Surface Fitting with Hierarchical Splines

David R. Forsey
Richard H. Bartels
Computer Graphics Laboratory
Computer Science Department
University of Waterloo
Waterloo, Ontario
Canada N2L 3G1

January 1, 1995

Abstract

We consider the fitting of tensor product parametric spline surfaces to gridded data. The continuity of the surface is provided by the basis chosen. When tensor product splines are used with gridded data, the surface fitting problem decomposes into a sequence of curve fitting processes, making the computations particularly efficient. The use of a hierarchical representation for the surface adds further efficiency by adaptively decomposing the fitting process into subproblems involving only a portion of the data. Hierarchy also provides a means of storing the resulting surface in a compressed format. Our approach is compared to multiresolution analysis and the use of wavelets.

1 Introduction

In [9] an adaptive process was presented for fitting surface data with a geometrically continuous collection of rectangular Bézier patches. The adaptivity resulted from fitting a portion of the data with a patch, testing the fit for satisfaction within a given tolerance, and subdividing the patch if the tolerance was not met. Geometric continuity was provided by using constrained least squares as the fitting process, with the constraints imposing the continuity conditions of tensor product β-splines.

The data used in [9] was organized in a rectangular array; that is, it was gridded. This is the output format typical of systematic measuring devices such as laser rangers, CAT imagery systems, and optical scanners. Advantage can be taken of this format by reducing the surface fitting problem into a sequence of much smaller curve fitting problems [1, 3].

Bézier patches require that a significant number of constraints be imposed on the control vertices to piece patches together in a continuous composite surface. The broader class of multi-patch tensor product spline surfaces; e.g. B-splines, β-splines, or their rational counterparts, provide continuity without the imposition of constraints in the least squares fitting process.

The hierarchical representation of [4] allows an adaptive approach to the fitting process. When areas of large scale data have been fit within a specified tolerance by a surface having a certain level of refinement, there may remain isolated areas of data that exceed tolerance. Smaller least squares problems involving more refined overlay surfaces can be generated to fit the data still out of tolerance. The overlay fitting problems must be constrained, but the constraints are of such a simple nature that they only serve to simplify the least squares process by decreasing the size of the fitting problem. The hierarchical representation makes a further contribution by enabling a certain economy of representation for the final composite surface.

2 Data Fitting

We are interested in fitting tensor product spline surfaces

$$\sum_{i=0}^{m} \sum_{j=0}^{n} \mathrm{V}_{i,j} B_i(u) C_j(v) \tag{1}$$

to given data points. The basis functions B_i and C_j will be left open throughout the discussion but are required to permit refinement by knot insertion. This is a property common to B-splines, β-splines, and rational splines derived from either of these kinds of bases.

The generation of equations for the data fitting problems proceeds as follows. For each data point P_λ we must associate a domain point $(u, v) = \delta_\lambda$. This is not a trivial problem, but a number of practical techniques exist [11].

After an association has been made, we form the equations

$$\left. \sum_{i=0}^{m} \sum_{j=0}^{n} \mathrm{V}_{i,j} B_i(u) C_j(v) \right|_{(u,v)=\delta_\lambda} = \mathrm{P}_\lambda \tag{2}$$

Data is gridded if

$$u \in \{u_0, \cdots, u_M\}$$
$$v \in \{v_0, \cdots, v_N\}$$

and if the δ's consist of all points in $\{u_0, \cdots, u_M\} \times \{v_0, \cdots, v_N\}$. In the gridded case the data fitting equations become

$$\sum_{i=0}^{m} \sum_{j=0}^{n} \mathrm{V}_{i,j} B_i(u_r) C_j(v_s) = \mathrm{P}_{r,s} \tag{3}$$

for $r = 0, \cdots, M$ and $s = 0, \cdots, N$.

This gridded set of equations can be looked at in either of two ways, from a compact (tensor product) point of view and from an exploded (Kronecker product) point of view. The tensor product view is

$$\begin{aligned} &\begin{bmatrix} B_0(u_0) & B_1(u_0) & \cdots & B_m(u_0) \\ B_0(u_1) & B_1(u_1) & \cdots & B_m(u_1) \\ \vdots & \vdots & \vdots & \vdots \\ B_0(u_M) & B_1(u_M) & \cdots & B_m(u_M) \end{bmatrix} \times \begin{bmatrix} \mathrm{V}_{0,0} & \mathrm{V}_{0,1} & \cdots & \mathrm{V}_{0,n} \\ \mathrm{V}_{1,0} & \mathrm{V}_{1,1} & \cdots & \mathrm{V}_{1,n} \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{V}_{m,0} & \mathrm{V}_{m,1} & \cdots & \mathrm{V}_{m,n} \end{bmatrix} \\ &\times \begin{bmatrix} C_0(v_0) & C_0(v_1) & \cdots & C_0(v_N) \\ C_1(v_0) & C_1(v_1) & \cdots & C_1(v_N) \\ \vdots & \vdots & \vdots & \vdots \\ C_n(v_0) & C(v_1) & \cdots & C(v_N) \end{bmatrix} = \begin{bmatrix} \mathrm{P}_{0,0} & \mathrm{P}_{0,1} & \cdots & \mathrm{P}_{0,N} \\ \mathrm{P}_{1,0} & \mathrm{P}_{1,1} & \cdots & \mathrm{P}_{1,N} \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{P}_{M,0} & \mathrm{P}_{M,1} & \cdots & \mathrm{P}_{M,N} \end{bmatrix} \end{aligned} \tag{4}$$

or, briefly,

$$\mathbf{BVC}^T = \mathbf{P} \tag{5}$$

Matrices $\mathbf{B}$ and $\mathbf{C}$ are those associated with the curve fitting problems for parametric curves in u (and respectively v) using the columns (respectively the rows) of the matrix of control vertices $\mathbf{V}$ as unknowns and using the columns (respectively the rows) of the matrix $\mathbf{P}$ as right hand sides.

The exploded point of view derives from taking the columns of the matrices $\mathbf{V}$ and $\mathbf{P}$ and stacking them above each other to produce the vectors

$$\mathrm{vec}(\mathbf{V}) = \begin{bmatrix} \mathrm{V}_{0,0} \\ \vdots \\ \mathrm{V}_{m,0} \\ \mathrm{V}_{0,1} \\ \vdots \\ \mathrm{V}_{m,1} \\ \vdots \\ \mathrm{V}_{m,n} \end{bmatrix}, \mathrm{vec}(\mathbf{P}) = \begin{bmatrix} \mathrm{P}_{0,0} \\ \vdots \\ \mathrm{P}_{M,0} \\ \mathrm{P}_{0,1} \\ \vdots \\ \mathrm{P}_{M,1} \\ \vdots \\ \mathrm{P}_{M,N} \end{bmatrix} \tag{6}$$

In terms of these vectors, the curve fitting problem becomes

$$\mathbf{A}\,\mathrm{vec}(\mathbf{V}) = \mathrm{vec}(\mathbf{P}) \tag{7}$$

where the matrix $\mathbf{A}$ is made of m by M blocks, arranged in N rows and n columns, each block being a copy of $\mathbf{B}$ multiplied by an element of $\mathbf{C}$:

$$\begin{bmatrix} C_0(v_0)\mathbf{B} & \mathrm{C}_1(v_0)\mathbf{B} & \cdots \\ C_0(v_1)\mathbf{B} & \ddots & \vdots \\ \vdots & \cdots & C_n(v_N)\mathbf{B} \end{bmatrix}$$

3 Kronecker Products

A useful notation [6] is that of the Kronecker product. If

$$\mathbf{F} = \begin{bmatrix} f_{0,0} & f_{0,1} & \cdots & f_{0,\beta} \\ f_{1,0} & f_{1,1} & \cdots & f_{1,\beta} \\ \vdots & \vdots & \cdots & \vdots \\ f_{\alpha,0} & f_{\alpha,1} & \cdots & f_{\alpha,\beta} \end{bmatrix} \tag{8}$$

is any matrix and $\mathbf{G}$ is any other matrix, then

$$\mathbf{F} \otimes \mathbf{G} = \begin{bmatrix} f_{0,0}\mathbf{G} & f_{0,1}\mathbf{G} & \cdots \\ f_{1,0}\mathbf{G} & f_{1,1}\mathbf{G} & \vdots \\ \vdots & \cdots & f_{\alpha,\beta}\mathbf{G} \end{bmatrix} \tag{9}$$

In this notation the exploded view of the curve fitting equations is

$$(\mathbf{C} \otimes \mathbf{B})\,\mathrm{vec}(\mathbf{V}) = \mathrm{vec}(\mathbf{P}) \tag{10}$$

The exploded view and the compact view represented by (5) can be connected by an extension of the "vec" operator:

$$\mathrm{vec}\left(\mathbf{BVC}^T\right) = (\mathbf{C} \otimes \mathbf{B})\,\mathrm{vec}(\mathbf{V}) \tag{11}$$

More generally, if matrices $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{H}$ have compatible dimensions so that the product $\mathbf{FGH}$ is defined, then

$$\mathrm{vec}(\mathbf{FGH}) = \left(\mathbf{H}^T \otimes \mathbf{F}\right)\mathrm{vec}(\mathbf{G}) \tag{12}$$

We obtain (7) with $\mathbf{A}$ given by (10) by applying the vec operator to both sides of (5).

Useful properties in addition to (12) are given in [6], and we list those of interest to us below. The inverses are assumed to exist, a stands for any real number, and dimensions are assumed compatible wherever needed to make the matrix sums and products exist.

$$\begin{aligned}
\mathbf{F} \otimes (a\mathbf{G}) &= (a\mathbf{F}) \otimes \mathbf{G} = a(\mathbf{F} \otimes \mathbf{G}) && (13)\\
(\mathbf{F}_0 + \mathbf{F}_1) \otimes \mathbf{G} &= (\mathbf{F}_0 \otimes \mathbf{G}) + (\mathbf{F}_1 \otimes \mathbf{G}) && (14)\\
\mathbf{F} \otimes (\mathbf{G}_0 + \mathbf{G}_1) &= (\mathbf{F} \otimes \mathbf{G}_0) + (\mathbf{F} \otimes \mathbf{G}_1) && (15)\\
(\mathbf{F} \otimes \mathbf{G}) \otimes \mathbf{H} &= \mathbf{F} \otimes (\mathbf{G} \otimes \mathbf{H}) && (16)\\
(\mathbf{F} \otimes \mathbf{G})^T &= \mathbf{F}^T \otimes \mathbf{G}^T && (17)\\
(\mathbf{F}_0 \otimes \mathbf{G}_0)(\mathbf{F}_1 \otimes \mathbf{G}_1) &= (\mathbf{F}_0\mathbf{F}_1) \otimes (\mathbf{G}_0\mathbf{G}_1) && (18)\\
(\mathbf{F} \otimes \mathbf{G})^{-1} &= \mathbf{F}^{-1} \otimes \mathbf{G}^{-1} && (19)
\end{aligned}$$

4 Interpolation and Least Squares

We can use the properties of the Kronecker product to interpolate tensor product arrangements efficiently. If (3) is a square, nonsingular system, then

$$\begin{aligned}
\mathrm{vec}(\mathbf{V}) &= (\mathbf{C} \otimes \mathbf{B})^{-1}\mathrm{vec}(\mathbf{P})\\
&= \left(\mathbf{C}^{-1} \otimes \mathbf{B}^{-1}\right)\mathrm{vec}(\mathbf{P}) \qquad (20)\\
&= \mathrm{vec}\left(\mathbf{B}^{-1}\mathbf{P}\mathbf{C}^{-T}\right)
\end{aligned}$$

implying that

$$\mathbf{V} = \mathbf{B}^{-1}\mathbf{P}\mathbf{C}^{-T} \qquad (21)$$

Equation (21) could, of course, be obtained directly from (5). The Kronecker product will be more useful below when we establish approaches to least squares and constrained least squares problems.

As has already been remarked, (21) constitutes a sequence of curve interpolation problems

$$\mathrm{co}\ell_j(\mathbf{V}) = \mathbf{B}^{-1}\mathrm{co}\ell_j(\mathbf{H}) \quad j = 0, \cdots, n \qquad (22)$$

where the data, H, is found from

$$\mathrm{row}_i(\mathbf{H}) = \mathrm{row}_i(\mathbf{P})\mathbf{C}^{-T} \quad i = 0, \cdots, M \qquad (23)$$

which constitutes a prior sequence of curve interpolation problems. The efficiency of this approach derives from the small size of the curve interpolation problems relative to the surface interpolation problem, and from the fact that only two matrix factorizations are required, one each of $\mathbf{B}$ and $\mathbf{C}$. From (19), $\mathbf{C} \otimes \mathbf{B}$ is nonsingular if and only if both $\mathbf{B}$ and $\mathbf{C}$ are nonsingular. In turn, each of these matrices separately will be nonsingular if and only if the data parameters are sufficiently evenly spaced among the knots; i.e., $t_r \leq u_r \leq t_{r+k}$, where the t's are the knots for the parameter u, and k is the order of the basis splines B in that parameter (and similarly for C) [1].

If (3) is an overdetermined system; that is, if the amount of data exceeds the number of control vertices, and a spline surface is sought that will approximate the data, then (7) with $\mathbf{A}$ given by (10) would be replaced by the normal equations for the least squares solution [3]

$$(\mathbf{C} \otimes \mathbf{B})^T (\mathbf{C} \otimes \mathbf{B}) \operatorname{vec}(\mathbf{V}) = (\mathbf{C} \otimes \mathbf{B})^T \operatorname{vec}(\mathbf{P}) \tag{24}$$

which can be rewritten by (17) and (18) as

$$\left[\left(\mathbf{C}^T\mathbf{C}\right) \otimes \left(\mathbf{B}^T\mathbf{B}\right)\right] \operatorname{vec}(\mathbf{V}) = \left(\mathbf{C}^T \otimes \mathbf{B}^T\right) \operatorname{vec}(\mathbf{P}) \tag{25}$$

or, using the vec operator,

$$\operatorname{vec}\left[\left(\mathbf{B}^T\mathbf{B}\right) \mathbf{V} \left(\mathbf{C}^T\mathbf{C}\right)\right] = \operatorname{vec}\left(\mathbf{B}^T\mathbf{P}\mathbf{C}\right) \tag{26}$$

which implies

$$\mathbf{V} = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{P}\mathbf{C}\left(\mathbf{C}^T\mathbf{C}\right)^{-1} \tag{27}$$

This is the least squares system corresponding to (21) for the interpolation problem, and it leads to successive, least squares, curve problems corresponding to (22),

$$\operatorname{co\ell}_j(\mathbf{V}) = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\operatorname{co\ell}_j(\mathbf{H}) \quad j = 0, \cdots, n \tag{28}$$

and to (23)

$$\operatorname{row}_i(\mathbf{H}) = \operatorname{row}_i(\mathbf{P})\mathbf{C}\left(\mathbf{C}^T\mathbf{C}\right)^{-1} \quad i = 0, \cdots, M \tag{29}$$

5 Constraints

The efficiencies carry over to least squares with linear constraints, provided that the constraints retain a Kronecker structure. Constraints with this restriction are still interesting, as we shall see. The constrained setting is

$$\begin{array}{rcc} (\mathbf{C} \otimes \mathbf{B}) \operatorname{vec}(\mathbf{V}) & \approx & \mathbf{P} \\ \text{subject to } (\mathbf{L} \otimes \mathbf{K}) \operatorname{vec}(\mathbf{V}) & = & \operatorname{vec}(\mathbf{D}) \end{array} \tag{30}$$

The symbol $\approx$ denotes the equations from the data fitting part of the problem and suggests that these equations can be satisfied, at best, only in an approximate sense. The constraint equations are expected to be fewer in number than the control vertices, otherwise they cannot usually be satisfied, or if satisfiable, they will so constrain the control vertices that no degrees of freedom remain to use in the least squares part of (30). The numerical analyst's approach [5] to solving (30) is a projective one. Letting $\mathbf{G}$ stand for the matrix $\mathbf{L} \otimes \mathbf{K}$, and v and d stand for the two respective vectors, the solution v to the constraint equations

$$\mathbf{G}\mathrm{v} = \mathrm{d} \tag{31}$$

can be written in terms of two components, $\mathrm{v}_{\mathcal{W}}$ and $\mathrm{v}_{\mathcal{Z}}$, which are the components of v projected respectively onto the range space of $\mathbf{G}$ and onto its orthogonal complement space. The precise expression of this fact is through the equations

$$\mathrm{v}_{\mathcal{W}} = \mathbf{G}^T\mathrm{f},\ \mathbf{G}\mathrm{v}_{\mathcal{Z}} = 0,\ \text{and } \mathrm{v} = \mathrm{v}_{\mathcal{W}} + \mathrm{v}_{\mathcal{Z}} \tag{32}$$

for some vector f. From this we have

$$\mathbf{G}\mathrm{v} = \mathbf{G}\mathrm{v}_{\mathcal{W}} + \mathbf{G}\mathrm{v}_{\mathcal{Z}} = \mathbf{G}\mathbf{G}^T\mathrm{f} = \mathrm{d} \tag{33}$$

Making the reasonable assumption that the constraints are linearly independent so that $\mathbf{G}\mathbf{G}^T$ is nonsingular,

$$\mathrm{f} = \left(\mathbf{G}\mathbf{G}^{\mathbf{T}}\right)^{-1}\mathrm{d} \tag{34}$$

This equation system can be solved in a fashion similar to the one used for solving the least squares part of the problem. Returning to Kronecker product notation and using $\mathrm{vec}(\mathbf{F})$ as f, we have

$$(\mathbf{L} \otimes \mathbf{K})\,\mathrm{vec}(\mathbf{V}) = (\mathbf{L} \otimes \mathbf{K})\,(\mathbf{L} \otimes \mathbf{K})^T\mathrm{vec}(\mathbf{F}) = \mathrm{vec}(\mathbf{D}) \tag{35}$$

or

$$\left[\left(\mathbf{L}\mathbf{L}^T\right) \otimes \left(\mathbf{K}\mathbf{K}^T\right)\right]\mathrm{vec}(\mathbf{F}) = \mathrm{vec}(\mathbf{D}) \tag{36}$$

which implies that

$$\mathbf{F} = \left(\mathbf{K}\mathbf{K}^T\right)^{-1}\mathbf{D}\left(\mathbf{L}\mathbf{L}^T\right)^{-1} \tag{37}$$

and

$$\mathbf{V}_{\mathcal{W}} = \mathbf{K}^T\mathbf{F}\mathbf{L} \tag{38}$$

in tensor product format. Since $\mathbf{V}_{\mathcal{W}}$ is known, the least squares part of the equation system (30) can now be written

$$(\mathbf{C} \otimes \mathbf{B})\, \mathrm{V}_{\mathcal{Z}} \approx \mathrm{P} - (\mathbf{C} \otimes \mathbf{B})\, \mathrm{V}_{\mathcal{W}} \tag{39}$$

that is

$$\mathbf{B}\mathbf{V}_{\mathcal{Z}}\mathbf{C}^T \approx \mathbf{P} - \mathbf{B}\mathbf{V}_{\mathcal{W}}\mathbf{C}^T \tag{40}$$

where

$$\mathrm{V}_{\mathcal{W}} = vec(\mathbf{V}_{\mathcal{W}}),\ \mathrm{V}_{\mathcal{Z}} = vec(\mathbf{V}_{\mathcal{Z}}),\ \text{and } \mathrm{P} = \mathrm{vec}(\mathbf{P}) \tag{41}$$

This is a problem of exactly the format we considered in Section 4, merely with a right hand side reduced by subtracting off the degrees of freedom fixed by the constraints. As a computational note, the inverses involving **B**, **C**, **K**, and **L** are not to be computed explicitly. For (21) a triangular matrix decomposition is generally used, and for (27) and (37) an orthogonal matrix decomposition is used [5, 7].

For the purposes of hierarchical fitting, the constraints of interest to us are those that fix the values the surface around its perimeter. Equivalently, these are constraints that fix the values of $\mathrm{V}_{i,j}$ for $0 \leq i \leq \ell eft$, $right \leq i \leq m$, $0 \leq j \leq bottom$, and $top \leq j \leq n$, where ℓeft, $right$, $bottom$, and top depend upon the order and knot structure of the basis functions B_i and C_j. For uniform bicubic B-spline surfaces, for example, ℓeft and $bottom$ will each equal 3, as will $m - right$ and $n - top$. In this case, the problem given by (30) and the solution system (40) have a special structure that are worth exploiting.

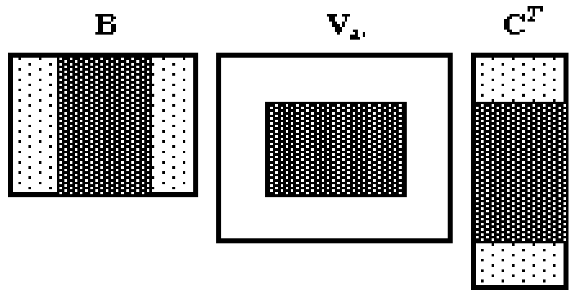
When individual control vertices $\mathrm{V}_{i,j}$ are constrained in value, we may write $\mathbf{V} = \mathbf{V}_\mathrm{f} + \mathbf{V}_\mathrm{v}$, where $\mathbf{V}_\mathrm{f}$ represents the fixed control vertices and $\mathbf{V}_\mathrm{v}$ represents the control vertices that are still free to vary. The equation system (5) becomes,

$$\mathbf{B}\,(\mathbf{V}_\mathrm{f} + \mathbf{V}_\mathrm{v})\,\mathbf{C}^T = \mathbf{P} \tag{42}$$

and the least squares problem to be solved is given by a simple version of (40),

$$\mathbf{B}\mathbf{V}_\mathrm{v}\mathbf{C}^T = \mathbf{P} - \mathbf{B}\mathbf{V}_\mathrm{f}\mathbf{C}^T \tag{43}$$

This problem involves a modified right hand side. Its left hand side is a reduced version of that in (5). To see this, we look into the structure of the matrix product on the left hand side:
The clear areas of $\mathbf{V}_\mathrm{v}$ represent the values of zero corresponding to the components of $\mathbf{V}_\mathrm{f}$. The entries of **B** in the lightly shaded area multiply those zeros, as do the lightly shaded entries of **C**. An equivalent matrix



product is formed by trimming away the zero rows and columns of $\mathbf{V}_{\mathrm{v}}$, as well as the lightly shaded columns of $\mathbf{B}$ and the lightly shaded rows of $\mathbf{C}^T$ (columns of $\mathbf{C}$). After this trimming, the left hand side of (4) becomes

$$\begin{bmatrix} B_{bottom+1}(u_0) & B_{bottom+2}(u_0) & \cdots & B_{top-1}(u_0) \\ B_{bottom+1}(u_1) & B_{bottom+2}(u_1) & \cdots & B_{top-1}(u_1) \\ \vdots & \vdots & \vdots & \vdots \\ B_{bottom+1}(u_M) & B_{bottom+2}(u_M) & \cdots & B_{top-1}(u_M) \end{bmatrix}$$

$$\times \begin{bmatrix} V_{bottom+1,\ell eft+1} & V_{bottom+1,\ell eft+2} & \cdots & V_{bottom+1,right-1} \\ V_{bottom+2,\ell eft+1} & V_{bottom+2,\ell eft+2} & \cdots & V_{bottom+2,right-1} \\ \vdots & \vdots & \vdots & \vdots \\ V_{top-1,\ell eft+1} & V_{top-1,\ell eft+2} & \cdots & V_{top-1,right-1} \end{bmatrix}$$

$$\times \begin{bmatrix} C_{\ell eft+1}(v_0) & C_{\ell left+1}(v_1) & \cdots & C_{\ell eft+1}(v_N) \\ C_{\ell eft+2}(v_0) & C_{\ell eft+2}(v_1) & \cdots & C_{\ell eft+2}(v_N) \\ \vdots & \vdots & \vdots & \vdots \\ C_{right-1}(v_0) & C_{right-1}(v_1) & \cdots & C_{right-1}(v_N) \end{bmatrix}$$

Such trimmed systems allow the imposition of a hierarchy upon the fitting process, with the potential of gaining substantial efficiency, as we shall show in Section 6.

6 Hierarchical Representation

In [4] a compact means of representing spline surfaces derived from successive refinement was described. There were three essential features of this representation. One was that only the modified portions of a hierarchical surface need to be stored in a data structure; the second was that each level of refinement, or overlay, was represented as an offset from reference position derived from a level of lower refinement, and third was that editing operates on points selected directly from the composite surface itself rather than through control vertices. The last feature is not relevant to surface fitting, which is concerned with the static approximation of given data. The other two features, however, lend themselves to a compact and efficient means of fitting surfaces to data.

The approach taken utilizes least squares to fit a template spline surface to the data; e.g. a spline approximation to a plane, sphere, cylinder, or torus. The template is chosen to model the main topology of the data appropriately. The fitting operates on each coordinate separately

$$(\mathbf{C} \otimes \mathbf{B})^T (\mathbf{C} \otimes \mathbf{B}) \mathrm{vec}(\mathbf{V}_\mathrm{c}) = (\mathbf{C} \otimes \mathbf{B})^T \mathbf{P}_\mathrm{c} \tag{44}$$

where c stands for each of x, y, and z, and the purpose served by choosing the proper template is ensuring that the data in each coordinate separately is single valued; that is, that it represents a height field. The residual between the surface and the data

$$\sum_i \sum_j \mathrm{V}_{i,j} B_i(u_r) C_j(v_s) - \mathrm{P}_{r,s} \tag{45}$$

is computed for all r. Out-of-tolerance points are those points (u_r, v_s) of the domain for which the data lies further than a selected tolerance from the surface. If no points are out of tolerance, the data has been fit sufficiently well by the surface. The response to the existence of any out-of-tolerance points will be to refine the surface and repeat the fitting process.

The situation of most interest arises after the surface has been sufficiently refined to provide regions that fit within tolerance, separating regions that are out of tolerance (Figure 1).

An out-of-tolerance region contained within a sufficient number of patches can be treated as an overlay forming a separate fitting problem, constrained around its perimeter as in the discussion of Section 5. Figure 2 shows a suitable refinement of the domain in Figure 1 for the out-of-tolerance region at the upper left. This region is treated as an overlay, assuming uniform, bicubic, B-spline patches. For such splines, at least four patches in either parametric direction are needed so that there exists any control vertex free to move after three control vertices around each margin have been constrained. Constraining the margins is necessary in order for the overlay to maintain its integrity with the surface on which it rests.

The process of finding such separable regions enclosed in rectangular arrays of patches can be treated as the construction of bounding boxes around connected regions in a raster display. Standard fill algorithms can be employed.

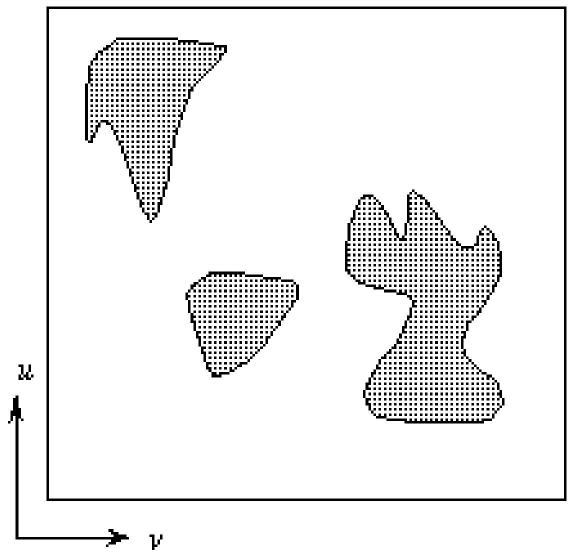


Figure 1: Areas of the surface domain corresponding to data outside of tolerance.

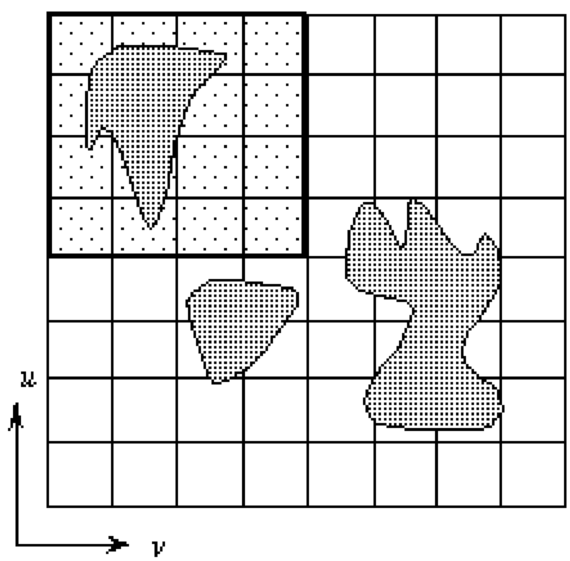


Figure 2: Refinement of the domain sufficient to enclose an area as a separate fitting region.

The fitting process can be summarized algorithmically. For the sake of uniformity, the initial stages of the fitting process can be regarded as working with overlays consisting of the entire surface supplied with an extra band of constrained control vertices around the perimeter. We begin with the initial template surface as a single overlay, compute residuals and test against the given tolerance, and designate the level of refinement to be $k = 0$.

```
while ( points out of tolerance remain ) }
    for ( each separate out of tolerance region in level k ) {
        solve the least squares problem on the overlay;
        calculate the resulting residuals;
        determine any separable out of tolerance regions and
            add each to the list of regions for level k + 1;
    }
    k = k + 1;
}
```

Economical storage is possible throughout the levels of the fitting process. To fit an overlay surface at level $k+1$ of refinement, the overlay is first generated as referenced to the surface of level k from which it is derived. The fitting process is viewed as modifying the offset information of the overlay [4]. Thus the least squares equations become

$$\mathbf{B}(\mathbf{R} + \mathbf{O})\mathbf{C}^T = \mathbf{P} \tag{46}$$

rather than (5), and the variables of the fitting process can be regarded as the offsets $\mathrm{O}_{i,j}$. This replaces the least squares problem by

$$\mathbf{BOC}^T = \mathbf{P} - \mathbf{BRC}^T \tag{47}$$

It is a result of the convex hull property of the basis functions that we are considering that any offset whose value is less than the fitting tolerance can be regarded as zero without perturbing the resulting overlay by more than the tolerance. Zero offsets often result from the local support of the spline basis being used. If portions of the surface at refinement level k are fit within tolerance, a number of offsets at level $k + 1$ are likely to be smaller in magnitude than the given tolerance. This can be true even if the regions of good fit cannot be conveniently enclosed in bounding boxes of the kind shown in Figure 2. Zero offsets do not have to be recorded in the data structure of a hierarchical spline surface, yielding a certain economy of storage as compared with a full tensor-product spline representation.

7 Comparison with Multiresolution Analysis

In preparing this paper and presenting its material to various groups, the question has come up about its possible relationship to wavelets techniques. We believe that, while there is a theoretical association, there are a number of distinctions that currently make the association of little pragmatic value. Briefly, (1) wavelet techniques currently gain their computational advantages only in a setting in which refinements are known and fixed in advance, (2) wavelet techniques have chiefly been developed for continuous inner products, and our data fitting context requires a discrete inner product, and (3) our fitting process is intended to be useful in a parametric setting, while wavelet techniques have been most fully developed in a functional setting. This section will sketch out a bit more background on these matters.

Since the two variables and basis functions of tensor product splines are independent of each other, it is sufficient (and more convenient) to carry out the discussion of this section in terms of the single variable u.

We begin with the simplest case of the fitting process, that for which the basis functions $B_i(u)$ are restricted to be the *canonical B-splines*; that is, B-splines of order ω with knots of single multiplicity placed at the integer positions. The only refinement permitted for such splines is to allow the insertion of exactly one knot at the midpoint between every two existing knots. In this restricted setting, the fitting process takes place in the setting usually encountered for multiresolution analysis [2]. Letting $B_{0,i}(u) = B_i(u)$ stand for the canonical B-splines, we note that $B_{0,i}(u) = B_{0,0}(u+i)$, where $B_{0,0}(u)$ is the canonical B-spline with support on the interval $[0, \omega]$ (knots $0, 1, \ldots, \omega$) and $i = 0, \pm 1, \pm 2, \ldots$. The B-splines encountered at step j of the refinement process will be given by $B_{j,i}(u) = B_{0,0}(2^j u + i)$. The index j can be regarded as giving the frequency octave of the B-splines and the index i gives the shift.

Let S_j^ω consist of all splines of the form

$$s_j(u) = \sum_{i=-\infty}^{+\infty} c_{j,i} B_{j,i}(u)$$

which are L^2 integrable

$$\int_{-\infty}^{+\infty} |s_j(u)|^2 \, dt < \infty$$

let V_j^ω consist of all L^2 functions $v_j(u)$ that can be the L^2 limits of sequences of such $s_j(u)$

$$\lim_{k\to\infty}\int_{-\infty}^{+\infty}\left|s_j^k(u)-v_j(u)\right|^2 du = 0$$

that is, V_j^ω consist of all L^2 functions that can be approximated arbitrarily closely by splines composed from the basis $B_{j,i}(u)$. The following are established in [2]:

Multiresolution Analysis Setting

1. $\cdots \subset V_{-1}^\omega \subset V_0^\omega \subset V_1^\omega \cdots$
2. Any $v \in L^2$ can be approximated arbitrarily closely in $\bigcup_j V_j^\omega$
3. $\bigcap_j V_j^\omega = \emptyset$
4. $v(t) \in V_j^\omega \Leftrightarrow v(2t) \in V_{j+1}^\omega$
5. $V_{j+1}^\omega = V_j^\omega \oplus W_j^\omega$

The space W_j^ω is the *orthogonal complement* of V_j^ω in V_{j+1}^ω. Given any L^2 function $f(u)$,

$$\begin{aligned} f(u) &= \cdots + g_{j-1}(u) + g_j(u) + g_{j+1}(u) + \cdots \\ &= f_j(u) + g_j(u) + g_{j+1}(u) + \cdots \end{aligned}$$

where $f_j(u) \in V_j^\omega$ and $g_j(u) \in W_j^\omega$. The term $f_j(u)$ represents an approximation to $f(u)$ up to the frequency or level of resolution given by all shifts of $B_{\mu,0}(u)$, $\mu < j$. The term $g_j(u)$ provides the extra level of resolution for frequency j.

For any given tolerance, $f(u)$ can be approximated within that tolerance by truncating the series, discarding all terms above some $j = J$:

$$f(u) \approx f_J(u) = \cdots + g_{J-2}(u) + g_{J-1}(u)$$

Moreover, since the B-splines have compact support, if we wished to approximate $f(u)$ on a compact set, $f_J(u)$ would reduce to $\sum_i c_i^J B_{J,i}(u)$ for i on a finite set of integers. A classical approach for determining coefficients c_i^J would be to solve the least squares problem via *normal equations*, which involves the *Gram matrix* having (α,β) entry as the value of the inner product $\langle B_{J,\beta}, B_{J,\alpha}\rangle$ where

$$\langle B_{J,\beta}, B_{J,\alpha}\rangle = \int_{-\infty}^{+\infty} B_{J,\beta}(u)\overline{B_{J,\alpha}(u)}du$$

(the overbar indicates complex conjugation) for continuous least squares fitting and

$$\langle B_{J,\beta}, B_{J,\alpha}\rangle = \sum_i B_{J,\beta}(u_i)\overline{B_{J,\alpha}(u_i)}$$

for least squares fitting based on discretely sampled data. Similarly the *transformed right-hand side* has α entry $\langle f(u)B_{J,\alpha}(u)\rangle$.

In multiresolution analysis this classical approach is not taken. Instead, special properties of the functions $B_{j,i}$, and of the g_j resulting from the Multiresolution Analysis Setting above, permit finding the approximation $f_J(u)$ very efficiently. (However, only the continuous least squares fitting case is treated in the literature.)

As a preliminary remark, we note that frequencies below a certain level usually contribute little to f, so that $g_{J-L} + \cdots + g_{J-1}$ could be used as an approximation to f in place of f_J; that is,

$$\begin{aligned} f_J(u) &= f_{J-1}(u) + g_{J-1}(u) \\ &= g_{J-1}(u) + \cdots + g_{J-L}(u) + f_{J-L}(u) \end{aligned}$$

and $f_{J-L}(u)$ is discarded.

Functions $\psi_{j,i}(u)$, called the *B-wavelets*, are a convenient basis for W_j^ω:

$$g_j(u) = \sum_i d_i^j \psi_{j,i}(u)$$

The coefficients d_i^j are, in theory, obtainable as $\left\langle f(u)\tilde{\psi}_{j,i}(u)\right\rangle$, where the functions $\tilde{\psi}_{j,i}(u)$ are the *B-wavelet duals*. In practice, the d_i^j and coefficients c_i^j in the expansion

$$f_j(u) = \sum_i c_i^j B_{j,i}(u)$$

can be computed very efficiently together in an interlaced bootstrapping fashion that involves only finite moving averages.

The result is an approximation satisfying

$$|\langle f(u) - g_{J-1}(u) - \cdots - g_{J-L}(u), f(u) - g_{J-1}(u) - \cdots - g_{J-L}(u)\rangle| < \epsilon_0^2$$

for a given tolerance ϵ (using the coefficients d), or an equally useful approximation

$$|\langle f(u) - f_J(u), f(u) - f_J(u)\rangle| < \epsilon_1^2$$

(using the coefficients c).

In [8] the Multiresolution Analysis Setting has been extended fully to B-splines. The nested spaces V_j^ω consist of any that can be composed by arbitrary knot insertions. Algorithms are given for computing the minimally supported B-wavelets in this setting.

A further construction method for computing spline wavelets that extends to discrete inner products; i.e., $\sum_i f(u_i)\overline{g(u_i)}$ instead of $\int_{-\infty}^{+\infty} f(u)\overline{g(u)}du$ is given in [10]. This is the setting that corresponds to the surface fitting approach we are using. However, three obstacles stand in our way of obtaining our surface fits via B-wavelets:

- In our setting we wish to determine the position at which knots will be inserted dynamically. This means that the B-wavelet construction could not be carried out in advance, which destroys the efficiency with which the coefficients (control vertices) for the fitting surface can be produced via wavelets. The gains provided by a wavelet approach depend fundamentally on the ability to precompute the wavelet basis.

- The hierarchical representation [4] that we wish to obtain involves geometric information not contained in a wavelet representation. Specifically, each refinement surface produced during the fitting process has control vertices represented in coordinate frames sited upon the parent surface from which it was obtained. This geometric information is a representational option made possible by the fact that we are fitting *parametric* splines rather than *functional* splines, and there has, as yet, been no significant investigations toward applying wavelet techniques to the parametric setting.

- While it is possible to use a discrete inner product to define the orthogonality between the spaces V and W of the multiresolution setting, neither practical nor theoretical studies have been carried out to determine the effectiveness of this in the functional setting, much less in the parametric setting.

8 Examples

All examples in this section were generated using uniform, bicubic, B-spline surfaces.

Plate 1 shows laser range data from a section of a simple machined part. The data has been filtered to remove noise and discontinuities. It is displayed in this plate as an array of tiny, shaded rectangles, each rectangle being composed from four data points in the obvious way. There are 29200 points in the data, so each rectangle is too small to be seen.

Plates 2 through 16 show various aspects of a simple refinement process: midpoint subdivision. The template surface used to begin the fit consisted of a single bicubic patch. The different levels of subdivision are shown in the even-numbered plates 2 through 14. Each odd-numbered plate 3 through 15 displays the original data colored according to the residuals between the data and the approximating spline surface of the previous, even-numbered plate; that is, Plate 2 shows the single, bicubic patch fit and Plate 3 displays the corresponding residuals. The green/white colors indicate areas in which the difference between the spline patches and the data are less than a preset tolerance, with white indicating a closer fit than green. The yellow/orange colors indicate areas in which the difference is larger than the tolerance, with orange indicating a larger difference than yellow. Along with this information about residuals, some information about the approximating spline can be seen. The patch structure is visible in the red patch-boundary lines shown in the odd-numbered plates. The fitting surface is colored according according to its level of subdivision in the even-numbered plates.

Plate 10 shows the fourth level of subdivision, which produces a fitting surface of 256 patches. At this level the fitting process required the factoring of two matrices, each of size 172×16 with 172 backsolves required for each matrix. The residual display in Plate 11 clearly shows two groups of isolated areas, each group located near the corners to the left and right, on which the residuals are still beyond tolerance. These areas were separated by the algorithm into two separate hierarchical problems for the next level of fitting.

Plate 12 shows the result of this hierarchical fit. The fitting surface consists of the surface of Plate 10 corrected by two hierarchical overlays, one consisting of $10 \times 20 = 200$ patches and the other consisting of $14 \times 32 = 448$ patches. The residual display of Plate 13 shows two smaller groups of residuals, which generate the next level of hierarchical corrective fits.

Plate 14 shows the last fit in the sequence, consisting of the hierarchical surface of Plate 12 with two further hierarchical corrections, one consisting of $12 \times 18 = 216$ patches and the other consisting of $18 \times 18 = 324$ patches. Plate 15 shows the corresponding residuals, all of which are within tolerance. The final surface has 1177 bicubic patches, which, counting patch adjacency, the sharing of control vertices, and econimizations for zero offsets [4], results in a significant reduction in storage from the original data.

Plate 16 shows the patch boundaries and the *edit points* of [4] for the surface of Plate 14. The surface produced can be modified using the methods of [4], and the editor of that reference was used as the means of rendering the successive fits.

Plate 17 shows data, kindly provided by Demitri Terzopoulos, that was used in [9]. As in Plate 1, the rendering is performed using rectangles from successions of data points and shading the resulting polygonal surface. Plate 18 shows a succession of spline surfaces produced from midpoint refinements, and Plate 19 shows an enlarged view of the final fit in Plate 18.

Plate 20 shows, in wire frame, a set of data used for adaptive, non-midpoint refinement [12]. The surface to be fit is flat save for a spike in the center. Plate 21 shows the patch structure of the resulting fit, Plate 22 shows normal vectors for the resulting fit, and Plate 23 shows the fit surface colored according to its maximum curvature component.

PLATES REMOVED TO CONSERVE SPACE

References

[1] de Boor, Carl. **A Practical Guide to Splines**. Springer-Verlag (1978).

[2] Chui, Charles. **An Introduction to Wavelets**. Academic Press (1992).

[3] Dierckx, Paul. An Algorithm for Least-Squares Fitting of Cubic Spline Surfaces to Functions on a Rectilinear Mesh over a Rectangle. Journal of Computational and Applied Mathematics 3, 2 (month, 1977), 113-129.

[4] Forsey, David, and Bartels, Richard. Hierarchical B-Spline Refinement. Proceeding of SIGGRAPH '88 (Atlanta, Georgia, August 1-5). In Computer Graphics 22, 4 (August, 1988), 205-212.

[5] Golub, Gene, and van Loan, Charles. **Matrix Computations**. The Johns Hopkins University Press (1983).

[6] Graham, Alexander. **Kronecker Products and Matrix Calculus with Applications**. Halsted Press (1981).

[7] Lawson, Charles, and Hansen, Richard. **Solving Least Squares Problems**. Prentice-Hall (1974).

[8] Lyche, Tom, and Mørken, Knut. Spline-Wavelets of Minimal Support. Preprint 1992-4, Institutt for Informatikk, Universitetet i Oslo, Postboks 1080 Blindern, N-0316 Oslo 3, Norway (1992).

[9] Schmitt, Francis, Barsky, Brian, and Du, Wen-Hui. An Adaptive Subdivision Method for Surface-Fitting from Sampled Data. Proceedings of SIGGRAPH '86 (Dallas, Texas, August 18-22, 1986). In Computer Graphics 20, 4 (August, 1986), 179-188.

[10] Sivalingam, Subendran, and Bartels, Richard. Matrix-nullspace Wavelet Construction. in Mathematical Methods in CAGD III, M. Dæhlen, T. Lyche, and L. L. Schumaker (eds.), Academic Press [to appear].

[11] Späth, H. **Eindimensionale Spline-Interpolations-Algorithmen**. R. Oldenbourg (1990).

[12] Srećković, Milan. Adaptive Hierarchical Fitting of Curves and Surfaces. Master's Thesis, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1 (1992).