ELSEVIER Computer Aided Geometric Design 13 (1996) 351–368

COMPUTER AIDED GEOMETRIC DESIGN

Interactive surface correction based on a local approximation scheme ☆

Bernd Hamann [a,*], Brian A. Jean [b,1]

[a] *Department of Computer Science, University of California, Davis, CA 95616-8562, USA*
[b] *NSF Engineering Research Center for Computational Field Simulation, Mississippi State University, P.O. Box 6176, Mississippi State, MS, 39762, USA*

Received July 1993; revised February 1994

Abstract

The paper presents a new interactive technique for correcting CAD/CAM data containing errors. An algorithm is described that can be used to correct surface data containing undesirable discontinuities ("gaps"/"holes," and "overlaps") and intersections among surface patches. Such surface problems commonly arise in the aircraft, automobile, and ship industry and make later processing of the data difficult, if not impossible. The new method provides a tool to correct wrong data requiring minimal user interaction. The input for the scheme can be a set of parametrically defined surfaces (e.g., Bézier, B-spline, or NURBS surfaces) or a set of triangles (or quadrilaterals) discretizing the original geometry. The output is a set of G^0 or G^1 (tangent plane) continuous, bicubic B-spline surfaces approximating the given data. Each of these B-spline surfaces is constructed using just four user-specified boundary curves.

Keywords: Approximation; B-spline surface; Coons surface; Grid/mesh generation; Hardy's reciprocal multiquadric method; Scattered data interpolation; Transfinite interpolation

1. Introduction

In most CAD/CAM systems, three-dimensional geometries are described in terms of parametric surfaces, e.g., Bézier, B-spline, or NURBS (nonuniform rational B-spline) surfaces. Very often, the geometries created by such a system contain inconsistencies. These inconsistencies consist of surface patches that do not connect properly or patches

☆ This work was supported by the National Grid Project consortium and the National Science Foundation.
* Corresponding author. E-mail: hamann@cs.ucdavis.edu.
[1] E-mail: brian@erc.msstate.edu.

SSDI 0167-8396(95)00049-6

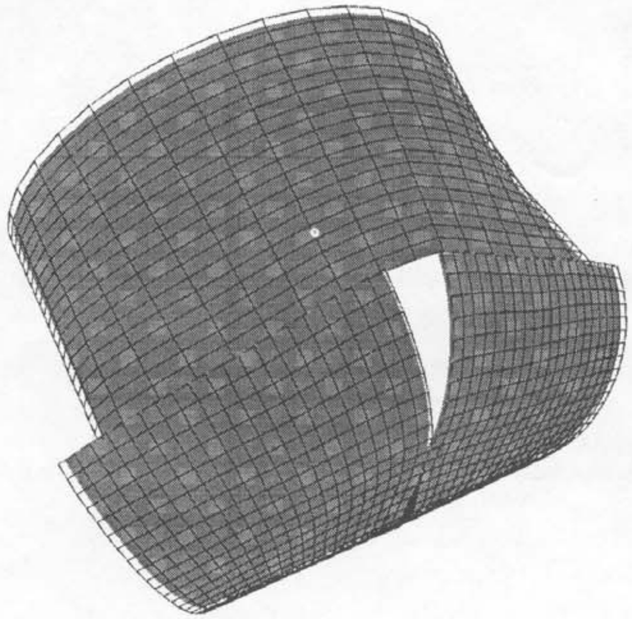

Fig. 1. Geometry containing "hole".

that intersect each other in ways that can not be used by some processes, e.g., grid generation, flow simulation, and manufacturing.

A critical issue in numerical grid generation is the correctness of the given geometry from which a mesh must be constructed. The grid generation system currently under development at the NSF Engineering Research Center for Computational Field Simulation (at Mississippi State University) contains an interactive surface correction module that is described in this paper. Before surface and volume grids can be automatically generated, it is best if the geometry is free of "gaps"/"holes," "overlaps," and transverse surface intersections. The method presented helps the user correct a given geometry containing errors by creating a new set of surface patches that replace the original ones.

The technique is based on the interactive creation of surface patches by projecting local approximants onto the given geometry. Essentially, a new surface patch is constructed by specifying four boundary curves, computing a Coons patch interpolating these curves, and projecting the Coons surface onto the original patches. The projection of the Coons surface onto the original surfaces replaces the given geometry (locally) and will later be used for further processing of the data. An interactive surface correction strategy is used, since it is the authors' belief that a completely automated approach is not feasible. However, the necessary user input is reduced to an absolute minimum. An example for "bad" geometry data is the 3D configuration shown in Fig. 1.

Neither the geometric modeling nor the grid generation literature provide help for solving the problem described. Most grid generation systems contain simple CAD functionality used to correct wrong CAD/CAM data. However, the correction step typically requires more time than the task of creating the final grid. The underlying mathematical methods are well understood in geometric modeling and have been combined with the goal of simplifying and accelerating the process of correcting 3D geometries.

The methods used throughout this paper have their origins in geometric modeling (computer aided geometric design), grid generation, and partially in solid modeling. Fundamental geometric modeling concepts are described in (Bartels et al., 1987; Faux and Pratt, 1979; Yamaguchi, 1988; Hosaka, 1992; Farin, 1993; Hoschek and Lasser, 1993). Different grid/mesh generation techniques are given in (Thompson et al., 1985; George, 1991). Algorithms related to NURBS curves and surfaces are given in (Piegl, 1991). Some aspects of solid modeling are discussed in (Mortensen, 1985).

2. The overall algorithm

The underlying idea is based on constructing an initial local surface approximant, which is projected onto the given geometry thereby defining a new B-spline surface. Over the whole model, the individual B-spline surfaces constructed are unioned and replace the original data. The entire approximation process is characterized by these steps:

(i) Creating/selecting four surface boundary curves
(ii) Constructing a bilinear Coons patch from four curves
(iii) Clipping the original surfaces against a 3D box containing the Coons patch
(iv) Projecting a set of points on the Coons patch onto original surfaces
(v) Determining "artificial projections" if previous step fails
(vi) Interpolating the points resulting from (iv) and (v)
(vii) Estimating the error between surface approximant and original surfaces
(viii) Enforcing continuity conditions along boundaries of all approximants

Step (i) is done by the user and will not be described here. The locally approximating surface obtained in step (vi) is a bicubic B-spline surface. The error estimate described in Section 7 is based on shortest (perpendicular) distances between points on the approximating surface and the original ones. Newton's method is used to solve this bivariate minimization problem. If the error estimate exceeds a specified tolerance, the number of points on the Coons patch projected onto the original surfaces (sampling rate) is increased.

Step (viii) is necessary, since all single local surface approximants are constructed independently from each other and generally are discontinuous along/across their boundary curves. These "small" discontinuities can easily be removed by adjusting corresponding control point pairs of surface approximants that must share a common boundary curve. The fact that two approximants actually should be "connected" (i.e., should be G^0 or G^1 (tangent plane) continuous along a boundary curve can be determined by considering the "distance" between pairs of boundary curves of different approximants. Unfortunately, the continuity conditions along boundary curves imply severe restrictions on the number of control points and the knot vectors used.

In Section 3, the construction on the initial Coons patch is described. This algorithm constructs a discretized patch. The construction of a bounding box of offsets of the approximant and its use in clipping are described in Section 4. The original geometry is clipped to aid performance and resolve ambiguities during projection. The projection is covered in Section 4 as well. Since the data may contain holes, the projection may have no solution. The technique used to determine additional points that span holes is described in Section 5.

3. Computing the initial Coons patch

The first step in correcting the given data is to construct an initial approximation to the desired geometry. This initial surface patch will be used to smooth rough data, guide the choices of interpolation points, and serve as a reference for filling in gaps.

For the computation of a single approximant, the user must specify four curves. These curves can be boundary curves of existing patches or can be created as new entities in space or on the original patches. It should be mentioned at this point that curves and surfaces are represented in NURBS format in the system under development. Examples of possible boundary curves are line segments, conics, or cubic splines. It is assumed that each of the curves specified by the user is C^1 continuous and the ends meet to form a single closed curve. The use of four arbitrary curves as input for the approximation technique makes it possible to retain existing boundaries of original patches. In general, these four curves can span across multiple original surface patches; they can even be "above" and "below" the given geometry.

The four curves specified by the user are discretized and blended bilinearly and define a "discrete" Coons patch (transfinite interpolation). The order in which these curves are specified implies the orientation of the patch. This Coons patch can be viewed as an initial local geometry approximant that is later used to compute a much better approximant. The Coons patch is defined over the unit square $[0,1] \times [0,1]$ and has the four boundary curves

$$\begin{aligned} &\boldsymbol{x}(u,0) = \boldsymbol{c}_1(u),\ \boldsymbol{x}(u,1) = \boldsymbol{c}_2(u),\ u \in [0,1], \\ &\boldsymbol{x}(0,v) = \boldsymbol{d}_1(v),\ \text{and}\ \boldsymbol{x}(1,v) = \boldsymbol{d}_2(v),\ v \in [0,1]. \end{aligned} \tag{1}$$

The bilinear Coons patch is written as

$$\begin{aligned} \boldsymbol{x}(u,v) = {}& [(1-u)\ u] \begin{bmatrix} \boldsymbol{x}(0,v) \\ \boldsymbol{x}(1,v) \end{bmatrix} + [\boldsymbol{x}(u,0)\ \boldsymbol{x}(u,1)] \begin{bmatrix} (1-v) \\ v \end{bmatrix} \\ & - [(1-u)\ u] \begin{bmatrix} \boldsymbol{x}(0,0) & \boldsymbol{x}(0,1) \\ \boldsymbol{x}(1,0) & \boldsymbol{x}(1,1) \end{bmatrix} \begin{bmatrix} (1-v) \\ v \end{bmatrix}, \quad u,v \in [0,1]. \end{aligned} \tag{2}$$

If the Coons patch is evaluated at uniformly distributed parameter values (u_I, v_J) yielding a structured surface grid, the resulting 3D points are

$$\boldsymbol{x}_{I,J} = \boldsymbol{x}(u_I, v_J) \quad u_I = \frac{I}{M},\ I = 0,\ldots,M, \quad v_J = \frac{J}{N},\ J = 0,\ldots,N. \tag{3}$$

Unfortunately, this surface grid might be highly nonuniform due to the parametrization of the four boundary curves. In order to obtain a reasonable surface grid for the Coons patch, a "discrete" construction of the Coons patch is preferred. The construction of the "discrete" Coons patch is achieved by first computing points on the boundary curves distributed uniformly according to the curves' arc lengths. One can then associate parameter values $(u_{I,0}, v_{I,0})$, $(u_{I,N}, v_{I,N})$, $(u_{0,J}, v_{0,J})$, and $(u_{M,J}, v_{M,J})$, defining (nearly) uniformly distributed points on the four boundary curves of the Coons patch. The points on this "discrete" Coons patch are

$$\boldsymbol{x}_{I,J} = \begin{bmatrix} (1-u_{I,J}) & u_{I,J} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{0,J} \\ \boldsymbol{x}_{M,J} \end{bmatrix} + \begin{bmatrix} \boldsymbol{x}_{I,0} & \boldsymbol{x}_{I,N} \end{bmatrix} \begin{bmatrix} (1-v_{I,J}) \\ v_{I,J} \end{bmatrix} - \begin{bmatrix} (1-u_{I,J}) & u_{I,J} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{0,0} & \boldsymbol{x}_{0,N} \\ \boldsymbol{x}_{M,0} & \boldsymbol{x}_{M,N} \end{bmatrix} \begin{bmatrix} (1-v_{I,J}) \\ v_{I,J} \end{bmatrix}, \tag{4}$$

where $u_{I,J}$, $J = 0, \ldots, N$, varies linearly between $u_{I,0}$ and $u_{I,N}$ and $v_{I,J}$, $I = 0, \ldots, M$, varies linearly between $v_{0,J}$ and $v_{M,J}$, respectively.

The Coons patch is used to roughly approximate the given geometry locally. A better approximation is obtained by projecting points of the Coons patch onto the given geometry and interpolating the projections.

4. Projecting the Coons patch onto the original surfaces

Each point $\boldsymbol{x}_{I,J}$ on the Coons patch is projected onto the original surfaces. It is explained what is meant by "projection," how to reduce the number of original surfaces considered for this step, and how to resolve the problem that arises when multiple or no projection points are obtained for a point $\boldsymbol{x}_{I,J}$.

First, the unit normal vector is computed at $\boldsymbol{x}_{I,J}$. It is given by

$$\boldsymbol{n}_{I,J} = \boldsymbol{n}(u_{I,J}, v_{I,J}) = \left. \frac{\frac{\partial}{\partial u}\boldsymbol{x}(u,v) \times \frac{\partial}{\partial v}\boldsymbol{x}(u,v)}{\left\| \frac{\partial}{\partial u}\boldsymbol{x}(u,v) \times \frac{\partial}{\partial v}\boldsymbol{x}(u,v) \right\|} \right|_{(u_{I,J}, v_{I,J})}, \tag{5}$$

where $\| \ \|$ denotes the Euclidean norm. The partial derivatives can be obtained directly from the bilinear Coons patch. Since the Coons patch has been discretized, the normal vector at a point $\boldsymbol{x}_{I,J}$ is approximated using central differences (and this is done in the implementation):

$$\boldsymbol{n}_{I,J} \approx \frac{(\boldsymbol{x}_{I+1,J} - \boldsymbol{x}_{I-1,J}) \times (\boldsymbol{x}_{I,J+1} - \boldsymbol{x}_{I,J-1})}{\|(\boldsymbol{x}_{I+1,J} - \boldsymbol{x}_{I-1,J}) \times (\boldsymbol{x}_{I,J+1} - \boldsymbol{x}_{I,J-1})\|}. \tag{6}$$

The points $\boldsymbol{x}_{I,J}$, their associated normal vectors $\boldsymbol{n}_{I,J}$, and the two distances $\pm d$ define points on an "upper" and a "lower" offset of the initial approximant. The points on the "upper" offset surface are denoted by $\boldsymbol{a}_{I,J}$, while the ones on the "lower" offset surface are denoted by $\boldsymbol{b}_{I,J}$. These points are

$$\boldsymbol{a}_{I,J} = \boldsymbol{x}_{I,J} + d\boldsymbol{n}_{I,J} \quad \text{and} \quad \boldsymbol{b}_{I,J} = \boldsymbol{x}_{I,J} - d\boldsymbol{n}_{I,J}. \tag{7}$$

The offset distance d is related to the extension of the Coons patch by setting

$$d = \frac{1}{8}\big(\|\boldsymbol{x}_{M,0} - \boldsymbol{x}_{0,0}\| + \|\boldsymbol{x}_{M,N} - \boldsymbol{x}_{M,0}\| \\ + \|\boldsymbol{x}_{0,N} - \boldsymbol{x}_{M,N}\| + \|\boldsymbol{x}_{0,0} - \boldsymbol{x}_{0,N}\|\big), \tag{8}$$

i.e., the offset distance reflects the extension of the quadrilateral given by the four corner points of the Coons patch. One could even allow a varying offset distance, e.g., a bilinearly varying value for d across the Coons patch. The choice of d is crucial, since it determines the set of original surfaces to be considered for the approximation. It is not clear at this point what is the best value for d for an arbitrary geometry. The construction of the points on the two offset surfaces is shown in Fig. 2.

It is assumed that the convex set defined by all the points $\boldsymbol{a}_{I,J}$ and $\boldsymbol{b}_{I,J}$ contains the (parts of) original surfaces that must be considered by the local approximation procedure. The convex hull of this convex set is approximated by computing a 3D bounding box for the point set $\{\boldsymbol{a}_{I,J}\} \cup \{\boldsymbol{b}_{I,J}\}$. Original surfaces are considered for the local approximation procedure only if they lie partially (or completely) inside this bounding box. Using Bézier, B-spline, or NURBS surface representation, all original surfaces that have no control points inside this bounding box do not have to be considered.

The surfaces whose control points indicate the possibility of lying (partially) inside the bounding box are evaluated using some predefined resolution. The resulting surface point set is triangulated, and another test is then applied to each triangle. If a triangle's vertices are all left of (right of, in front of, behind, below, above) the bounding box it is no longer considered; only those triangles possibly lying (partially) inside the bounding box are needed when projecting a point $\boldsymbol{x}_{I,J}$ onto the original surfaces. It is possible to reduce the number of triangles even further when projecting a particular point $\boldsymbol{x}_{I,J}$ onto the given surfaces. It is unknown at this point for which values for M and N such a further reduction is advantageous. Clipping is illustrated in Fig. 3.

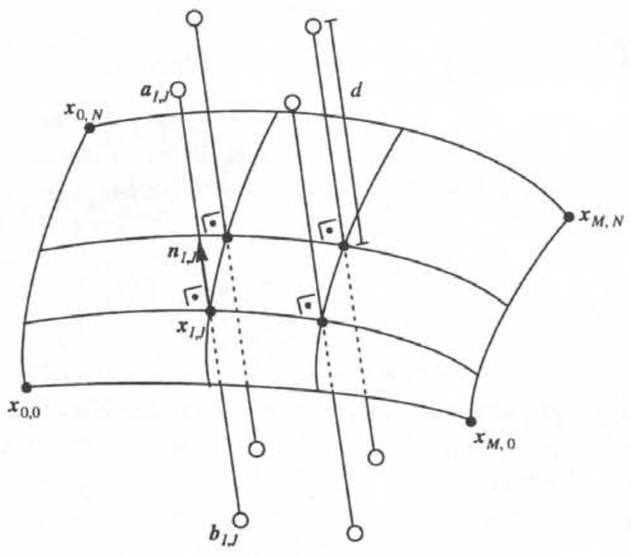

Fig. 2. Construction of point pairs on offset surfaces.

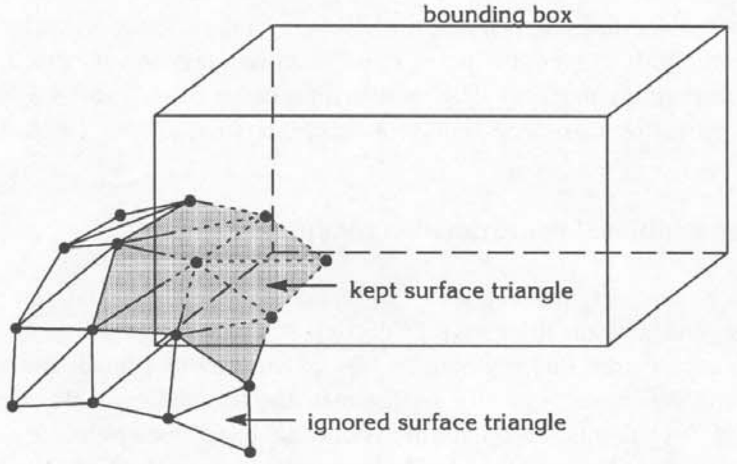

Fig. 3. Clipping original surfaces and surface triangles.

Having determined the set of triangles lying (partially) inside the bounding box, each point $\boldsymbol{x}_{I,J}$ is projected in direction of $\boldsymbol{n}_{I,J}$ onto these triangles. The projection of $\boldsymbol{x}_{I,J}$ must satisfy the condition that it lie between $\boldsymbol{a}_{I,J}$ and $\boldsymbol{b}_{I,J}$. This ensures a local "behavior" of the approximation technique. Projecting a point $\boldsymbol{x}_{I,J}$ onto the set of triangles might yield zero, one, or multiple projections onto triangles.

No problem arises if one intersection is found between line segment $\overline{\boldsymbol{a}_{I,J}\boldsymbol{b}_{I,J}}$ and the set of triangles. If more than one intersection is found, the point closest to the point $\boldsymbol{x}_{I,J}$, the point on the initial Coons patch, is chosen. The reason for this is that, in general, the initial Coons patch should already be very close to the part of the geometry to be approximated. Notice this choice is independent of the orientation of the Coons patch, since "upper" and "lower" offsets were used. If no intersection is found, a bivariate scattered data approximation method is used to derive "artificial projections." This is described in the next section.

If the parametric representation of the original surfaces is known the projections (currently lying in the interior of certain triangles) can be mapped onto the real surfaces. If a parametric representation is not known, i.e., the original geometry is given by a set of triangles or quadrilaterals, this adjustment step can not be performed. All projections obtained from intersecting line segments $\overline{\boldsymbol{a}_{I,J}\boldsymbol{b}_{I,J}}$ and surface triangles are points in the planes containing these triangles. Therefore, each intersection point can be expressed as a barycentric combination of the vertices of the triangle and be mapped onto the actual surface. Let $\boldsymbol{p}_{I,J}$ be an intersection point written as

$$\boldsymbol{p}_{I,J} = \bar{u}_1\boldsymbol{p}_1 + \bar{u}_2\boldsymbol{p}_2 + \bar{u}_3\boldsymbol{p}_3, \tag{9}$$

where $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, and $\boldsymbol{p}_3$ are the vertices of the triangle containing $\boldsymbol{p}_{I,J}$. Knowing the barycentric coordinates of $\boldsymbol{p}_{I,J}$, one computes the parameter tuple

$$(\bar{u}, \bar{v}) = \bar{u}_1(u_1, v_1) + \bar{u}_2(u_2, v_2) + \bar{u}_3(u_3, v_3), \tag{10}$$

where (u_i, v_i) is the parameter tuple of vertex $\boldsymbol{p}_i$. The original parametric surface $\boldsymbol{s}(u, v)$ containing $\boldsymbol{p}_i$ is evaluated at $(\bar{u}, \bar{v})$, and $\boldsymbol{p}_{I,J}$ is replaced by $\boldsymbol{s}(\bar{u}, \bar{v})$. Assuming that the resolution chosen for the representation of the original surfaces is fairly high and the surfaces are not highly curved, the point $\boldsymbol{s}(\bar{u}, \bar{v})$ can not deviate very much from $\boldsymbol{p}_{I,J}$. It has been found that this mapping is advisable for a better overall surface approximation. The points obtained by this "map-onto-real-surface technique" are denoted by $\boldsymbol{y}_{I,J}$.

5. Computing additional approximation conditions

Unfortunately, the original surfaces might be discontinuous and therefore contain "gaps"/"holes." As a result the above projection strategy might not yield any intersection point for a particular line segment $\overline{\boldsymbol{a}_{I,J}\boldsymbol{b}_{I,J}}$. On the other hand, the approximation technique that is used later to locally approximate the original geometry requires exactly $(M+1) \times (N+1)$ points. To obtain the additional conditions points are estimated for line segments without intersection(s). The computational method chosen is based on a bivariate scattered data approximation technique, called "Hardy's reciprocal multiquadric method" (Franke, 1982).

Each intersection point $\boldsymbol{p}_{I,J}$ obtained by intersecting line segment $\overline{\boldsymbol{a}_{I,J}\boldsymbol{b}_{I,J}}$ with the surface triangles can be written as a linear combination of $\boldsymbol{a}_{I,J}$ and $\boldsymbol{b}_{I,J}$, i.e.,

$$\boldsymbol{p}_{I,J} = \boldsymbol{p}(t_{I,J}) = (1 - t_{I,J})\, \boldsymbol{a}_{I,J} + t_{I,J}\, \boldsymbol{b}_{I,J}, \quad t_{I,J} \in [0,1]. \tag{11}$$

The values $t_{I,J}$ are computed (and stored) when performing the projection step. These values remain unchanged, even if intersection points are mapped onto the real parametric surfaces. Hardy's reciprocal multiquadric method is used to compute a bivariate function $t(u,v)$ interpolating all parameter values $t_{I,J}$ corresponding to intersection points. Using a global scattered data interpolation approach, the conditions to be considered are

$$\begin{aligned} t_{I,J} &= t(u_{I,J}, v_{I,J}) \\ &= \sum_{j\in\{0,\dots,N\}} \sum_{i\in\{0,\dots,M\}} c_{i,j} \left(R + (u_{I,J} - u_{i,j})^2 + (v_{I,J} - v_{i,j})^2\right)^{-\gamma}, \\ & I \in \{0,\dots,M\},\ J \in \{0,\dots,N\}, \end{aligned} \tag{12}$$

where R and γ are fixed parameters and only those values $t_{I,J}$, $u_{I,J}$, $u_{i,j}$, $v_{I,J}$, and $v_{i,j}$ are considered for which an intersection point has been found. Denoting the "average" parameter spacing in the two parameter directions by

$$\delta_u = \frac{1}{2M} \sum_{I=0}^{M} (u_{I+1,0} - u_{I,0}) + (u_{I+1,N} - u_{I,N})$$

and

$$\delta_v = \frac{1}{2N} \sum_{J=0}^{N} (v_{0,J+1} - v_{0,J}) + (v_{M,J+1} - v_{M,J}),$$

it has been found that the value $R = 0.5(\delta_u + \delta_v)$ and the exponent $\gamma = 0.001$ yield good results.

In most cases, this global approach (considering *all* intersections found) has a significant drawback: If the number of missing intersections is very small relative to

$(M+1) \times (N+1)$, it is computationally too expensive to solve the implied linear system of equations. Therefore, a localized version of Hardy's reciprocal method is used. This method only considers a certain number of found intersections surrounding a "hole."

In order to localize Hardy's scheme and thereby make it more computationally efficient, the indices of intersection points are considered to define the distance between them and a "hole" (= points on the initial Coons patch without projection). If no intersection point has been found for line segment $\overline{\boldsymbol{a}_{I,J}\boldsymbol{b}_{I,J}}$ only the K "closest" intersections points surrounding the "hole" are considered. A norm purely based on points' indices is used to accelerate the process of identifying the closest points. The distance $D(i_1, j_1, i_2, j_2)$ for two points with indices (I_1, J_1) and (I_2, J_2) is defined as

$$D(I_1, J_1, I_2, J_2) = |I_2 - I_1| + |J_2 - J_1| \tag{13}$$

("checker board norm," "Manhattan norm"). For each index (I, J) without intersection, the at least K closest intersection points are identified using this norm. These at least K points are effectively computed by first considering all intersection points having distance one, then distance two, then distance three, and so on, until the number K is reached or exceeded. It has been found that values for K between five and ten are sufficient.

The localized version of Hardy's reciprocal multiquadric implies the linear system

$$t_k = \sum_{i=1}^{\overline{K} \geqslant K} c_i \big(R + (u_k - u_i)^2 + (v_k - v_i)^2\big)^{-\gamma}, \quad k = 1, \ldots, \overline{K} \geqslant K, \tag{14}$$

where (u_k, v_k) and (u_i, v_i) are parameter values for which intersection points could be found.

One has to solve several linear systems if several intersections are missing. Nevertheless, since the resulting linear systems obtained by the localized approach are very small, it is still much faster than the global approach. In addition to this, the local approach has a tendency to preserve local surface properties much better than the global version.

Once one has determined parameter values $t_{I,J}$ for all line segments $\overline{\boldsymbol{a}_{I,J}\boldsymbol{b}_{I,J}}$ for which intersection points could not be found initially, the "artificial projections" are obtained by computing the linear combinations

$$\boldsymbol{z}_{I,J} = (1 - t_{I,J})\boldsymbol{a}_{I,J} + t_{I,J}\boldsymbol{b}_{I,J}. \tag{15}$$

It is differentiated between the points on the Coons patch ($\boldsymbol{x}_{I,J}$), the chosen intersection points lying in the interior of triangles ($\boldsymbol{p}_{I,J}$), the points mapped onto the parametric surfaces ($\boldsymbol{y}_{I,J}$), and points that have been computed using Hardy's reciprocal multiquadric method ($\boldsymbol{z}_{I,J}$). The union of all points $\boldsymbol{y}_{I,J}$ and $\boldsymbol{z}_{I,J}$, which forms a quasi-rectangular mesh, is used to construct a single local surface approximant.

6. Constructing a local surface approximant

In this section, the construction of a C^1 continuous bicubic B-spline surface interpolating the $(M+1) \times (N+1)$ points generated in the previous steps is discussed. A C^1

approximant is chosen in order to reduce possible oscillations. Also, C^1 continuity is sufficient for most practical purposes. The resulting B-spline surface is written as

$$s(u,v) = \sum_{j=0}^{n} \sum_{i=0}^{m} d_{i,j} N_i^4(u) N_j^4(v), \tag{16}$$

where $d_{i,j}$ is a B-spline control point and $N_i^4(u)$ and $N_j^4(v)$ are the normalized B-spline basis functions of order four. The numbers m and n are related to the resolution parameters of the Coons patch by $m = 3M$ and $n = 3N$. The (normalized) knot vectors given by values ξ_i ($\xi_i < \xi_{i+1}$) and η_j ($\eta_j < \eta_{j+1}$) have quadruple knots at both ends and triple knots in the interior and are written as

$$\begin{aligned} u &= (u_0, u_1, \ldots, u_{m+4}) \\ &= (0,0,0,0,\xi_1,\xi_1,\xi_1,\ldots,\xi_{M-1},\xi_{M-1},\xi_{M-1},1,1,1,1) \quad \text{and} \\ v &= (v_0, v_1, \ldots, v_{n+4}) \\ &= (0,0,0,0,\eta_1,\eta_1,\eta_1,\ldots,\eta_{N-1},\eta_{N-1},\eta_{N-1},1,1,1,1). \end{aligned} \tag{17}$$

The domain of the B-spline surface is $[u_3, u_{m+1}] \times [v_3, v_{n+1}]$.

Obviously, the B-spline representation degenerates to the piecewise bicubic Bézier representation. The indexing scheme used for control points and knots follows (Bartels et al., 1987). The current implementation uses uniform knot spacing, i.e., $\xi_i = i/M$ and $\eta_j = j/N$. The reason for this becomes obvious when all local surface approximants are connected to form an overall G^0 (or G^1) continuous approximant. For more information about B-splines see (Bartels et al., 1987; Farin, 1993).

Since the resulting approximant degenerates to a C^1 continuous, piecewise bicubic Bézier surface, the control points $d_{i,j}$ are derived in a straightforward manner using a C^1 cubic interpolation scheme for all rows and columns of points to be interpolated. Applying C^1 continuity conditions at all patch corners yields the four interior control points for each bicubic patch. The interior control points for all "row" and "column" curves are obtained by considering central differences of the points to be interpolated (scaled by the Euclidean distances between those points). For more detail regarding C^1 curve interpolation techniques it is referred to (Farin, 1993).

Once the control points for interpolating all rows and columns of points have been determined, the remaining four interior control points of each patch are computed such that C^1 continuity is guaranteed. Obviously, the control point $d_{3i,3j}$ is the corner of (generally) four patches. Having performed the curve interpolation step, the four surrounding points $d_{3i-1,3j}$, $d_{3i+1,3j}$, $d_{3i,3j-1}$, and $d_{3i,3j+1}$ are known as well. These control points alone are used to compute four interior control points of the (at most) four patches sharing $d_{3i,3j}$. These four interior control points are

$$\begin{aligned} d_{3i\mp1,3j+1} &= d_{3i\mp1,3j} + \left(d_{3i,3j+1} - d_{3i,3j}\right) \quad \text{and} \\ d_{3i\mp1,3j-1} &= d_{3i\mp1,3j} + \left(d_{3i,3j-1} - d_{3i,3j}\right). \end{aligned} \tag{18}$$

The coplanarity condition enforced by these equations and the uniform knot spacing guarantee C^1 continuity among all patches of the entire B-spline surface.

7. Error computation

Roughly speaking the error between a locally approximating B-spline surface $s(u,v)$ and the given geometry is the maximum of the minimum distances between points on the approximant and original geometry. The possibility of holes and overlaps in the original geometry makes a precise definition difficult. In the implementation, the exact maximum distance is approximated by a discrete error measure. The points on the approximant $s(u,v)$ used for the error computation are:

$$\begin{aligned} e_{I,J} &= s\big((0.5(\xi_I+\xi_{I+1}),\eta_J)\big), \quad I=0,\ldots,(M-1),\ J=0,\ldots,N, \\ f_{I,J} &= s\big((\xi_I, 0.5(\eta_J+\eta_{J+1}))\big), \quad I=0,\ldots,M,\ J=0,\ldots,(N-1), \\ g_{I,J} &= s\big((0.5(\xi_I+\xi_{I+1}), 0.5(\eta_J+\eta_{J+1}))\big), \\ &\quad I=0,\ldots,(M-1),\ J=0,\ldots,(N-1). \end{aligned} \tag{19}$$

These points are chosen for a simple reason: The approximant most likely has its greatest deviation from the original geometry in the interior of the bicubic B-spline patches and the interior of isoparametric curves of constant u- and v-values due to the oscillation properties of cubics.

The shortest (perpendicular) distance between the points on the approximant and the original surface(s) are used as distance measure. The implied bivariate minimization problem to find the closest point(s) is solved using Newton's method. In general, the shortest distance is not computed for all points $e_{I,J}$, $f_{I,J}$, and $g_{I,J}$ on the approximant. The reason for this is the fact that the points interpolated by $s(u,v)$ are partially points obtained by projecting points of the Coons patch onto the original surfaces and partially points obtained by Hardy's reciprocal multiquadric method. Points on $s(u,v)$ corresponding to a "hole" in the original geometry must not be considered in the error computation.

In particular, the shortest distance for a point $e_{I,J}$ is computed if both $s(\xi_I,\eta_J)$ and $s(\xi_{I+1},\eta_J)$ are points on original surfaces. The shortest distance for a point $f_{I,J}$ is computed if both $s(\xi_I,\eta_J)$ and $s(\xi_I,\eta_{J+1})$ are points on original surfaces. The shortest distance for a point $g_{I,J}$ is computed if all four points $s(\xi_I,\eta_J)$, $s(\xi_{I+1},\eta_J)$, $s(\xi_{I+1},\eta_{J+1})$, and $s(\xi_I,\eta_{J+1})$ are points on original surfaces. The maximum value among all shortest distances computed for a single surface $s(u,v)$ is used as the error.

If the resulting error is too large, then the resolution is increased and the process is started over again from the chosen boundary curves. Since there is no guarantee that this process will converge, it is terminated after a fixed number of iterations and the user is notified.

8. Connecting all local approximants

The user repeatedly specifies the four boundary curves for each local surface approximant. Topologically, all B-spline surfaces constructed are four-sided entities, which can have at most four neighbors. Two B-spline surfaces are neighbors when sharing a common boundary curve. Bifurcations (more than two surfaces sharing the same boundary

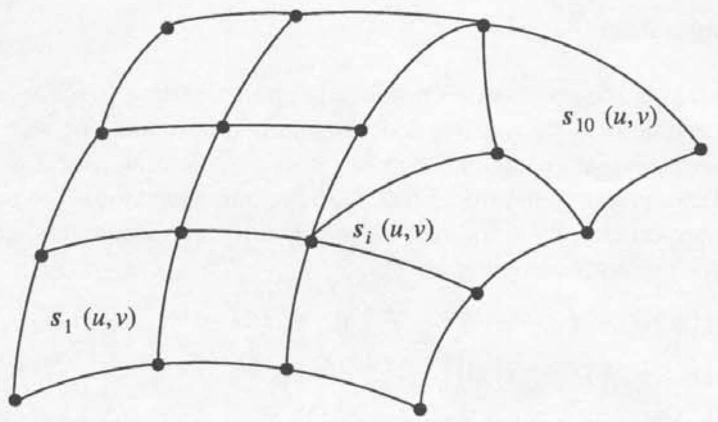

Fig. 4. Possible final B-spline surface topologies.

curve) in the set of all B-spline surfaces are not allowed. Care must be taken so that the resulting geometries are full-face interface geometries. Examples of possible surface topologies are illustrated in Fig. 4. In particular, the B-spline surfaces used in the overall approximation must satisfy these topological conditions:

- Each boundary curve is shared by at most two surfaces.
- A corner point of a surface can be common to any number of surfaces.
- If a corner point of a surface is shared by a second surface then this point is also a corner point of the second surface (full-face interface property).

A connectivity table is determined for the set of B-spline surfaces storing the (up to) four neighbors of each surface. Two surfaces are identified as neighbors if a distance estimate between pairs of boundary curves of two surfaces is smaller than some predefined tolerance. Again, the implied minimization problem needed for finding the distance between two curves utilizes Newton's method. The connectivity table is used to enforce the continuity conditions along shared boundary curves and at common corner points. Generally, it is not possible to "combine" original surface patches with the B-spline surfaces constructed, unless such original surface patches are also cubic B-spline surfaces with the same number of control points and the same knot vectors.

All B-spline surfaces used in the final approximation must be compatible, i.e., they must be bicubic B-spline surfaces, must have the same number of control points, and must have the same knots along common boundary curves. For an arbitrary configuration, this implies that one must use *one* global number of control points in both parameter directions, *one* global knot vector used in both parameter directions, and, of course, *one* global order. This is a weakness inherent to most approximation schemes based on tensor product surfaces and direct control point manipulation.

Once all local surface approximants are constructed, they are "connected" along shared boundary curves and at common corner points. It is assumed that the connectivity information among all B-spline surfaces is known (which boundary curves and common points are to be shared by surfaces). In order to form an overall G^0 (or G^1) continuous surface, it is necessary that all cubic B-spline surfaces have the same number of control points and the same number of (and values for) knots.

Let the global control point resolution for all surfaces be $(L+1) \times (L+1)$. This resolution is chosen as the maximum resolution of all resolutions (considering both parameter directions) of all B-spline surfaces. The local approximation procedure (re-using initial boundary curves) must be repeated for all those B-spline surfaces that do not have this global control point resolution.

In the current implementation, G^0 continuity is enforced among all B-spline surfaces. It is assumed that the boundary curves of all B-spline surfaces have been placed at exactly those locations where only G^0 continuity is desired (e.g., a wing tip). The enforcement of G^0 continuity is straightforward. It must be ensured that the B-spline control points along a shared boundary curve of two B-spline surfaces coincide. This is simply achieved by averaging corresponding pairs of control points along the shared boundary curve.

Enforcing tangent plane continuity conditions is slightly more complicated for an arbitrary topology of B-spline surfaces. Continuity conditions must be enforced along shared boundary curves and around shared corner points of B-spline surfaces (similar to the construction of the single C^1 B-spline surfaces themselves, see Section 6). At a shared corner point of a B-spline surface, three, four, five, or more B-spline surfaces can come together. They all must have the same tangent plane at a shared corner.

The conditions that must be satisfied in order to obtain an overall tangent plane continuous approximation are stated in (Faux and Pratt, 1979, p. 217, referred to as "gradient continuity"). The conditions are simple coplanarity conditions for certain B-spline control points along shared boundary curves and around shared corner points. The main problem is to achieve tangent plane continuity at a corner shared by multiple B-spline surfaces.

Let the shared corner point of multiple B-spline surfaces be $\boldsymbol{c}$. This point will be preserved. The B-spline surface boundary curves emanating from $\boldsymbol{c}$ are denoted by $\boldsymbol{c}_i$. The first common control point of all these curves is $\boldsymbol{c}$, while the second control point for curve $\boldsymbol{c}_i$ is denoted by $\boldsymbol{d}_i$. A common tangent plane is ensured at $\boldsymbol{c}$ if all points in the set $\{\boldsymbol{c}\} \cup \{\boldsymbol{d}_i\}$ lie in the same plane.

The normal of the tangent plane at $\boldsymbol{c}$ is constructed by averaging the normal vectors of the different B-spline surfaces at $\boldsymbol{c}$. The control points $\boldsymbol{d}_i$ are then projected onto this plane. Thus, tangent plane continuity is guaranteed at all shared common corner points of all B-spline surfaces. By enforcing collinearity for control point triples along shared boundary curves for each pair of neighbor B-spline surfaces overall tangent plane continuity is obtained.

9. Examples

Figs. 5 through 8 show single B-spline surfaces approximating different geometries containing "errors." The examples clearly demonstrate the ability of the approximation method to handle "gaps"/"holes," "overlaps," and intersections. The approximating surfaces are lying partially "above" and partially "below" the original surfaces. The approximating B-spline surfaces are obtained by specifying combinations of points and curves on the original geometries. Figs. 5 and 6 show the line segments used to obtain sample points. These line segments are perpendicular to the Coons patches defined by points and curves lying on the original surfaces.

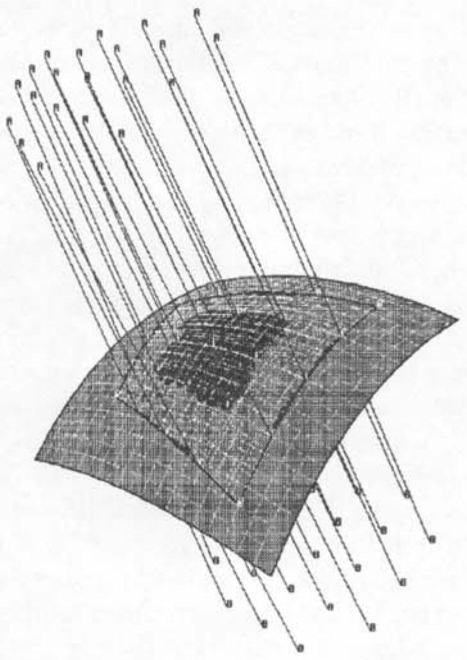
Fig. 5. Approximation of part of single patch.

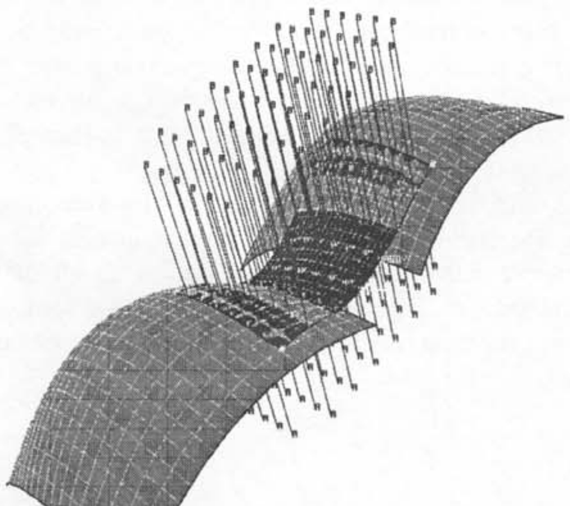
Fig. 6. Approximation of surfaces with "gap"/"hole."

Table 1 shows the resulting errors (see Section 7) when approximating the same parts of the geometries shown in Figs. 5 through 8 using different sampling resolutions. All these geometries are residing approximately in a unit cube.

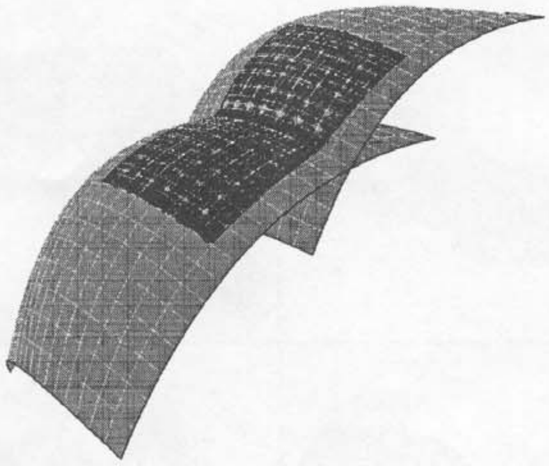

Fig. 7. Approximation of intersecting surfaces.

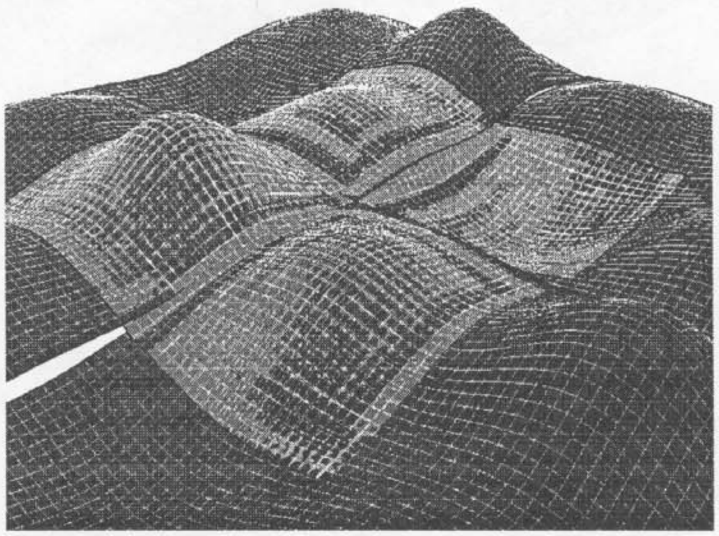

Fig. 8. Approximation of highly oscillating surfaces.

Table 1
Errors ($\times 10^{-3}$) for geometries shown in Figs. 5–9 using different sampling resolutions.

Figure \ Sampling resolution	4×4	6×6	8×8	10×10	12×12
Fig. 5	0.377	0.142	0.074	0.045	0.119
Fig. 6	35.262	26.344	26.534	39.807	31.183
Fig. 7	8.890	3.503	3.076	2.333	3.011
Fig. 8	14.515	6.466	3.434	3.935	3.526

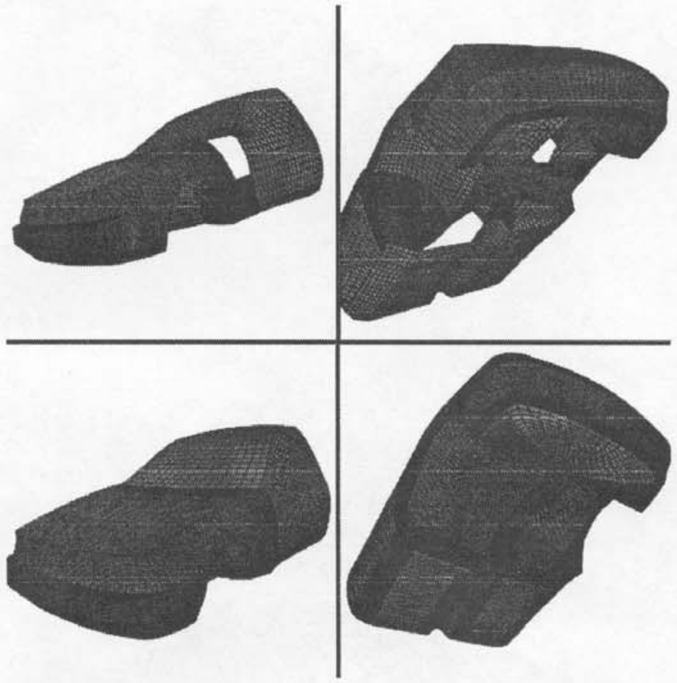

Fig. 9. Approximation of car body.

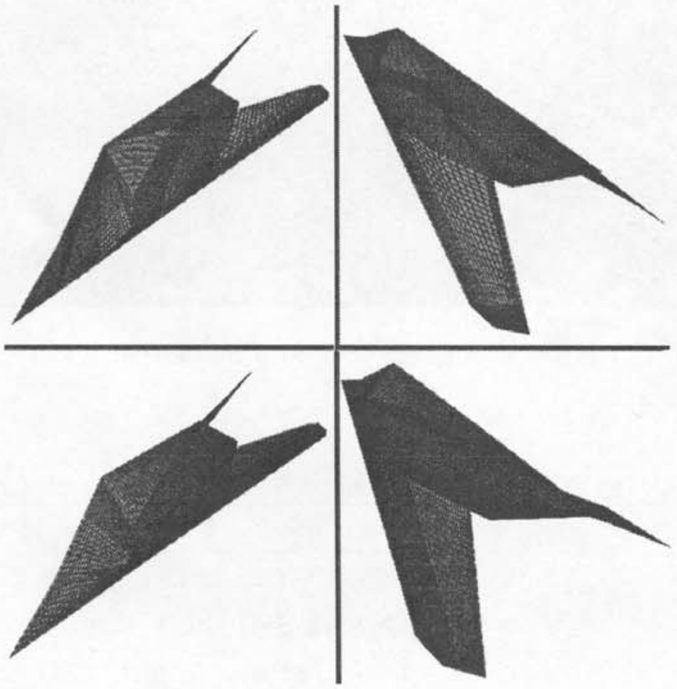

Fig. 10. Approximation of aircraft.

Figs. 9 and 10 show simple geometries and entire approximations of them (car body and aircraft configuration). Both figures show the original surfaces (top) and the resulting approximations consisting of multiple B-spline surfaces (bottom).

10. Conclusion

The interactive technique presented allows the approximation of 3D surface geometries while preserving original boundary curves of given surfaces. The method can be used to correct geometries containing errors such as "gaps"/"holes," "overlaps," and intersections. Each local B-spline surface approximant is constructed from a set of four user-specified curves. These curves can be existing boundary curves of the given geometry, parts thereof, curves in the interior of original patches, or even curves "above" and "below" the given geometry. The final approximation is a set of bicubic B-spline surfaces determining an overall G^0 surface.

It will be explored in the future whether it is possible to reduce the required user input even further, i.e., it will be investigated whether local approximants can be constructed automatically (without requiring the user to specify boundary curves). When creating the single approximants G^1 continuity should be guaranteed automatically by using the same positional/tangent plane information along boundary curves. This would eliminate the task of enforcing continuity conditions among all approximants as an additional step. The algorithm currently used for projecting a Coons patch onto the original surfaces (see Section 4) is computationally too expensive. Applying an adaptive subdivision scheme to the original surfaces based on an octree data structure may improve the performance and will be investigated.

In the system, all resulting B-spline surfaces are stored as NURBS surfaces with all control point weights set to one. Allowing variable weights might be used in the approximation procedure in order to derive surfaces with smaller numbers of control points. Unfortunately, the proper selection of weights for rational approximation is still an unsolved problem in parametric curve and surface approximation.

Acknowledgements

Special thanks go to everyone who was involved in implementing and testing the technique, in particular to Ming-Laing Chen, Richard P. Felder, J. Adam Gaither, Shekhar Mahadevan, Kelly L. Parmley, Michael G. Remotigue, Po-Yu Tsai, and John B. Whitmire. John F. Dannenhoffer III, Gerald Farin, C. Wayne Mastin, Bharat K. Soni, Joe F. Thompson, and Nigel P. Weatherill contributed significantly to this work with their experience and advice.

References

Bartels, R.H., Beatty, J.C. and Barsky, B.A. (1987), *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann Publishers, Inc., Los Altos, CA.

Farin, G. (1993), *Curves and Surfaces for Computer Aided Geometric Design*, Third Edition, Academic Press, San Diego, CA.

Faux, I.D. and Pratt, M.J. (1979), *Computational Geometry for Design and Manufacture*, Ellis Horwood Publishers, Ltd., New York, NY.

Franke, R. (1982), Scattered data interpolation: tests of some methods, Math. Comp. 38, 181–200.

George, P.L. (1991), *Automatic Mesh Generation*, Wiley, New York, NY.

Hosaka, M. (1992), *Modeling of Curves and Surfaces in CAD/CAM*, Springer-Verlag, New York, NY.

Hoschek, J. and Lasser, D. (1993), *Fundamentals of Computer Aided Geometric Design*, A.K. Peters, Ltd., Wellesley, MA.

Mortensen, M.E. (1985), *Geometric Modeling*, Wiley & Sons, New York, NY.

Piegl, L.A. (1991), Rational B-spline curves and surfaces for CAD and graphics, in: D.F. Rogers and R.A. Earnshaw, eds., *State of the Art in Computer Graphics*, Springer, New York, NY, 225–269.

Thompson, J.F., Warsi, Z.U.A. and Mastin, C.W. (1985), *Numerical Grid Generation*, North-Holland, New York, NY.

Yamaguchi, F. (1988), *Curves and Surfaces in Computer Aided Geometric Design*, Springer, New York, NY.