Approximate Rational Parameterization of Implicitly Defined Surfaces*

Elmar Wurm[1], Bert Jüttler[1], and Myung-Soo Kim[2]

[1]Johannes Kepler University Linz
Institute of Applied Geometry
email: {elmar.wurm|bert.juettler}@jku.at,
homepage: http://www.ag.jku.at/

[2]Seoul National University
Department of Computer Science
email: mskim@cse.snu.ac.kr,
homepage: http://3map.snu.ac.kr

Abstract. We present a method for approximate rational parameterization of algebraic surfaces of arbitrary degree and genus (or more general implicitly defined surfaces), based on numerical optimization techniques. The method computes patches of maximal size on these surfaces subject to certain quality constraints. It can be used to generate local low degree approximations and rational approximations of non-parameterisable surfaces.

1 Introduction

In geometric modelling and computer aided design, various different representations for curves and surfaces exist, such as implicitly defined curves and surfaces, parametric representations by (piecewise) rational functions, procedurally defined surfaces, or triangular meshes. The duality of implicit and parametric representations makes each of them especially well suited for certain applications, cf. [3].

Parametric descriptions are suitable for fast generation of point meshes, fast visualization and interactive modeling. On the other hand, the use of implicitly defined surfaces provides simple criteria to decide whether points are located on, inside or outside a surface. These surfaces support simple techniques to define blend surfaces between objects, and they can easily be intersected with lines. Moreover the class of algebraic surfaces is closed under geometric operations such as intersection and offsetting (although this is a more theoretical advantage, since the resulting degrees are rather high).

Most computational applications yield optimal performance for one particular representation. Regardless, there exist some areas where it is crucial that both

* This work was supported by the Austrian Science Fund (FWF) through subproject 15 of the Special Research Area SFB F013 "Numerical and Symbolic Scientific Computing".

descriptions are available. An example is surface-surface intersection. Ideally, one of the surfaces should be given in implicit form, and the other in parametric form. In the case of the detection of self–intersections, both representations of the same surface should be available.

This paper is devoted to the problem of converting an algebraic surface (or, more generally, an implicitly defined surface) to a (rational) parametric representation, which we shortly refer to as parameterization.

Several exact methods based on algebraic techniques are known. Most of them are constrained to special curves and surfaces (e.g., of low degree) [1, 2, 5, 12, 18]. Algorithms for solving the general parameterization problem are available [17].

Clearly, the algebraic techniques can be used only if an exact rational parameterization exists. In the surface case, both the arithmetic genus and the second plurigenus have to be equal to 0.

Alternatively, one may use approximate methods, which should be able to generate patches on any input surface. Also, we expect them to be computationally less expensive than exact methods.

In [4], a combination of algebraic and numerical techniques is used to construct G^1 spline approximations of algebraic surfaces. The algorithm starts with the computation of the singular points and curves. Later, Padé approximation and Taylor expansion are used to generate an approximation. The resulting surface maintains differential properties of the input surface and preserves the singularities.

The numerical parameterization method investigated in [8] uses the so called normal-form of a curve/surface. The output is a procedurally defined parameterization, i.e., an algorithm that maps a parameter (pair) to a point on the curve or surface $\mathcal{C}$. First a parametric patch relatively close to $\mathcal{C}$ is generated and then a parameter (pair) can be mapped to the according footpoint on $\mathcal{C}$. Note that $\mathcal{C}$ needs to be free of singularities in the area of interest.

In the remainder of this paper we present a numerical method for generating an approximate rational parameterization of an algebraic surface. We combine nonlinear minimization techniques with a region growing approach, in order to obtain good initial solutions for the nonlinear minimization.

The paper is organized as follows. Section 2 describes the objective function. Its main ingredient is a distance functional, measuring the deviation of the rational surface patch from the given algebraic surface. Section 3 discusses the actual minimization procedure and the region growing process. Starting with a small initial patch we alternate minimization and extrapolation steps to obtain an approximation of maximal size subject to certain quality criteria. Various examples are described in section 4. Finally we conclude this paper.

2 Rational parameterization as nonlinear optimization

A parameterization of a given surface is generated by computing a (possibly local) minimizer of an objective function of the form

$$S = I + \omega_J J + \omega_L L + \omega_R R + \omega_E E \tag{1}$$

among all rational surface patches of a given degree. The next section describes the space of rational patches, while the different contributions to the objective function will be explained in subsections 2.2–2.5.

2.1 Preliminaries

Consider an algebraic surface $\mathcal{F}$ of degree d. It consists of all points satisfying $F(x, y, z) = 0$, where F is a polynomial of total degree d in x, y and z with coefficients g_{ijk},

$$F(x,y,z) = \sum_{i=0}^{d} \sum_{j=0}^{d-i} \sum_{k=0}^{d-i-j} x^i y^j z^k g_{ijk}. \tag{2}$$

For reasons of numerical stability, F should be represented in Bernstein–Bézier form. The techniques described below can be applied to any implicitly defined surface, provided that the function F is C^2.

We generate a rational surface patch $\mathcal{P}$ which approximates $\mathcal{F}$. It is represented as

$$\mathcal{P} : \; p(u,v) = \left(\frac{x(u,v)}{w(u,v)}, \frac{y(u,v)}{w(u,v)}, \frac{z(u,v)}{w(u,v)} \right)^T \qquad (u,v) \in [0,1] \times [0,1] \, .$$

The three numerators $x(u, v)$, $y(u, v)$, $z(u, v)$ and the common denominator $w(u, v)$ are tensor–product polynomials of degree (m, n) in the parameters (u, v). Using the Bernstein polynomials $B^l_k(.)$, and homogeneous coordinates, we may represent it as a tensor-product Bézier patch p^* in $\mathbb{R}^4$ (cf. [10]),

$$\begin{aligned} \mathcal{P}^* : \; p^*(u,v) &= (x(u,v), y(u,v), z(u,v), w(u,v))^T \\ &= \sum_{i,j=0}^{m,n} B^m_i(u) B^n_j(v) \mathbf{c}_{ij} \qquad (u,v) \in [0,1] \times [0,1] \, . \end{aligned}$$

The control points $\mathbf{c}_{ij}$ consist of four coordinates c^x_{ij}, c^y_{ij}, c^z_{ij} and c^w_{ij}. Note that a parameterization $p(u, v)$ in $\mathbb{R}^3$ corresponds to a one dimensional space of parameterizations $p^*(u, v)$ in $\mathbb{R}^4$, since multiplying all control points $\mathbf{c}_{ij}$ with a constant factor changes $p^*(u, v)$, but the related parameterization $p(u, v)$ remains invariant.

2.2 Distance measure

The main objective is to approximate $\mathcal{F}$ by a patch $\mathcal{P}$. Hence, we need to measure the approximation quality, which is given by the distance of the two surfaces. As a measure for the approximation quality, we consider the integral

$$I = \int_0^1 \int_0^1 \frac{F^2(p(u,v))}{\|\nabla F(p(u,v))\|^2} \, \mathrm{d}u \, \mathrm{d}v, \tag{3}$$

whose integrand is the so-called squared "Sampson distance" [16]. I is a positive rational functional in the control points $\mathbf{c}_{ij}$. A local minimum represents a local best approximation of $\mathcal{F}$ by a patch $\mathcal{P}$.

Unfortunately, simple minimization of I is a task that is not well posed. The patch $\mathcal{P}$ is neither constrained in size nor position. Consequently, we obtain a local minimum for any patch $\mathcal{P}$ degenerating to a single point located on $\mathcal{F}$. This means (3) yields an infinite number of local minima. In order to obtain a unique solution, additional constraints have to be introduced.

2.3 Constraining the weights

As described in the previous section, multiplying all control points $\mathbf{c}_{ij}$ with a constant factor leaves $p(u,v)$ invariant. Hence, we have to introduce a normalization in the linear space of the $m \times n$ homogeneous control points.

In addition, a point $p(\tilde{u},\tilde{v})$ with vanishing denominator (weight) $w(\tilde{u},\tilde{v}) = 0$ corresponds to a point at infinity, or to a base point (if the three denominators vanish, too). Since we are only regular patches $p(u,v)$ without points at infinity, we have to satisfy the side–condition $w(u,v) \neq 0$.

Both requirements can be taken into account by introducing the auxiliary term

$$J = \int_0^1\!\!\int_0^1 (w(u,v)-1)^K \,\mathrm{d}u\,\mathrm{d}v,$$

where K is an even number K, in the objective function. (We chose $K = 8$.)

J is a non-negative functional that measures the deviation of the weight coordinates c^w_{ij} from 1. Let ω_J be a small positive weight factor. By adding $\omega_J J$ to the objective function, we obtain a patch $\mathcal{P}$ close to $\mathcal{F}$, with its weight coordinates c^w_{ij} being close to 1. This approach controls the weight coordinates of the control points. By choosing the weight ω 'sufficiently small', points at infinity (poles) can be avoided (see Section 3.3 for more information).

2.4 Controlling the inner geometry

Despite the additional term J, the minimization problem still does not have a unique solution. For instance, shrinking a patch $\mathcal{P}$ will usually decrease the values of I and J. Consequently, we have to constrain the size and shape of $\mathcal{P}$. For this purpose we use additional terms which are related to the inner geometry of the surface patch, in the sense of differential geometry [13].

Let g_{11}, g_{12} and g_{22} denote the first metric fundamental forms,

$$g_{11} = \langle p_u, p_u\rangle, \quad g_{12} = \langle p_u, p_v\rangle, \quad g_{22} = \langle p_v, p_v\rangle,$$

where $p_u = (\partial/\partial u)\,p(u,v)$ and $p_v = (\partial/\partial v)\,p(u,v)$ are the partial derivative vectors and $\langle\cdot,\cdot\rangle$ denotes the inner product. For any pair of positive constants

(l_1, l_2), the integral

$$L = \int_0^1\int_0^1 (g_{11} - l_1)^2 + (g_{22} - l_2)^2 \,\mathrm{d}u\,\mathrm{d}v$$

measures the deviation of the length of the first derivatives p_u and p_v from $\sqrt{l_1}$ and $\sqrt{l_2}$.

We choose another small positive weight ω_L and add the term $\omega_L L$ to the objective function. This leads to a more uniformly parameterized surface patch: in the limit $\omega_L \to \infty$, the parameter lines are traced with the constant speed $\sqrt{l_1}$ and $\sqrt{l_2}$.

The term L does not take the angle between the parameter lines into account. This can be achieved by introducing the term

$$R = \int_0^1\int_0^1 g_{12}^2 \,\mathrm{d}u\,\mathrm{d}v\,.$$

It penalizes the deviation of the angle between the parameter lines of $p(u, v)$ from a right angle. By adding $\omega_R R$ to the objective function (where ω_R is another non–negative constant), one obtains a patch that approximates the given implicit surface and has almost orthogonal parameter lines. More precisely, in the limit $\omega_L, \omega_R \to \infty$, the surface patch becomes an isometric embedding of a rectangle of size $\sqrt{l_1} \times \sqrt{l_2}$.

Remark 1. Another functional, which has a similar effect to L and R, can be obtained by considering the length of all tangent vectors at a point. If a linear parameterization q maps the parameter domain $[0, 1]^2$ into a rectangle with lengths $\sqrt{l_1} \times \sqrt{l_2}$, then the directional derivative vectors

$$|| \, \frac{\mathrm{d}}{\mathrm{d}t} q(u_0 + t\sqrt{l_2}\cos(\phi), v_0 + t\sqrt{l_1}\cos(\phi))\Big|_{t=0} || \tag{4}$$

at all points (u_0, v_0) are unit vectors. Hence, for a general surface p, one might consider the functional

$$\begin{aligned}&\int_0^{2\pi} (||\frac{\mathrm{d}}{\mathrm{d}t} p(u_0 + t\sqrt{l_2}\cos(\phi), v_0 + t\sqrt{l_1}\cos(\phi))||^2 - 1)^2 \mathrm{d}\phi \\ &= (2 + +\tfrac{3}{4} g_{11}^2 l_2^2 - 2g_{11} l_2 \tfrac{3}{4} g_{22}^2 l_1^2 - 2g_{22} l_1 + \tfrac{1}{2} g_{11} g_{22} l_1 l_2 + g_{12}^2 l_1 l_2)\pi.\end{aligned} \tag{5}$$

As a potential advantage, this approach gives functionals which provide certain invariance properties with respect to transformations of the parameter domain. However, this may not be so important, since the space of functions which we are using (tensor–product polynomials) does not have such invariance properties anyway. In contrast with this, the space of all polynomials of certain degree would be invariant.

2.5 Controlling the position

While the size and the inner geometry of the patch has now been constrained, its position on the given surface $\mathcal{F}$ is still variable, i.e., the patch can still "float" on the surface. We resolve this by pulling the points $p(u_i, v_i)$ of one or more parameter pairs (u_i, v_i) towards user– (or automatically) chosen positions $P_i(p_i^x, p_i^y, p_i^z)$. The sum of the squared Euclidean distances of the points $p(u_i, v_i)$ and points P_i is given by

$$E = \sum_i \| \frac{1}{w(u_i, v_i)} \begin{pmatrix} x(u_i, v_i) \\ y(u_i, v_i) \\ z(u_i, v_i) \end{pmatrix} - \begin{pmatrix} p_i^x \\ p_i^y \\ p_i^z \end{pmatrix} \|_2^2 \tag{6}$$

By adding $\omega_E E$ to the objective function, where ω_E is another non–negative constant, the points $p(u_i, v_i)$ will be tied to the points P_i on the surface. As a consequence, the position of the resulting patch is approximately determined. In the examples in section 4 we prescribe the position of the four corner points of the parametric patch.

Note that specifying more than one triple (u_i, v_i, P_i) also affects the inner geometry of the resulting patch. In this case one has to pay attention concerning the term L, i.e., the values l_1 and l_2 need to be chosen suitable to prevent possible conflicts in the constraints.

3 Finding a solution

The objective function (1) is obtained as the weighted sum of the terms described in the last section. It is a positive rational functional in the control points $\mathbf{c}_{ij}$ of $\mathcal{P}$. As a necessary criteria for a local minimum of S, the first partial derivatives have to vanish. This leads to a system of $M = 4(n+1)(m+1)$ nonlinear equations

$$\frac{\partial S}{\partial \alpha} = 0 \qquad \text{where } \alpha \in \{c_{ij}^x, c_{ij}^y, c_{ij}^z, c_{ij}^w,\}_{\substack{i=0,\ldots,m \\ j=0,\ldots,n}}. \tag{7}$$

We solve it using Newton's algorithm ([7]), which guarantees fast convergence, provided that a good initial solution is available.

Alternatively, this can be seen as sequential quadratic programming, applied to the problem $S \to \min$. In each step, the objective function is replaced with a local quadratic approximation.

3.1 Computational details

For each step of Newton's algorithm we need to solve a system of linear equations of size $M \times M$. The elements of the according matrices are the second partial derivatives of S,

$$\frac{\partial^2 S}{\partial \alpha \partial \beta} \qquad \text{where } \alpha, \beta \in \{c_{ij}^x, c_{ij}^y, c_{ij}^z, c_{ij}^w,\}_{\substack{i=0,\ldots,m \\ j=0,\ldots,n}}\ . \tag{8}$$

In order to generate this system, we need to compute $\frac{1}{2}M(M+1)$ integrals for each of the terms I, J, L, R, and E. For instance, related to I, we have to evaluate the integrals

$$\int_0^1\int_0^1 \frac{\partial^2}{\partial\alpha\partial\beta}\frac{F^2(p(u,v))}{\|\nabla F(p(u,v))\|^2}\,\mathrm{d}u\,\mathrm{d}v\ ,$$

Though possible, the exact evaluation of the integrals is quite expensive. A simple alternative is to use Gaussian quadrature ([7]). As the integrands related to I and E are rational expressions, Gaussian quadrature will yield only approximations of these integrals. The integrals related to J, L, and R will be evaluated exactly, provided that the order of the numerical quadrature is sufficiently high.

Remark 2. In order to facilitate the evaluation of integrals, one may also use polynomial alternatives to the rational integrands in I and E. According to our numerical experience, however, the rational functionals give better results.

3.2 Choice of the initial solution, extrapolation and iteration

Convergence. The convergence of any Newton–type method depends strongly on the choice of a suitable initial solution. If the initial solution is sufficiently close to the minimum, then the algorithm converges quadratically.

In our situation, we may construct a good initial solution by a geometric approach. If we start with a sufficiently small planar patch which is part of the tangent plane to F at a point, then the iteration process can be expected to converge.

Patch growing. Clearly, starting with a small planar patch we will obtain only a small resulting patch. Hence, we consider an iterative process to generate larger patches.

We start with a small patch, which has been obtained after several iterations of the Newton method. This patch is extrapolated in order to obtain a larger patch, which is then used as starting patch for a new cycle of Newton's algorithm. The extrapolation is restricted by the distance error, by the weights and by the inner geometry of the obtained bigger patch. This can be expressed by certain thresholds for the resulting value of the objective function.

The feasible values of the extrapolation parameters can be found by a simple bisection procedure.

Note that after each extrapolation step we need to reassign the values l_1, l_2 and the points P_i. The new locations of the P_i (typically representing the expected vertices) can be found by projecting the vertices of the extrapolated patch back onto the surface.

Termination criteria. As termination criteria for both Newton and extrapolation steps we use the properties of the current patch, which are expressed by the values of the various contributions to the objective function. The overall process is controlled by user defined global limits and thresholds for single steps.

3.3 Adaptation of the objective function

Automatic choice of points and lengths The quantities l_1, l_2 and the points P_i specify the position of the patch and the expected parametric speed $\sqrt{l_1}$ and $\sqrt{l_2}$ of the parameter lines. These values have to comply with the current patch in the iteration process, in order to avoid chaotic behaviour. In our implementation we choose $\sqrt{l_1}$ and $\sqrt{l_2}$ to be equal to the lengths of the current patch. The points P_i are chosen as the footpoints of the points $p(u_i, v_i)$ on $\mathcal{F}$, where p is the current patch.

Automatic adaptation of the weights The sum S and the resulting patch are affected crucial by the choice of the weights ω_J, ω_L, ω_R and ω_E. Of course, optimal values are not know a priori. Our implementation bypasses this problem by using an automatic adjustment of the weights according to the current contributions to the objective function. During the first steps of the algorithm, higher weights may be necessary in order to stabilize the algorithm, while they may later spoil the approximation quality.

Our main objective is to minimize the distance part I. The other terms are considered as secondary objectives. Let us assume that during the algorithm one of the values $\omega_J J$, $\omega_L L$, $\omega_R R$ or $\omega_E E$ is getting larger than I. This means that we spend most of the effort on minimizing that term instead of I. By lowering the according weight the focus is shifted again to the Sampson distance.

Our implementation uses initial values for the weights ω_J, ω_L, ω_R and ω_E and additional lower thresholds. A weight ω_T is reduced if $\omega_T T$ gets larger than I, and ω_T is larger than the threshold. This approach guarantees a minimal influence of each term.

4 Examples

In order to demonstrate the capabilities and possible applications of the algorithm, we have chosen four examples ranging from very simple to quite challenging.

A summary of the computation time and the performed extrapolation and Newton steps is given in table 1. Note that these examples all have been computed with the same parameters. The starting values and lower bounds for the weights are shown in table 2. The upper bound for the total error S was $1e^{-6}$. In all figures, the size of the bounding cube is 1. In all cases, the algorithm was stable and we obtained satisfying results.

Surface	d	m,n	Time (sec.)	Extrapolation steps	Newton steps
Sphere	2	2,2	1.1	14	46
Minimal Surface	12	3,3	32	10	50
Self-intersecting	8	3,3	10	18	57
Whitney Umbrella	3	3,3	14.91	18	113

Table 1. Examples: degrees, computing time, # steps

	ω_J	ω_L	ω_R	ω_E
start	100	1	1	1e-2
lower threshold	1e-1	1e-5	1e-5	1e-4

Table 2. Values of the weights

4.1 The sphere

Our first example is the approximation of a sphere by a rational biquadratic patch. Figure 1 shows the planar starting patch (left), the approximation after the first round of Newton steps (center), and the final approximation (right). Our method generates an approximating patch whose parameter lines are nearly iso-parametric and approximately orthogonal. The distribution of the parameter lines is visualized by the checkerboard pattern on the surface.

These properties can be enforced by increasing the (lower bounds of the) corresponding weights. On the other hand, the resulting patch will then stay smaller. Figure 2 displays the graphs of the total error S, the approximation error I, and the position error E with during the 46 Newton steps. The 14 extrapolation steps correspond to the small peaks. They are also marked by small plus signs on top of the three graphs.

Finally, Figure 3 shows the squared Sampson distance of the final patch with respect to the implicitly given sphere as graph of the domain $[0,1]^2$. Due to the terms controlling the inner geometry, which tend to flatten the surface, the maximal error is present at the four vertices of the patch. Note that the approximation is highly accurate, since the squared Sampson distance (which is a good approximation of the squared distance) is in the order of $1e-5$, while the radius of the sphere equals 1.

4.2 An approximate minimal surface

The second example, which is shown in Figure 4, is the approximation of a minimal surface taken from the Costa-Hoffman-Meeks surface family (see [6, 9]). Note that the upper part of the surface has been cut away, in order to get a better insight into the structure of the surface.

An exact rational parameterization cannot be found for this surface, since its topological genus is 1. Using our numerical method we can still generate finite patches approximating the surface with a high accuracy. Similar to the sphere case, the figure shows the initial solution and the final result.

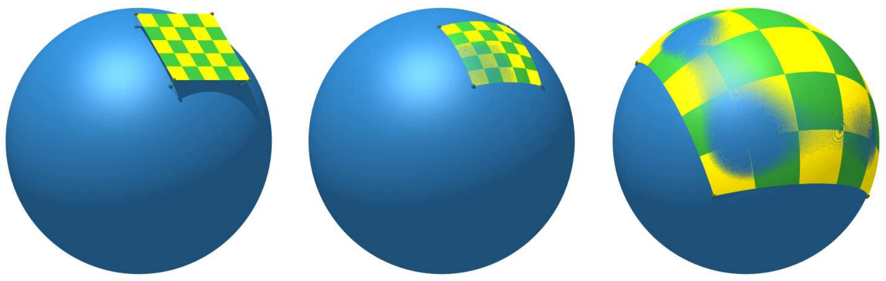

Fig. 1. Biquadratic patch approximating a sphere.

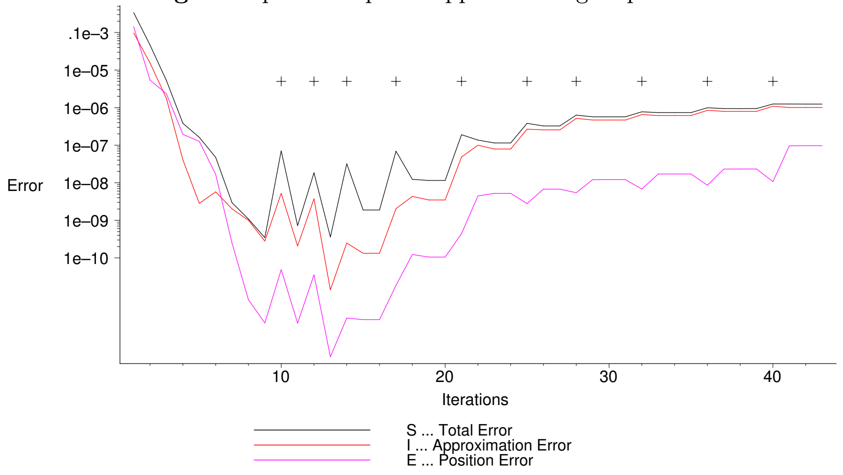

Fig. 2. Contributions to the objective function during the iteration steps.

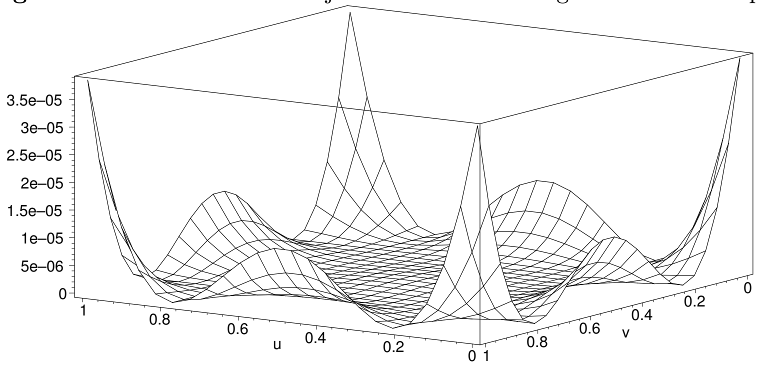

Fig. 3. The Sampson Distance of the final patch

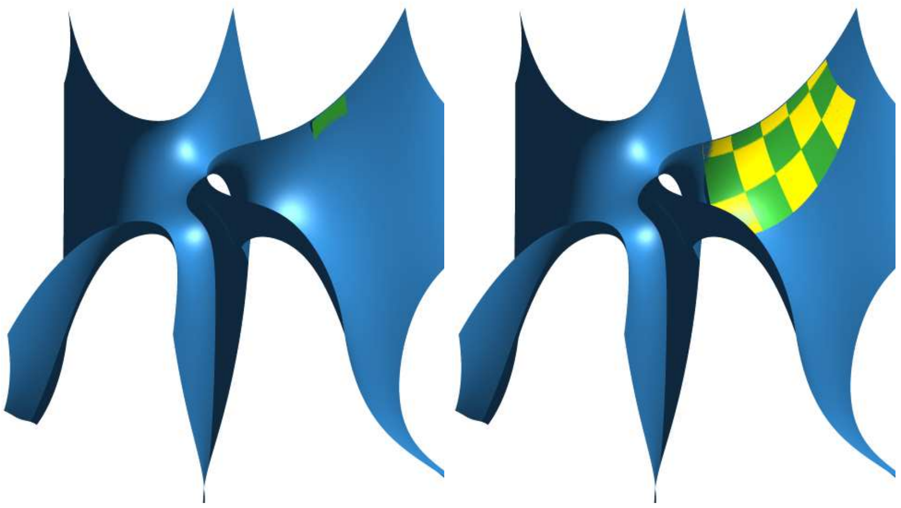

Fig. 4. Parameterization of an algebraic approximation of a minimal surface.

4.3 Surfaces with singularities

The remaining two examples (Figure 5 and 6) demonstrate that the method is able to handle self-intersections. We start with a small patch on one side of the self-intersection curve and finally get an approximation that 'dives through' the singularity and continues on the correct branch of the surface.

Once again, the figures shows the initial solution and the final result.

5 Concluding remarks

We presented a method for approximation of an implicitly defined surface of by a rational patch. The main ingredient is the minimization of the Sampson distance of the two surfaces, while additional side constraints are used to determine the inner geometry and the position of the parametric patch. The objective functional is minimized using of Newton's algorithm and Gaussian quadrature. In order to maintain a good initial solution, we alternate extrapolation steps and approximation steps, producing surface patches of optimal size, according to the specified criteria.

As a matter of future research, we will consider the problem of covering the whole implicitly defined surface. This can be achieved either by collecting several patches, which have been obtained starting from several seed points, or by parameterizing the surface not with a single patch, but with a spline surface.

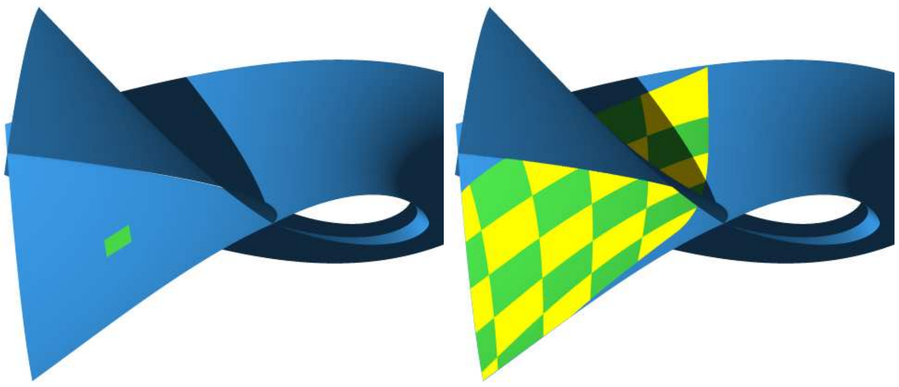

Fig. 5. Self-intersecting surface of degree 8

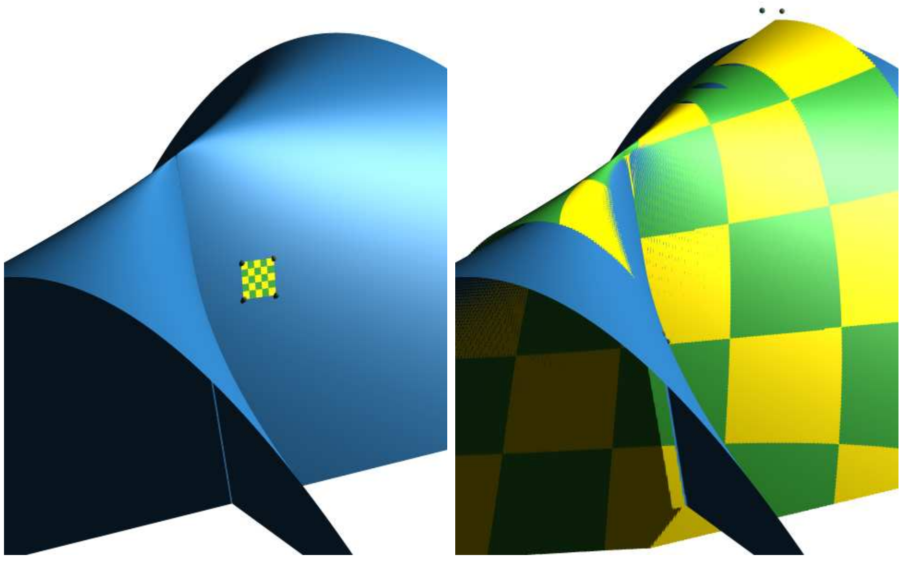

Fig. 6. Whitney Umbrella

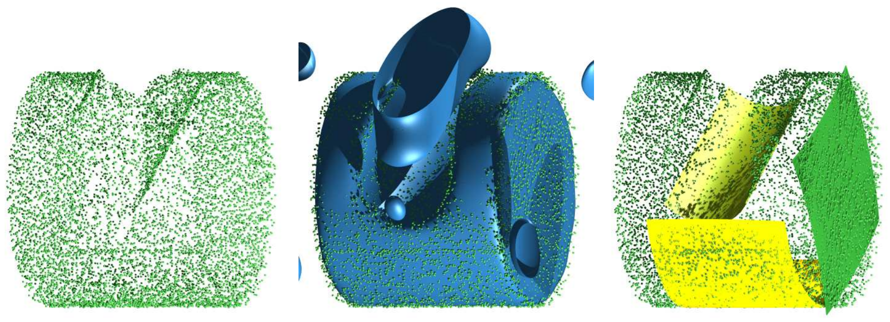

Fig. 7. Surface Reconstruction: Point cloud (left), piecewise implicit approximation (middle), parametric approximation (right)

In order to use the latter approach, the extrapolation step should be modified so as to permit adding new segments to the spline surface.

The possible applications of the parameterization technique include the construction of rational surface patches from unorganized point data. As a first step, one may fit an algebraic spline surface to these data, e.g., using techniques as in [11]. In a second step, the implicitly defined surface can then be parameterized, using the technique described in this paper.

While other methods either have to address the parameterization problem [10] or depend on an initial solution [15], the combination of implicit fitting and approximate parameterization may help to circumvent both problems. Moreover, it allows for fully exploiting the weights of the rational surface representation. This can be highly useful for generating exact descriptions of many important classes of surfaces, such as natural quadrics.

Preliminary results are shown in Figure 7. We start from a point cloud with $11,366$ points, which represents a cylinder with a cylindrical hole. The point cloud is the input data for the approximate implicitization algorithm described in [19]. The result, shown in figure 7 (middle), is a piecewise algebraic surface, which consists of 214 subpatches of tri-degree 3. One peculiar disadvantage of this implicit approximation is that it introduces additional branches.

We use the implicit approximation as input for the approximate rational parameterization algorithm described in this paper. Figure 7 (right) shows the results for three different starting patches on different sides of the object. As a byproduct of the procedure, we may identify the cylinder, the hole and the planar top. (Clearly, similar results can be obtained using existing techniques for automatic segmentation, which are often based on the analysis of the surface normals [14].) Note that the algorithm stops from growing the patches near regions of high curvature. This is due to the terms controlling the inner geometry.

References

1. Abhyankar S., Bajaj C.: Automatic Parameterization of Rational Curves and Surfaces I: Conics and Conicoids. Computer Aided Design, **19** (1987), 11-14
2. Abhyankar S., Bajaj C.: Automatic Parameterization of Rational Curves and Surfaces II: Cubics and Cubicoids. Computer Aided Design, **19** (1987), 499-502
3. Bajaj C.: The Emergence of Algebraic Curves and Surfaces, in Geometric Design – Directions in Geometric Computing (R. Martin, ed.), Information Geometers Press, 1993, 1-29.
4. Bajaj C., Xu G.: Spline Approximations of Real Algebraic Surfaces. J. Symb. Comp., Special Issue on Parametric Algebraic Curves and Applications, **23** (1997), 315-333
5. Bajaj C., Holt R.L., Netravali A.N.: Rational Parametrizations of Nonsingular Real Cubic Surfaces. ACM Trans. Graph. **17**(1): 1-31 (1998)
6. Costa A.: Examples of a Complete Minimal Immersion in of Genus One and Three Embedded Ends. Bil. Soc. Bras. Mat. **15** (1984), 47-54.
7. Hämmerlin G., Hoffmann K.H.: Numerical Mathematics, Springer, 1991,
8. Hartmann E.: Numerical parameterization of curves and surfaces, Computer Aided Geometric Design **17** (2000), 251-266.
9. Hoffman D. and Meeks W. H. III: A Complete Embedded Minimal Surfaces in with Genus One and Three Ends. J. Diff. Geom. **21** (1985), 109-127.
10. Hoschek J., Lasser D.: Fundamentals of Computer Aided Geometric Design, 1993, A. K. Peters.
11. Jüttler, B., Felis, A., Least-squares fitting of algebraic spline surfaces, Advances in Computational Mathematics **17** (2002), 135–152.
12. Jüttler B. and Rittenschober K., Using line congruences for parameterizing special algebraic surfaces. The Mathematics of Surfaces X (R. Martin and M. Bloor, eds.), pages 223-243, Berlin, 2003. Springer.
13. Kreyszig, E., Differential Geometry, Dover, New York, 1990.
14. Varadý, T., Martin, R: Reverse Engineering, in Handbook of Computer Aided Geometric Design (G. Farin, J. Hoschek, M.-S. Kim, eds.) , North-Holland, 2002, 651–681.
15. Pottmann H. and Leopoldseder S.: A concept for parametric surface fitting which avoids the parametrization problem Computer Aided Geometric Design, **20** (2003), 343-362
16. Sampson, P.: Fitting conic sections to very scattered data: an iterative refinement of the Bookstein algorithm, Computer Graphics and Image Processing **18** (1982), 97–108.
17. Schicho J.: Rational parametrization of surfaces, J. Symb. Comp., **26** (1998), 1-30.
18. Sederberg T. W., Snively J.: Parameterizing Cubic Algebraic Surfaces. The Mathematics of Surfaces II (R.R. Martin, ed.), Oxford University Press, (1987), pp. 299–320.
19. Wurm E., Jüttler B.: Approximate Implicitization via Curve Fitting, in Symposium on Geometry Processing (L. Kobbelt, P. Schröder, H. Hoppe, eds.), Eurographics / ACM Siggraph, New York 2003, 240-247.