D', then W and W' are C^1 continuous on the common face of D and D'.

Unlike the tri-cubic interpolation, the subdivision does not lead to extra degrees of freedom.

5 Visualization of Multiple Data Sets

The reconstructed spline model can be used to extract polygonal isosurfaces that can be directly rendered. In Figure 4, two examples of isosurfaces visualizations are shown. The data is a scattered volumetric sampling of the electrostatic potential function for a caffeine molecule.

When dealing with manifold data, one often wants to visualize the reconstructed manifold $\mathcal{S}$ (domain) together with the associated fields. A simple way of doing this is by coloring each pixel of the domain surface with a color proportional to the field value at that point, according to a predefined colormap. Alternatively, one might draw isocontours of the field over the manifold. Another popular method consists in displaying the field as a *function-on-surface*, i.e. a surface obtained by projecting the field from $\mathbf{R}^4$ to $\mathbf{R}^3$. One approach is to use a radial projection from the center of the domain surface $\mathcal{S}$. However, if the domain surface $\mathcal{S}$ is not convex or has non-zero genus, this method has serious difficulties. Another more natural way is to use a



(a) (b)

FIGURE 4: Isosurfaces from a reconstructed piecewise polynomial function. The input scattered volumetric data represent values of the electrostatic potential function for the caffeine molecule.

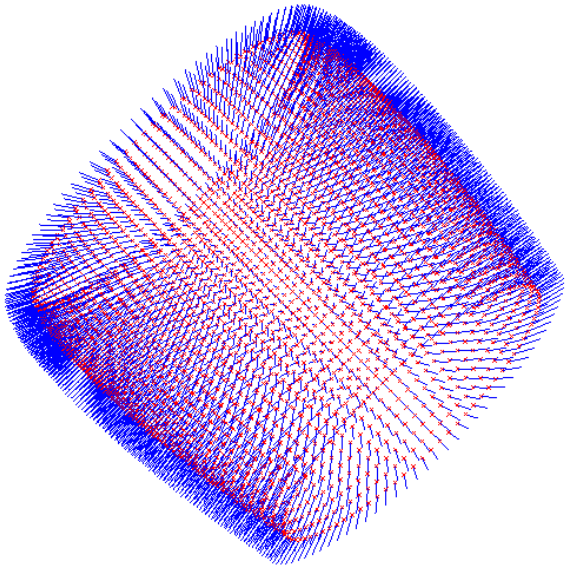
normal projection, that is, project the point p on the domain surface $\mathcal{S}$ along the normal to $\mathcal{S}$ to a distance proportional to the value of the field $\mathcal{F}$ in p:

$$G(p) = p + L\frac{\nabla S(p)(F(p) - F_{min})}{||\nabla S(p)||(F_{max} - F_{min})}$$

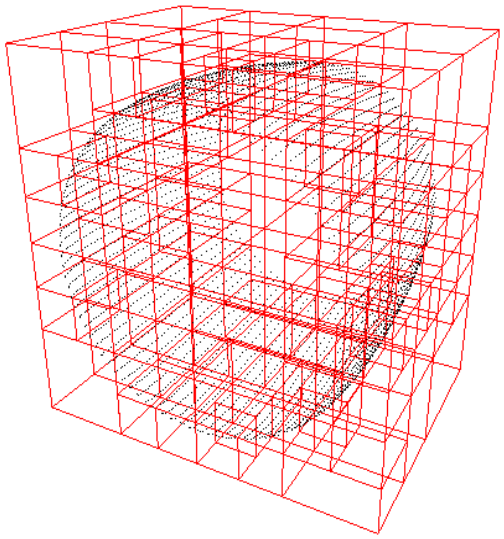
where L is a positive scaler, F_{min} and F_{max} are minimum and maximum values of F on $\mathcal{S}$. However again, if L is not given properly, the surface G may self-intersect when the domain surface $\mathcal{S}$ is not convex.

To avoid self-intersections, as in [7], we take $L < \gamma = \min_{p\in\mathcal{S}}\{R_{min}(p)\}$, where $R_{min}(p)$ is the minimal principal curvature radius at p of surface $\mathcal{S}$. Since $\mathcal{S}$ is a piecewise C^1 continuous surface and each piece is C^∞, hence $\gamma > 0$. The main task here is to compute γ. In general, it is not easy to compute the exact value of γ, however an approximate γ serves our purpose as well. When we produce the triangulation of each surface patch on $\mathcal{S}$, we also compute $R_{min}(p)$ for vertices p, and then take $\gamma = \min_{p\in\mathcal{S}}\{R_{min}(p) : p \text{ is a vertex}\}$. For an implicit surface $f(x_1, x_2, x_3) = 0$, let $f_i = \frac{\partial f}{\partial x_i}$, $f_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$. The principal curvature radius are the absolute values of the roots of the equation $ax^2 + bx + c = 0$, with:

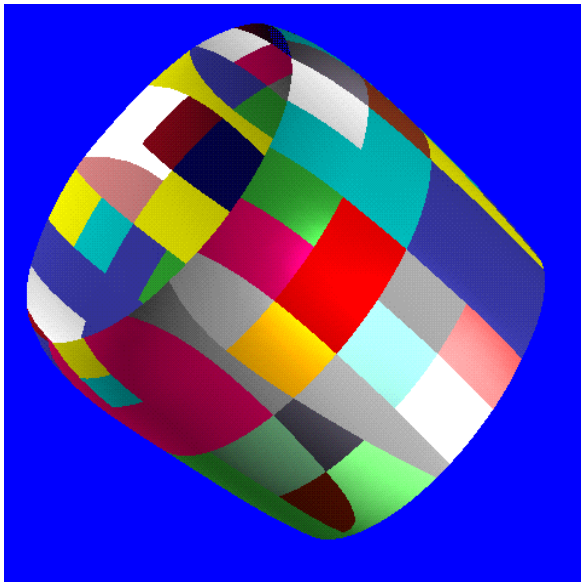
$$\begin{aligned}
a &= \sum_{i=1}^{3} f_i^2(f_{i+1,i+1}f_{i+2,i+2} - f_{i+1,i+2}^2) \\
&+ 2\sum_{i\neq j, k\neq i, k\neq j} f_i f_j (f_{ik}f_{jk} - f_{ij}f_{kk}) \\
b &= ||\nabla f||(\sum_{i=1}^{3} f_i^2(f_{i+1,i+1} + f_{i+2,i+2}) \\
&- 2\sum_{i\neq j} f_i f_j f_{ij}) \\
c &= ||\nabla f||^4
\end{aligned}$$



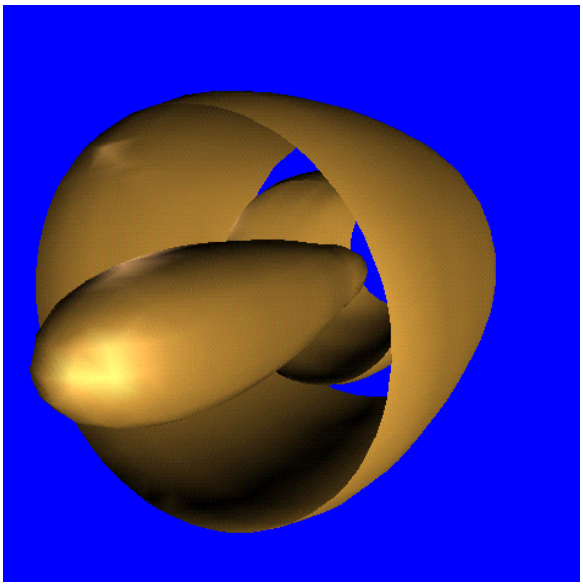
(a)



(b)

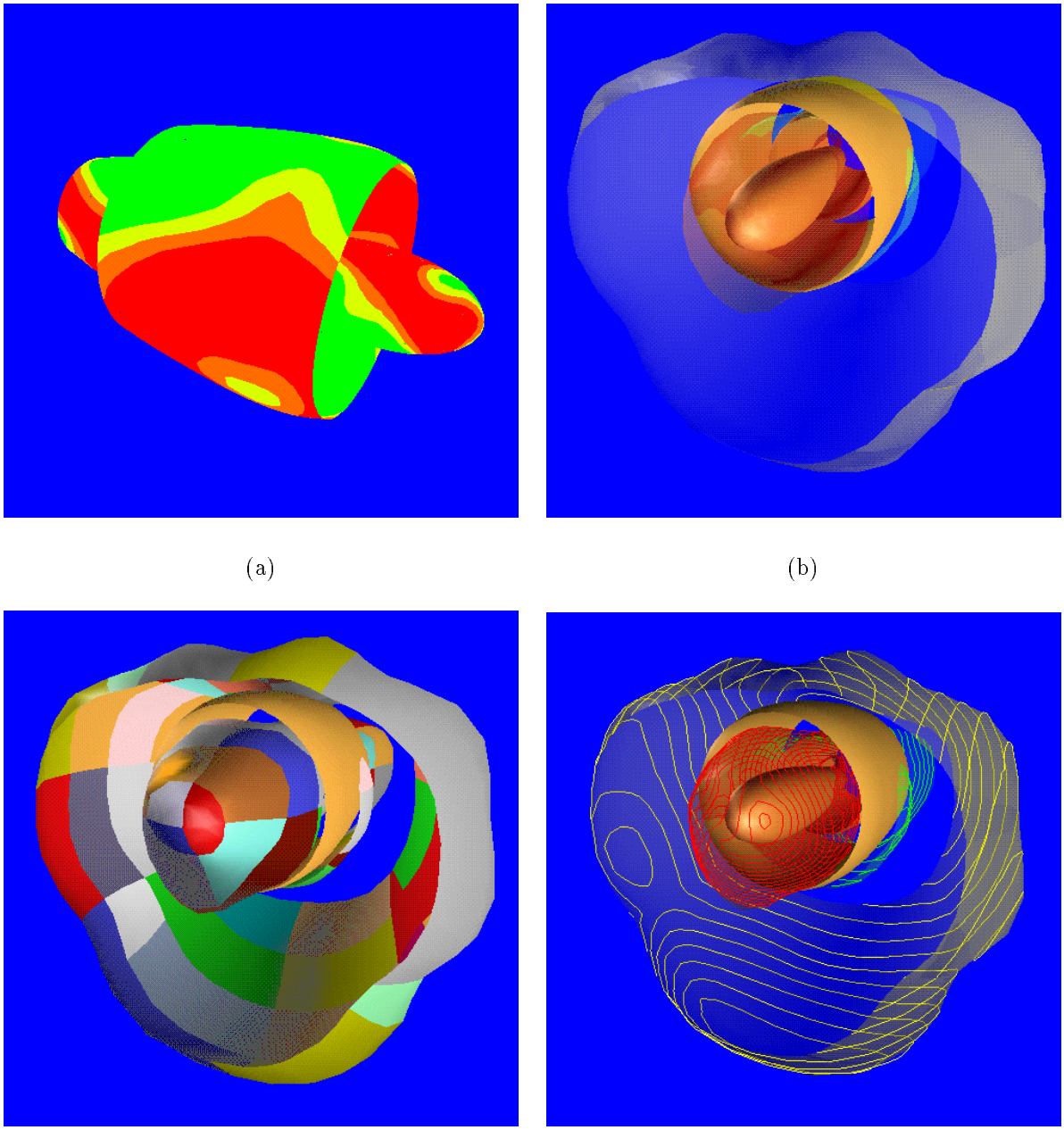


(c)



(d)

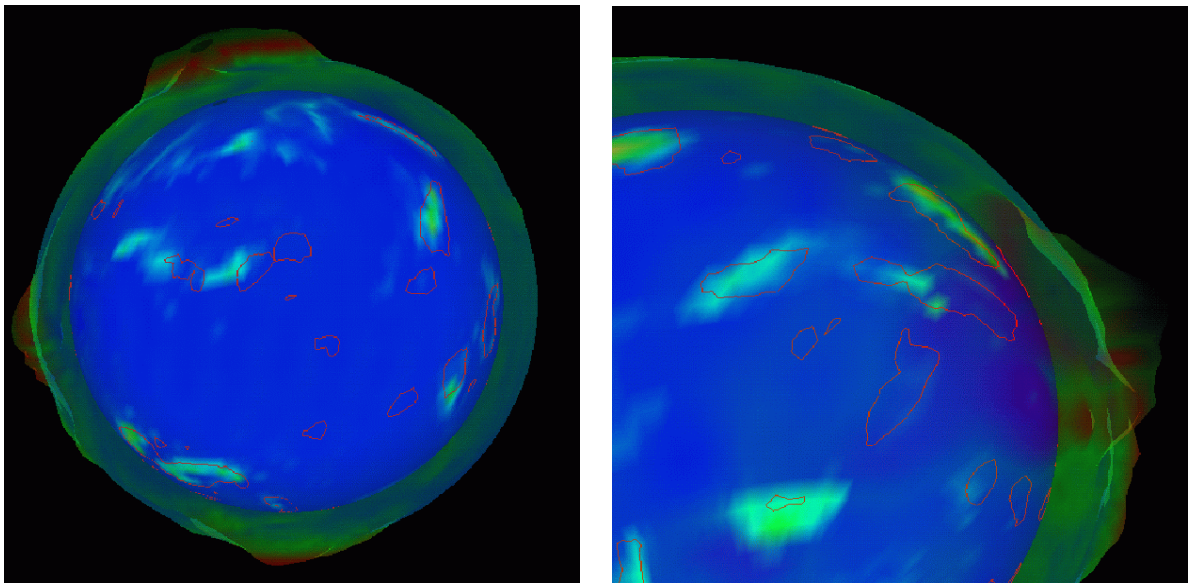
FIGURE 5: Reconstruction of a jet engine from manifold data: (a) Input points for the outer cowl, with oriented normals. (b) Octree subdivision generated by the approximation algorithm. (c) Polynomial spline approximation. (d) Reconstructed engine.



(a) (b)

(c) (d)

FIGURE 6: Visualization of the reconstructed jet engine and a pressure field defined on its surface: (a) Isoregions of the pressure field show on the jet engine surface. (b) The pressure field displayed with the local normal projection method. (c) The polynomial spline patches of the pressure field. (d) Isocontours of the pressure field displayed on the normally projected surface.



(a) (b)

FIGURE 7: Visualization of multiple fields from scattered manifold data. The data represent four different functions (elevation, pressure and two measures of precipitation) over the surface of the Earth. Isocontouring, pseudo-coloring and a function-on-surface obtained with the normal projection method are used.

where the indices are taken mod 3. Another approach to determine L is to take

$$L < \delta = \min_{p \in T_d} \left\{ \min_{q} \left\{ \frac{\|q-p\|^2 \|\nabla S(p)\|}{2(q-p)^T \nabla S(p)} : [p,q] \in T_d \text{ and } (q-p)^T \nabla S(p) > 0 \right\} \right\}$$

where T_d is the triangulation of $\mathcal{S}$ used to display $\mathcal{S}$;, $p \in T_d$ means p is a vertex and $[p,q] \in T_d$ means $[p,q]$ is an edge. It should be noted that if $\mathcal{S}$ is convex(i.e., $(q-p)^T \nabla S(p) \leq 0$) or planar(i.e., $(q-p)^T \nabla S(p) = 0$), the δ can be chosen arbitrarily.

Figures 5 and 6 show an example of a reconstructed object and an associated scalar field from manifold data. The sampled points come from the surface of a jet engine, while the associated values measure the pressure on the engine (data from a simulation). Figure 5 shows several steps of the reconstruction process on one part of the engine. We used the normal propagation method in this example to define an approximate signed distance function. The 3780 data points for the outer cowl are preprocessed to associate local fitting planes and orient the associated normals (Figure 5(a)). The approximation algorithm begins with a given grid (in this case, a uniform subdivision into $5 \times 5 \times 5$ equally-sized boxes) and then adaptively refines it until the error bound conditions are met (the error in this example was set to 0.01 times the max size of the object). The final subdivision is displayed in Figure 5(b). A C^1-smooth spline surface is obtained, as shown in Figure 5(c). The full reconstructed engine is finally shown in Figure 5(d). The complete reconstruction of this object took about 30 seconds on a SGI MIPS4400 workstation. Figure 6 shows different visualizations of the reconstructed spline model for the air-pressure field over the surface of the jet engine. In Figure 6(a) some isoregions of the pressure field have been drawn on the engine surface. In Figure 6(b) we have used the normal-projection method to show the scalar field as a function-on-surface. Points on the domain surface have been projected along the surface normal direction to a distance proportional to the value of the field at that point. The field surface patches are visible in Figure 6(c). Finally, in Figure 6(d), we show isocontours projected on the function-on-surface.

Using the normal projection method one can simultaneously visualize multiple co-registered fields. This can be extremely helpful in understanding correlation in complex data sets. For example, one might visualize a manifold domain, and draw multiple function-on-surfaces, one for each field, using different offsets. Alternatively, one can draw one field as a function-on-surface, and use the value of other fields to pseudo-color (or draw isocurves on) the domain or the function-on-surface or both. Figure 7 shows a visualization of multiple scalar fields over the surface of the Earth, combining several of the techniques described above.

6 Conclusions

Techniques are presented to incrementally and adaptively build polynomial spline models, based on an octree subdivision, from a set of multiple data values scattered in space. The modeling of manifold data is reduced to the problem to volumetric field reconstruction, *via* a signed-distance function. The geometry of the manifold and the associated physical properties are modeled by trivariate polynomial splines defined over a common domain, a feature that solves the data co-registration problem and simplifies visualization.

Availability of a spline model coupled with search structures supports *interrogative visualization*, i.e. quantitatively querying the data for information like value, differential properties and topological characteristics. We have implemented sll of the algorithms in our Shastra environment and used it to produce the pictures shown in this paper.

Acknowledgments

This survey paper provides a snapshot of ongoing research with Fausto Bernardini and Guoliang Xu. I am thankful to them as well as to Daniel Schikore for his help with several of the visualization pictures. Thanks also to the following persons and organizations for providing some of the data sets used in the paper: J. Simons, Dept. of Chemistry, University of Utah (Figure 1), Brookhaven Protein Data Bank (Figure 4), NASA Research Center (Figures 5 and 6), R. Oglesby, Dept. of Earth & Atmospheric Sciences, Purdue University (Figure 7).

References

[1] Alfeld, P. Scattered data interpolation in three or more variables. In *Mathematical Methods in Computer Aided Geometric Design*, T. Lyche and L. Schumaker, Eds. Academic Press, Boston, 1989, pp. 1–34.

[2] Bajaj, C., Bernardini, F., and Xu, G. Adaptive reconstruction of surfaces and scalar fields from dense scattered trivariate data. Tech. Rep. CSD-TR-95-028, Department of Computer Sciences, Purdue University, Apr. 1995.

[3] Bajaj, C., Bernardini, F., and Xu, G. Automatic reconstruction of surfaces and scalar fields from 3D scans. In *Computer Graphics Proceedings* (1995), Annual Conference Series. Proceedings of SIGGRAPH 95, ACM SIGGRAPH, pp. 109–118.

[4] Bajaj, C., Chen, J., and Xu, G. Modeling with cubic A-patches. *ACM Transactions on Graphics 14*, 2 (1995), 103–133.

[5] Bajaj, C., and Evans, S. Splines and geometric modeling. In *CRC Handbook of Discrete and Computational Geometry*, J. E. Goodman and J. O'Rourke, Eds. CRC Press, 1997. 52 chapters, approx. xiv + 987 pages; projected publication: July 1997.

[6] Bajaj, C., and Lin, K. N. Scalar field modeling and visualization on the Intel Delta. Tech. Rep. 94-039, Purdue University. Dept. of Computer Sciences, June 1994.

[7] Bajaj, C., and Xu, G. Modeling scattered function data on curved surfaces. In *Fundamentals of Computer Graphics*, Z. T. J. Chen, N. Thalmann and D. Thalmann, Eds. World Scientific Publishing Co., Beijing, China, 1994, pp. 19–29.

[8] Bajaj, C. L., Pascucci, V., and Schikore, D. R. Fast isocontouring for improved interactivity. In *Proceedings of 1996 Symposium on Volume Visualization* (Oct. 1996), pp. 39–46.

[9] Barnhill, R. E. Surfaces in computer aided geometric design: A survey with new results. *Computer Aided Geometric Design 2* (1985), 1–17.

[10] Barnhill, R. E., Opitz, K., and Pottmann, H. Fat surfaces: a trivariate approach to triangle-based interpolation on surfaces. *Computer Aided Geometric Design 9* (1992), 365–378.

[11] Bolle, R. M., and Vemuri, B. C. On three-dimensional surface reconstruction methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 1 (Jan. 1991), 1–13.

[12] Chen, X., and Schmitt, F. Surface modelling of range data by constrained triangulation. *Computer Aided Design 26*, 8 (Aug. 1994), 632–645.

[13] Choi, B. K., Shin, H. Y., Yoon, Y. I., and Lee, J. W. Triangulation of scattered data in 3D space. *Computer Aided Design 20*, 5 (June 1988), 239–248.

[14] Dahmen, W. Smooth piecewise quadratic surfaces. In *Mathematical Methods in Computer Aided Geometric Design*, T. Lyche and L. Schumaker, Eds. Academic Press, Boston, 1989, pp. 181–193.

[15] Dahmen, W., and Thamm-Schaar, T.-M. Cubicoids: modeling and visualization. *Computer Aided Geometric Design 10* (1993), 93–108.

[16] deBoor, C., Hollig, K., and Riemenschneider, S. *Box Splines.* Springer Verlag, New York, 1993.

[17] Edelsbrunner, H., and Mücke, E. P. Three-dimensional alpha shapes. *ACM Trans. Graph. 13*, 1 (Jan. 1994), 43–72.

[18] Faugeras, O. D., Hebert, M., Mussi, P., and Boissonnat, J. D. Polyhedral approximation of 3-D objects without holes. *Computer Vision, Graphics and Image Processing 25* (1984), 169–183.

[19] Foley, T. A. Interpolation to scattered data on a spherical domain. In *Algorithms fo Approximation II*, M. Cox and J. Mason, Eds. Chapman and Hall, London, 1990, pp. 303–310.

[20] Forsey, D. R., and Bartels, R. H. Hierarchical B-spline refinement. In *Computer Graphics (SIGGRAPH '88 Proceedings)* (Aug. 1988), J. Dill, Ed., vol. 22, pp. 205–212.

[21] Franke, R. Recent advances in the approximation of surfaces from scattered data. In *Multivariate Approximation*, C.K.Chui, L.L.Schumarker, and F.I.Utreras, Eds. Academic Press, New York, 1987, pp. 275–335.

[22] Goshtasby, A. Design and recovery of 2-D and 3-D shapes using rational Gaussian curves and surfaces. *International Journal of Computer Vision 10*, 3 (1993), 233–256.

[23] Guo, B. *Modeling arbitrary smooth objects with algebraic surfaces.* PhD thesis, Computer Science, Cornell University, 1991.

[24] Hoppe, H., DeRose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schwitzer, J., and Stuelzle, W. Piecewise smooth surface reconstruction. In *Computer Graphics Proceedings* (1994), Annual Conference Series. Proceedings of SIGGRAPH 94, ACM SIGGRAPH, pp. 295–302.

[25] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuelzle, W. Surface reconstruction from unorganized points. *Computer Graphics 26*, 2 (July 1992), 71–78. Proceedings of SIGGRAPH 92.

[26] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. Mesh optimization. In *Proc. SIGGRAPH '93* (1993), pp. 19–26.

[27] Liao, C. W., and Medioni, G. Surface approximation of a cloud of 3D points. *Graphical Models & Image Processing 57*, 1 (Jan. 1995), 67–74.

[28] Moore, D., and Warren, J. Approximation of dense scattered data using algebraic surfaces. In *Proceedings of the 24th annual Hawaii International Conference on System Sciences* (1991), V. Milutinovic and B. D. Shriver, Eds., vol. 1.

[29] Nielson, G. M. Modeling and visualizing volumetric and surface-on-surface data. In *Focus on Scientific Visualization*, H. Hagen, H. Muller, and G. M. Nielson, Eds. Springer, 1992, pp. 219–274.

[30] Nielson, G. M. Scattered data modeling. *IEEE Computer Graphics & Applications 13* (1993), 60–70.

[31] Nielson, G. M., Foley, T. A., Hamann, B., and Lane, D. Visualizing and modeling scattered multivariate data. *IEEE Computer Graphics & Applications 11*, 3 (May 1991), 47–55.

[32] Patrikalakis, N. M., and Kriezis, G. A. Representation of piecewise continuous algebraic surfaces in terms of B-splines. *The Visual Computer 5* (1989), 360–374.

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

[34] Schmitt, F., Barsky, B. A., and Du, W. An adaptive subdivision method for surface fitting from sampled data. *Computer Graphics 20*, 4 (1986), 179–188. Proceedings of SIGGRAPH 86.

[35] Sederberg, T. W. Piecewise algebraic surface patches. *Computer Aided Geometric Design 2* (1985), 53–59.

[36] Veltkamp, R. C. *Closed object boundaries from scattered points.* PhD thesis, Center for Mathematics and Computer Science, Amsterdam, 1992.

[37] Wilhelms, J., and Van Gelder, A. Topological considerations in isosurface generation: Extended abstract. *Computer Graphics (San Diego Workshop on Volume Visualization) 24*, 5 (Nov. 1990), 79–86.