A concept for parametric surface fitting which avoids the parametrization problem

Helmut Pottmann and Stefan Leopoldseder

Institute of Geometry, Vienna University of Technology, Wiedner Hauptstr. 8-10, A-1040 Wien, Austria [1]

Abstract

An active contour model to surface approximation is presented. It adapts to the model shape to be approximated with help of local quadratic approximants of the squared distance function. The approach completely avoids the parametrization problem. The concept is open for inclusion of smoothing operators and shape constraints.

Key words: surface approximation, surface fitting, parametrization, active contour, B-spline surface, spline conversion, degree reduction, offset surface approximation

1 Introduction

Approximating a given surface (in any representation) or an unstructured cloud of points by a B-spline surface is a widely investigated problem. The main approach uses a least squares formulation with a regularization term that expresses the fairness of the final result (see e.g. [14,19,20]).

The principle is as follows. Let $\mathbf{p}_k,\ k = 1, \dots, N$, be the given data points or samples on a given model surface. We are looking for an approximating B-spline surface or another parametric surface with a representation of the form

$$\mathbf{x}(u, v) = \sum_{i=1}^{n} B_i(u, v)\mathbf{d}_i. \tag{1}$$

[1] E-mail: pottmann@geometrie.tuwien.ac.at, stefan@geometrie.tuwien.ac.at

The basis functions $B_i(u, v)$ are usually polynomial, piecewise polynomial or piecewise rational. We assume that the functions B_i are given or precomputed from the input; thus weights or knots are already determined. Then, one estimates the surface parameters $(u_k, v_k), k = 1, \ldots, N$, of those points $\mathbf{x}(u_k, v_k)$ on the approximant which should be close to the corresponding data points $\mathbf{p}_k$. The approximant is computed as minimizer of a functional

$$F = \sum_k \|\mathbf{x}(u_k, v_k) - \mathbf{p}_k\|^2 + \lambda F_s. \tag{2}$$

The first part is a quadratic function in the unknown control points $\mathbf{d}_i$,

$$\sum_k \|\mathbf{x}(u_k, v_k) - \mathbf{p}_k\|^2 = \sum_k \left[\sum_{i=1}^{n} B_i(u_k, v_k)\mathbf{d}_i - \mathbf{p}_k\right]^2 .$$

The second part F_s in (2) is a smoothing term (see e.g. [7]). A frequently used example is the simplified thin plate energy, a quadratic function in the second partial derivatives,

$$F_s = \int\int (\mathbf{x}_{uu}^2 + 2\mathbf{x}_{uv}^2 + \mathbf{x}_{vv}^2) du dv. \tag{3}$$

It is also quadratic in the unknowns $\mathbf{d}_i$ and thus the minimization of F is the minimization of a quadratic function and amounts to the solution of a linear system of equations.

It is a difficult task to estimate the parameters (u_k, v_k). This parameter choice largely effects the result (see e.g. [28] and the references therein). Therefore, iterative parameter correction procedures have been suggested [19]. The final approximant should exhibit error vectors $\mathbf{x}(u_k, v_k) - \mathbf{p}_k$ which are orthogonal to the approximating surface $\mathbf{x}(u, v)$.

A different approach to the approximation of curves and surfaces are *active contour models*, which are mainly used in Computer Vision and Image Processing. The origin of this technique is the seminal paper by Kass et al. [24], where a variational formulation of parametric curves, coined *snakes*, is presented for detecting contours in images. There are various other applications and a variety of extensions of the snake model (see e.g. [4]).

Instead of a parametric representation of a curve, one may use an implicit form as zero set (level set) of a bivariate function. The formulation of active contour models via level sets goes back to Osher and Sethian [34]. The *level set method* [33,46] has been successfully applied to the solution of a variety of problems, e.g. for segmentation and analysis of medical images [31]. There are also several extensions to surfaces. An application to the surface fitting problem to scattered data sets has been given by Zhao et al. [52,53]. In [13] active implicit surfaces are used for visualization of discrete particles and morphing of surfaces. Other approaches to surface reconstruction via implicit surfaces define a signed distance function to the data set and denote the zero isocontour of the signed distance function as the reconstructed implicit surface [1,6]. For the fitting of algebraic tensor-product spline surfaces to scattered data see e.g. the recent paper of Jüttler and Felis [23].

An elegant formulation of curve and surface reconstruction and segmentation problems is the concept of *geodesic active contours* [8,9,43]. There, the curve to be reconstructed, e.g. from a medical image, is found as geodesic in a Riemannian space whose metric is derived from the input (image). Analogously, surface reconstruction is reformulated as minimal surface computation in a Riemannian space. The literature on this highly interesting topic is rapidly increasing. A good overview of the methods is found in the book by G. Sapiro [43].

The implicit function, whose zero set is obtained in the level set method as a representation of an evolving shape S, is in general not the signed distance function of S [2]. As an alternative method to the Hamilton-Jacobi equation, Gomes and Faugeras [17] and Zhao et al. [51] introduced a PDE formulation where the implicit representation always remains a distance function. An analogous concept has been presented by Faugeras and Gomes [15] for the evolution of objects of arbitrary dimension and codimension. The distance function preservation has several advantages from the geometric and numerical point of view [17,51].

In the present investigation, we assume as input a rather dense cloud of points or even a given surface representation. We refer to it as *model shape*. The first situation arises when we are processing data of modern 3D scanners, or if we would like to fit a surface to a dense mesh. A surface Φ as input may arise when the representation of Φ is not in the desired parametric representation. For example, we may have as input an implicit representation or a B-spline surface with a too high degree or too many knots. Thus, we are also contributing to the problems of *spline conversion*, *degree reduction* and *knot removal* [19,36]. Moreover, surfaces derived from a given surface in various ways might not be in the desired form. A well-known example are offset surfaces of NURBS surfaces, which are usually not NURBS surfaces themselves. The present approach is very well suited for *offset surface approximation* [30]. The topics of degree reduction and offset approximation within our framework of active B-spline approximation have been addressed in [39].

Our method is an active contour model in the sense that an initial shape, e.g. a B-spline surface with a regularly arranged set of control points, is iteratively deformed so that it better approximates the given model shape. This is done with help of local quadratic approximants of the squared distance function d^2

to the model shape. The local approximants help to move the active surface $\mathbf{x}(u, v)$ to lower levels of d^2, without having to specify which point $\mathbf{x}(u, v)$ should move to which point of the model shape. Thus, we are avoiding the parametrization problem.

The shrink wrapping algorithm presented by Kobbelt et. al [26] takes a similar approach to the approximation of polygonal surfaces by subdivision surfaces. A coarse base mesh is refined step by step such that the finer levels adapt to the shape of the target surface. The flow of the active subdivision surface is not guided by the distance function to the target shape, however, but a projection method is used to find point-point correspondences.

The organization of our paper is as follows. In section 2, we review our recent work on local quadratic approximants of the squared distance function d^2 to curves and surfaces [37], since it is a basic building block for the new approximation technique. In section 3, the new concept for approximation is outlined. It is illustrated at hand of examples in section 4. There, we also show that problems which are rather complicated in the traditional approach may become much simpler within the new framework. This is demonstrated with the approximation of a given surface or point cloud by a ruled surface, a problem which appears in wire electric discharge machining (EDM) and peripheral NC machining with a cylindrical cutter [10,21,27]. Finally, in section 5, we describe possible extensions and indicate directions for future research.

2 Local quadratic approximants of the squared distance function to curves and surfaces

The algorithmic concept we are proposing heavily relies on local quadratic approximants to the squared distance function of the surface Φ or point cloud to which we would like to fit a B-spline surface.

Let us first consider the distance function to a curve or surface Φ, which assigns to each point $\mathbf{p}$ of the embedding space the shortest distance of $\mathbf{p}$ to Φ. A variety of contributions deals with the computation of this function; in many cases this computation aims towards the singular set of the function, i.e., towards points where the function is not smooth since those points lie on the medial axis (or skeleton) of the input shape.

Early work on the geometry of the distance function comes from the classical geometric literature of the 19th century. One looks at its graph surface, which consists of developable surfaces of constant slope and applies results of classical differential geometry, line and sphere geometry (for a modern presentation, see e.g. [41]). Recent work on the closely related medial axis transform comes from H.I. Choi [11,12], especially the decomposition result for the efficient computation, which also appears in Kimmel et al [25]. Also in a discrete setting (pixel plane, voxel space), distance transforms and the skeleton received a lot of attention (see e.g. [32,44,45,48]).

The distance function is also the (viscosity) solution of the so-called eikonal equation. Its numerical computation is not trivial because it is a hyperbolic equation and an initially smooth front may develop singularities (shocks) as it propagates. Precisely the latter belong to the medial axis and are of particular interest. The computation of viscosity solutions with the level set method of Osher and Sethian [33,34] proved to be a very powerful approach (see e.g. [43,46,47]).

For our approach to surface approximation, not the distance function itself but the *squared* distance function is important. We are especially interested in local quadratic approximants of that function. For a derivation and proofs of the following results we refer the reader to [37]. For a better understanding, we first present local quadratic approximants to planar curves and then move to surfaces and space curves.

2.1 Local quadratic approximants of the squared distance function to a planar curve

In a Euclidean plane, we consider a C^2 curve $\mathbf{c}(t)$ with parameterization $(c_1(t), c_2(t))$. The Frenet frame at a curve point $\mathbf{c}(t)$ consists of the unit tangent vector $\mathbf{e}_1 = \dot{\mathbf{c}}/\|\dot{\mathbf{c}}\|$ and the normal vector $\mathbf{e}_2(t)$. The two vectors form a right-handed Cartesian system in the plane.

We are interested in the squared distance function d^2 which assigns to each point $\mathbf{p}$ in π the square of its shortest distance to the curve $\mathbf{c}(t)$. In the following we give the formula for a local quadratic (Taylor) approximant of the squared distance function with respect to a local Frenet coordinate system. Note that the squared distance funtion is not smooth at points of the medial axis. Thus, we will not compute local quadratic Taylor approximants for points on the medial axis.

Consider a point $\mathbf{p}$ in π whose coordinates in the Frenet frame at the normal footpoint $\mathbf{c}(t_0)$ are $(0, d)$. The curvature center $\mathbf{k}(t_0)$ at $\mathbf{c}(t_0)$ has coordinates $(0, \rho)$. Here, ρ is the inverse curvature $1/\kappa$ and thus has the same sign as the curvature, which depends on the orientation of the curve.

In the Frenet frame, the second order Taylor approximant F_d of the squared distance function d^2 at $(0, d)$ is given by

$$F_d(x_1, x_2) = \frac{d}{d-\rho} x_1^2 + x_2^2. \tag{4}$$

For a derivation of this result and a discussion of the different types of the graph surface Γ_d of F_d we refer the reader to [37].

2.2 Local quadratic approximants of the squared distance function to a surface

Consider an oriented surface $\mathbf{s}(u, v)$ with a unit normal vector field $\mathbf{n}(u, v) = \mathbf{e}_3(u, v)$. At each point $\mathbf{s}(u, v)$, we have a local right-handed Cartesian system whose first two vectors $\mathbf{e}_1, \mathbf{e}_2$ determine the principal curvature directions. The latter are not uniquely determined at an umbilical point. There, we can take any two orthogonal tangent vectors $\mathbf{e}_1, \mathbf{e}_2$. We will refer to the thereby defined frame as *principal frame* $\Sigma(u, v)$. Let κ_i be the (signed) principal curvature to the principal curvature direction $\mathbf{e}_i$, $i = 1, 2$, and let $\rho_i = 1/\kappa_i$. Then, the two principal curvature centers at the considered surface point $\mathbf{s}(u, v)$ are expressed in Σ as $\mathbf{k}_i = (0, 0, \rho_i)$. The quadratic approximant F_d to d^2 at $(0, 0, d)$ is the following.

Proposition 1 *The second order Taylor approximant of the squared distance function to a surface at a point* $\mathbf{p}$ *is expressed in the principal frame at the normal footpoint via*

$$F_d(x_1, x_2, x_3) = \frac{d}{d-\rho_1} x_1^2 + \frac{d}{d-\rho_2} x_2^2 + x_3^2. \tag{5}$$

Let us look at two important special cases.

- For $d = 0$ we obtain
$$F_d(x_1, x_2, x_3) = x_3^2.$$
This means that the second order approximant to d^2 at a surface point $\mathbf{p}$ is the same for the surface Φ and for its *tangent plane* at $\mathbf{p}$. Thus, if we are close to the surface, the squared distance function to the tangent plane at the closest point to the surface is a very good approximant. At least at first sight it is surprising that the tangent plane, which is just a first order approximant, yields a second order approximant when we are considering the squared distance function d^2, to surface and tangent plane, respectively.
- For $d = \infty$ we obtain
$$F_\infty(x_1, x_2, x_3) = x_1^2 + x_2^2 + x_3^2.$$

This is the squared distance to the footpoint on the surface.

We see that distances to normal footpoints are just good if we are in the 'far field' of the surface Φ. In the near field it is much better to use other local quadratic approximants. The simplest one is the squared distance to the tangent plane at the normal footpoint.

Remark 2 *The Taylor approximants may be indefinite. As shown in [37] we can use as appropriate nonnegative quadratic approximants*

$$F_d^+(x_1, x_2, x_3) = \frac{d}{d+\rho_1} x_1^2 + \frac{d}{d+\rho_2} x_2^2 + x_3^2, \tag{6}$$

where d, ρ_1, ρ_2 are taken as positive. Points beyond the principal curvature centers are ruled out, but they do not arise anyway when we consider global distances.

2.3 Local quadratic approximants of the squared distance function to a space curve

In case that boundary curves of surfaces are involved, it is also useful to know about the function d^2 for a space curve.

Given a point $\mathbf{p}$ in $\mathbb{R}^3$, the shortest distance to a C^2 space curve $\mathbf{c}(t)$ occurs along a normal of the curve or at a boundary point of it. The latter case is trivial and thus we exclude it. At the normal footpoint $\mathbf{c}(t_0)$ we form a Cartesian system with $\mathbf{e}_1$ as tangent vector and $\mathbf{e}_3$ in direction of the vector $\mathbf{p} - \mathbf{c}(t_0)$. This *canonical frame* can be viewed as limit case of the principal frame for surfaces, when interpreting the curve as pipe surface with vanishing radius. By this limit process, we can also show the following result.

Proposition 3 *The second order Taylor approximant of the squared distance function to a space curve $\mathbf{c}(t)$ at a point $\mathbf{p}$ is expressed in the canonical frame Σ at the normal footpoint via*

$$F_d(x_1, x_2, x_3) = \frac{d}{d-\rho_1} x_1^2 + x_2^2 + x_3^2. \tag{7}$$

Here, $(0, 0, \rho_1)$ are the coordinates (in Σ) of the intersection point of the curvature axis of $\mathbf{c}(t)$ at the footpoint $\mathbf{c}(t_0)$ with the perpendicular line $\mathbf{pc}(t_0)$ from $\mathbf{p}$ to $\mathbf{c}(t)$.

3 Approximation with an active surface in the 'squared distance field'

Our approach to surface approximation has as input a model shape M. This can be a sufficiently dense point cloud or a surface in any mathematical representation. From the model shape, we compute – for example with help of second order Taylor approximants – local quadratic approximants of the squared distance function d^2 to the model shape M. Thus, for any point $\mathbf{p} \in \mathbb{R}^3$, we have a way to compute such a local quadratic approximant $F_{d,\mathbf{p}}$. In order to apply Equ. (5) or (6) we need to compute the closest point to $\mathbf{p}$ on M, and the principal curvature directions and principal curvatures in this surface point. Since finding the foot-point of $\mathbf{p}$ on the target surface M is a rather time-consuming task, our algorithm can be speeded up considerably by precomputing the local quadratic approximants F_d. In our very recent paper [40] the space surrounding M is partitioned in an appropriate octree cell structure with smaller cells in the near field and larger cells in the far field of M. Each cell C carries a (nonnegative) quadratic function $F_{d,C}$ and for each point $\mathbf{p}$ inside C we define $F_{d,\mathbf{p}} = F_{d,C}$. The fast computation of the functions $F_{d,C}$ in [40] is based on the evaluation of the distance function of M on a hierarchical grid. Here we used a variant of Zhao's sweeping algorithm [54] which is an improvement of the Fast Marching Method introduced by Sethian [46]. Example 4 in Sec.4 was calculated with such an underlying octree cell structure. A similar data structure for adaptively sampled distance fields has been introduced by Frisken et al. [16].

In its simplest formulation, the method outlined below assumes that the local quadratic approximants $F_{d,\mathbf{p}}$ are *nonnegative* quadratic functions (cf. Remark 2). In a general coordinate system (applying a coordinate transformation on the local coordinates used in section 2) we have

$$F_{d,\mathbf{p}}(\mathbf{x}) = \mathbf{x}^T A_{\mathbf{p}} \mathbf{x} + \mathbf{b}_{\mathbf{p}}^T \mathbf{x} + c_{\mathbf{p}} \tag{8}$$

with a symmetric, positive semidefinite Matrix $A_{\mathbf{p}}$.

The *active surface model* we are using shall be of the following nature. It is governed by *control points* $\mathbf{d}_i,\ i = 1, \ldots, n$, and there is a *linear* relation which computes from the control point set a larger set of surface points $\mathbf{s}_k,\ k = 1, \ldots, N$, or points on a refined model. For example, we may have a B-spline surface of the form (1), and compute the surface points

$$\mathbf{s}_k = \mathbf{x}(u_k, v_k) = \sum_{i=1}^{n} B_i(u_k, v_k)\mathbf{d}_i,$$

for a regular grid of points (u_k, v_k) in the parameter domain. Another example is a subdivision surface: The points $\mathbf{d}_i$ can be the vertices of a coarse level and the points $\mathbf{s}_k$ can be vertices of a refined model, after application of a few steps of the subdivision rule [49]. The set of points $\mathbf{s}_k$ must be large enough to well capture the shape of the active surface. In the following we use the notation

$$\mathbf{s}_k = L_k(\mathbf{d}_1, \ldots, \mathbf{d}_n) \tag{9}$$

to express the linear computation of $\mathbf{s}_k$ from the control points $\mathbf{d}_i$.

The key idea now is to iteratively change the input control points $\mathbf{d}_i$ so that the active surface deforms towards the model shape M. We do not use the gradient flow in the squared distance field but solve in each step a minimization problem which ensures that we quickly move to lower levels of the function d^2.

The method now proceeds in the following steps.

(1) Initialize the active surface and determine the boundary conditions. This requires the computation of an initial set of control points $\mathbf{d}_i$, $i = 1, \ldots, n$, the proper treatment of boundaries (such as fixing vertices of a patch) and the avoidance of model shrinking during the following steps. More details are described after the outline of the algorithm.

(2) Repeatedly apply the following steps a.-c. until the approximation error or change in the approximation error falls below a user defined threshold. In our examples we have terminated the algorithm when in a certain iteration the improvement of the mean squared approximation error was less then 0.5%. The steps a.-c. are:

   a. With the current control points $\mathbf{d}_i$, compute, for $k = 1, \ldots, N$, the active surface point $\mathbf{s}_k = L_k(\mathbf{d}_1, \ldots, \mathbf{d}_n)$ and a local quadratic approximant $F_{d,\mathbf{s}_k} =: F_d^k$ of the squared distance function to the model shape M at the point $\mathbf{s}_k$. This has to be a nonnegative quadratic function, $F_d^k(\mathbf{x}) \geq 0, \forall \mathbf{x} \in \mathbb{R}^3$.

   b. Compute displacement vectors $\mathbf{c}_i$, $i = 1, \ldots, n$, for the control points $\mathbf{d}_i$ by minimizing the function

$$F = \sum_{k=1}^{N} F_d^k(L_k(\mathbf{d}_1 + \mathbf{c}_1, \ldots, \mathbf{d}_n + \mathbf{c}_n)) + \lambda F_s. \tag{10}$$

   Thus, our goal is that the new surface points

$$\mathbf{s}_k^* = L_k(\mathbf{d}_1 + \mathbf{c}_1, \ldots, \mathbf{d}_n + \mathbf{c}_n),$$

   which are linear combinations of the new control points $\mathbf{d}_i^* = \mathbf{d}_i + \mathbf{c}_i$, are closer to the model shape than the old model points $\mathbf{s}_k$. The functional F_s is a smoothing functional which shall be quadratic in the control points of the active surface. Thus it is a quadratic funtion in the new control points $\mathbf{d}_i + \mathbf{c}_i$, and also quadratic in the unknowns $\mathbf{c}_i$. We see that this step requires the minimization of a quadratic function F in the displacement vectors $\mathbf{c}_i$ of the control points. This amounts to the solution of a linear system of equations.

c. With $\mathbf{c}_i$ from the previous step, the updated control points are $\mathbf{d}_i + \mathbf{c}_i$.

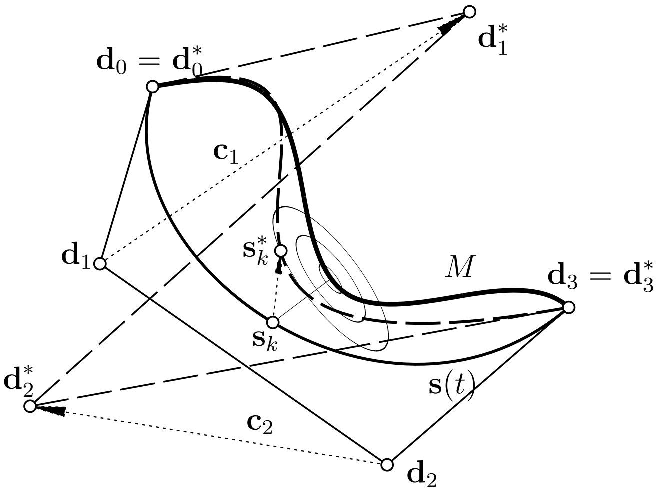

Fig. 1. One step in the curve approximation procedure. The curve M is approximated by a B-spline curve.

See Fig. 1 for an illustration of a lower-dimensional example. The model shape M is a curve that shall be approximated by a B-spline curve. Fig. 1 shows an initial position of the B-spline curve $\mathbf{s}(t)$, with control points $\mathbf{d}_i$, and the updated B-spline curve, with control points $\mathbf{d}_i^*$, after one iteration step. In particular, for one of the sample points $\mathbf{s}_k$ the local quadratic approximant F_d^k of the squared distance function is indicated by three of its level sets which are concentric ellipses.

In each iteration step we have to solve a linear system of equations for the unknown displacement vectors $\mathbf{c}_i, i = 1, \ldots, n$. For the task of B-spline surface approximation we will summarize in more detail how the linear system for the minimization problem looks like:

The model points $\mathbf{s}_k$ are certain evaluation points of the active B-spline surface, i.e.,

$$\mathbf{s}_k = \sum_i B_{i,k} \mathbf{d}_i,$$

where $B_{i,k}$ denotes the B-spline basis function B_i evaluated at (u_k, v_k). The new position $\mathbf{s}_k^*$ of $\mathbf{s}_k$ after the current iteration step will be

$$\mathbf{s}_k^* = \sum_i B_{i,k}(\mathbf{d}_i + \mathbf{c}_i) = \mathbf{s}_k + \sum_i B_{i,k} \mathbf{c}_i.$$

For each of the model points $\mathbf{s}_k$ we have a local quadratic approximant F_d^k of the squared distance function, i.e.,

$$F_d^k(\mathbf{x}) = \mathbf{x}^T A_k \mathbf{x} + \mathbf{b}_k^T \mathbf{x} + c_k,$$

with a symmetric and positive semidefinite matrix A_k.

The quadratic function F, Equ. (10), which shall be minimized in the current iteration step is (without the smoothing term λF_s)

$$\begin{aligned} F = & \sum_k (\mathbf{s}_k^{*T} A_k \mathbf{s}_k^* + \mathbf{b}_k^T \mathbf{s}_k^* + c_k) = \\ & \sum_k \Big[(\sum_i B_{i,k} \mathbf{c}_i)^T A_k (\sum_i B_{i,k} \mathbf{c}_i) + (2A_k \mathbf{s}_k + \mathbf{b}_k)^T (\sum_i B_{i,k} \mathbf{c}_i) + \\ & \quad (\mathbf{s}_k^T A_k \mathbf{s}_k + \mathbf{b}_k^T \mathbf{s}_k + c_k) \Big], \end{aligned}$$

which is quadratic in the unknown vectors $\mathbf{c}_i$.

After collecting all the variables $\mathbf{c}_i, i = 1, \ldots, n$, into the column vector $C = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n)$, we can rewrite above function as

$$F = C^T A C + \mathbf{b}^T C + c,$$

with a $3n \times 3n$-matrix A, a $3n$ column vector $\mathbf{b}$, and a scalar c. The minimization of F amounts to the solution of the linear system

$$2AC + \mathbf{b} = 0.$$

Let us now look at some details, improvements and refinements which are important for a successful implementation of the proposed method:

(1) The choice of the *initial shape* is not critical, but it must fulfil the following criteria. The initial surface must possess enough flexibility (e.g., a sufficient number of control points) to represent the model shape within the required accuracy. Furthermore, the initial shape must already exhibit the *correct topology.* A change in topology is not possible with a subdivision surface or parametric surface. It would be possible if we view the active surface as level set of a trivariate function; this is the approach taken in the level set method [46] which has been applied to surface approximation by Zhao et al. [52,53].

(2) One has to impose appropriate *boundary conditions.* For example, we may want to fix the vertices of a surface patch or want to approximate boundary curves of the model shape. One way to do this is to apply the curve analogue of the present method in a first step; then we keep corresponding

control points fixed in the surface approximation procedure. However, it can be possible to reach an overall better surface quality at some sacrifice of accuracy at the boundary. Then it is better to add the functional for boundary approximation as a penalty term to F of equation (10). We show an example in section 4.

(3) In a totally unrestricted flow, the active surface may shrink to a single point of the model surface and then in a trivial way yield the minimal approximation error zero. Strategies for *shrinking avoidance* depend on the special situation. It can be handled by appropriate boundary conditions which avoid shrinking. It can also be approached by an appropriate quadratic penalty function F_p added to F in equation (10). For a closed surface, we can start with an initial shape which lies entirely outside the model shape M. The active surface then deflates towards the model shape, if we forbid that it traverses the interior. We can define in $\mathbb{R}^3$ a trivariate potential function $P(x, y, z)$, which assumes high values in the interior of M and small values near the surface and outside M. We then compute at each control point $\mathbf{d}_k$ a local quadratic approximant Q_p^k of the penalty function P and consider the new functional

$$F^* = F + \mu \sum Q_p^k(\mathbf{d}_k + \mathbf{c}_k).$$

This will avoid that the control points enter the forbidden region. In a refined version one can use the model points $\mathbf{s}_k$ instead of the control points: With Q_p^k as quadratic approximant to the penalty function at $\mathbf{s}_k$ one considers the minimization of

$$F^* = F + \mu \sum Q_p^k(L_k(\mathbf{d}_1 + \mathbf{c}_1, \dots, \mathbf{d}_n + \mathbf{c}_n)).$$

In both cases, we add a quadratic function in the unknowns $\mathbf{c}_j$ and thus we remain within our principle of minimizing a quadratic function in each iteration step.

(4) If we have an active B-spline surface or another parametric surface and get model points $\mathbf{s}_k$ by evaluation, it is not necessary to keep the parameter values (u_k, v_k) at which we evaluate fixed. An adaptive evaluation which guarantees a nearly uniform distribution over the active surface, or emphasizes especially important regions with help of more model points, will be useful. Moreover, we can introduce further knots and thus more control points during the algorithm if the desired accuracy cannot be reached with the coarser model. One sees that the method naturally supports a multiresultion modeling strategy. For the lower-dimensional case of B-spline curve approximation there is an excellent paper of Yang et.al. [50] which addresses control point insertion and removal for our active B-spline approximation scheme.

(5) The assumption of *nonnegativity* of the approximants F_d^k of the squared distance function to M can be avoided if the change of control points (and thus model points) is restricted. This can be achieved by adding

a term to F which expresses the distance of the new control points to the old ones, e.g. $\sum \|\mathbf{c}_i\|^2$. It might, however, be difficult to make sure that the corresponding model points $\mathbf{s}_k$ do not move outside the positive region of their local quadratic approximants.

(6) The smoothing term λF_s in Equ. (10) is especially important in the first iteration steps. Without this term the active B-spline surface may develop a bad parametrization or even small loops. In later iterations — when the active surface is already close to the desired shape M — the smoothing term is less important. Thus the smoothing factor λ is decreased from iteration to iteration.

For F_s we can use any quadratic smoothing functional. It may change in each iteration step. Thus, we can also build Greiner's method for the minimization of nonlinear fairness functionals [18] into our surface approximation technique.

(7) Further improvements in the parametrization of the active surface $\mathbf{x}(u,v)$ can be achieved by adding a weighted function μF_t to F in equation (10) where

$$F_t = \int\int (\mathbf{x}_u^2 + \mathbf{x}_v^2)\ dudv.$$

This function is quadratic in the unknowns $\mathbf{c}_i$, and its minimization favors a surface $\mathbf{x}(u,v)$ whose isoparametric lines are traced with constant speed.

4 Examples

In the following we give several examples for the approximation of a given surface by a B-spline surface. An important special case is the surface approximation by *ruled* B-spline surfaces.

4.1 Approximation with B-spline surfaces

Example 1 In the first example, see Fig. 2, left, the initial position of the active B-spline surface patch (dark gray) is chosen as the bilinear patch connecting the corner vertices of the model surface (light gray). The active B-spline is of degree $(3,3)$ with 8×6 control points. The result of our algorith after only seven iterations is shown in Fig. 2, right. As a boundary condition, the four vertices of the model patch have been fixed, all the other 44 B-spline control points have three degrees of freedom. In order to approximate the boundaries of the patch as well, one can first apply our algorithm just to the boundary curves, using quadratic approximants of the squared distance function to these space curves. This determines the boundary control points of the B-spline patch. In a second step the active surface flow towards the model

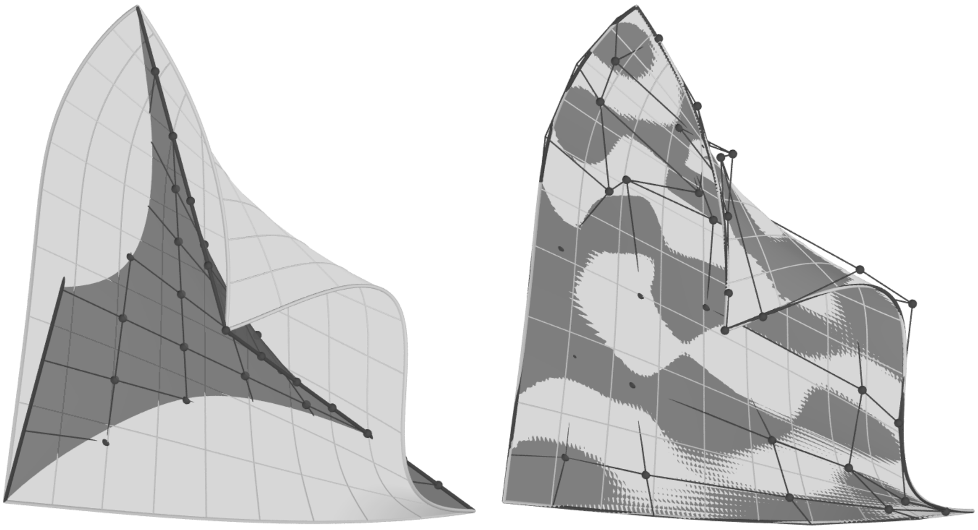

Fig. 2. Surface approximation by a B-spline patch, fixing the corner vertices. Left: initial position, Right: final position after 7 iteration steps — with boundary curve approximation.

surface is computed while keeping all the boundary control points fixed, see Fig. 2, right.

The deviations between the model surface M and the active B-spline surface are declining quickly from iteration to iteration. Several termination criteria based on average or maximum deviation can be used. In all our examples presented here we have stopped the algorithm when in a certain iteration step the improvement of the mean squared error was below 0.5%. Based on a scaling of the model surface to a bounding box of unit size 1, a mean squared distance deviation of $6.25 \cdot 10^{-9}$ and a maximum squared distance deviation of $2.25 \cdot 10^{-7}$ is reached after 7 iterations.

Example 2 In this example the model surface has a feature across the patch. Again the active B-spline surface is initialized bilinearly, see Fig. 3. The surface is of degree $(3, 3)$ and is initialized with 6×4 control points. We illustrate the two ways of boundary curve approximation mentioned in Sec. 3.

Method A. As described in Example 1 one may first approximate the boundary curves, and thus determine the boundary control points of the B-spline surface. The 8 inner control points remain unchanged (Fig. 4, left). Afterwards all the boundary control points are fixed and only the inner control points are displaced iteratively, such that the active surface flows closer to the model surface (Fig. 4, right).

Method B. It is also possible to use a weighted sum of the quadratic functionals for boundary and surface approximation, respectively. In this way the displacement vectors of all of the control points (except the fixed patch vertices) are

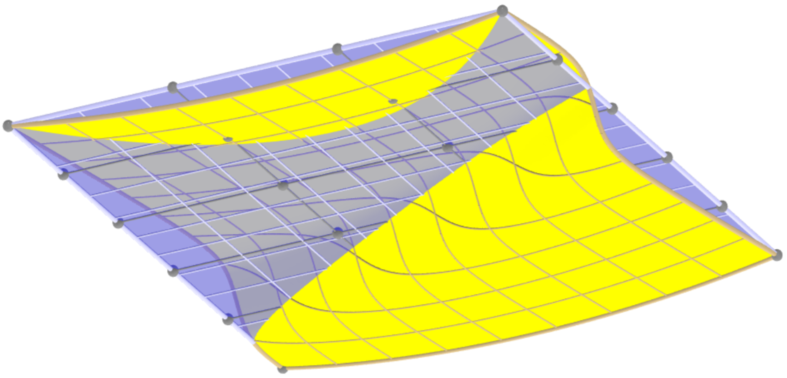

Fig. 3. Surface with feature and initial position of approximating B-spline surface.

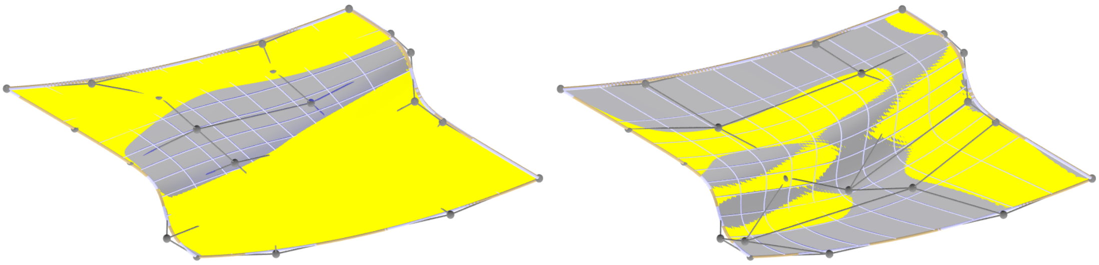

Fig. 4. Boundary curve approximation, Method A.

computed simultaneously, see Fig. 5.

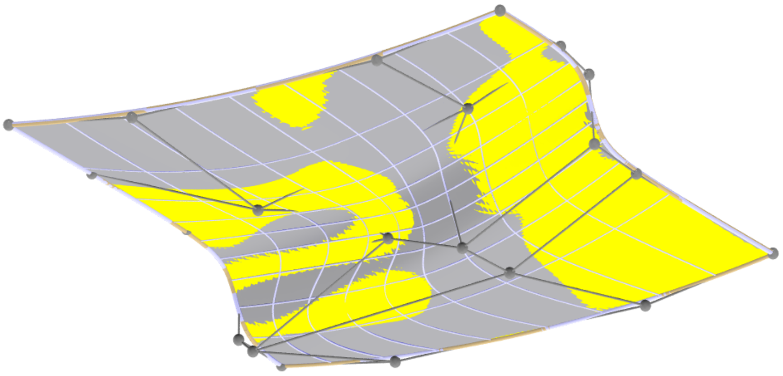

Fig. 5. Boundary curve approximation, Method B.

Both methods yield similar results. The surface parameter lines in Fig. 4 and Fig. 5 are the isoparametric lines of the active B-spline surface. The feature of the given surface turns out to be modelled approximately in isoparametric direction.

Remark The present concept of an active surface under the influence of the squared distance function of a model surface is also well suited for *offset surface approximation*, see [39]. This is a widely investigated topic, see e.g. [19,30,35] for surveys on offset surfaces.

For a point $\mathbf{p} \in \mathbb{R}^3$, we have Eq. (5) to describe a quadratic approximant of the squared distance function to a surface $\mathbf{s}(u,v)$, expressed in the principal frame $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ at the normal footpoint. Let us consider $\mathbf{s}$'s one-sided offset surface $\mathbf{s}^*(u,v) = \mathbf{s}(u,v) + \alpha \cdot \mathbf{e}_3(u,v)$ to distance α. In corresponding points $\mathbf{s}(u,v)$ and $\mathbf{s}^*(u,v)$ the principal directions $\mathbf{e}_j, j = 1,2,3$, and the two principal curvature centers are the same. With respect to the principal frame with origin in $\mathbf{s}^*(u,v)$ we have coordinates $\mathbf{p} = (0,0,d-\alpha)$ and $\mathbf{k}_i = (0,0,\rho_i - \alpha)$, cf. subsection 2.2. The second order Taylor approximant of the squared distance function to the offset surface $\mathbf{s}^*(u,v)$ at a point $\mathbf{p}$ is expressed in the principal frame at the normal footpoint via

$$F_d^*(x_1,x_2,x_3) = \frac{d-\alpha}{d-\rho_1}x_1^2 + \frac{d-\alpha}{d-\rho_2}x_2^2 + x_3^2. \tag{11}$$

With the quadratic approximant (5) of the squared distance function to a surface $\mathbf{s}(u,v)$ it is therefore simple to derive the corresponding quadratic approximant for its offset surface $\mathbf{s}^*(u,v)$ to distance α.

4.2 Approximation with ruled surfaces

Example 3 In several applications including CAD, architectural design, wire EDM and NC milling with a cylindrical cutter one has to solve the following problem: approximate a given surface (or point cloud) by a ruled surface. With traditional concepts this is a tackling problem [10,21].

We will illustrate the approach of the present paper at hand of an example surface $\mathbf{s}(u,v)$ that was already used as a test surface in [10]. Distortion terms have been applied to a ruled surface to generate this test surface $\mathbf{s}(u,v)$, see Fig. 6.

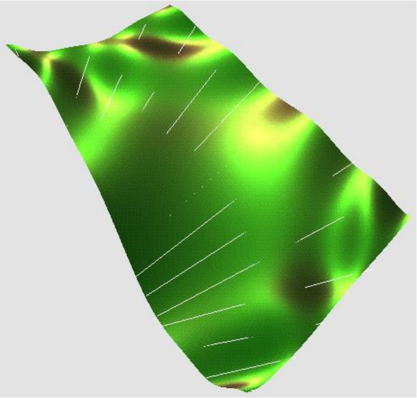

Fig. 6. Distorted ruled surface $\mathbf{s}(u,v)$.

The ruled B-spline surface which shall approximate $\mathbf{s}(u,v)$ is of degree $(n,1)$

and its $m \times 2, m > n$, control points are initially chosen in a regular planar grid, see Fig. 7, left, for $n = 3, m = 7$. The boundary rulings of the active surface (dark gray) are initially positioned at some distance to the model surface $\mathbf{s}(u, v)$ (light gray). These boundary rulings are kept fixed in the iterative surface flow. The shape of the active surface is influenced by the squared distance function to $\mathbf{s}(u, v)$ and by the smoothing term F_s. The final position of the approximating ruled surface is given in Fig. 7, right. Although no special care has been taken in choosing the fixed boundary rulings appropriately, the generated ruled surface fits well in the trimmed region of the model surface, see the close-up in Fig. 8.

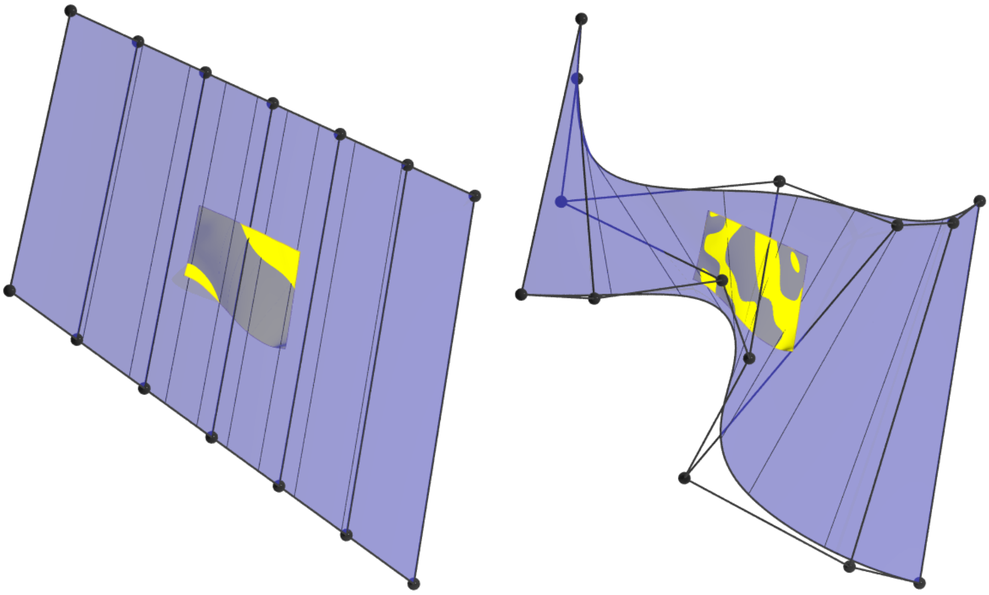

Fig. 7. Surface approximation with ruled surface. Left: initial position of ruled surface (dark gray) and model surface $\mathbf{s}(u, v)$ (light gray). Right: final position of ruled surface after 5 iterations.

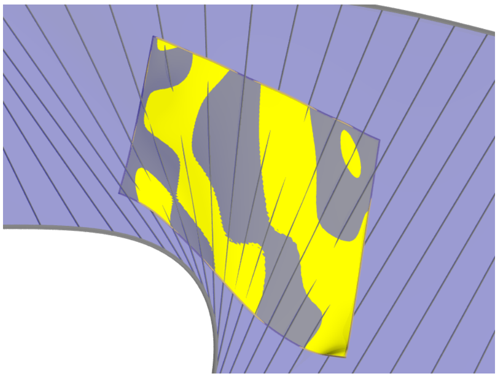

Fig. 8. Surface approximation of model surface (light gray) with ruled surface (dark gray). Detail of Fig. 7, right.

Note that during the iterative surface flow not all active surface points are in the influence region of the model surface $\mathbf{s}(u,v)$: If the closest model point to an active surface point lies on the boundary of $\mathbf{s}(u,v)$, this active surface point is classified as outlier and is discarded. This outlier detection has to be repeated in each iteration step and may be speeded up by threshholding techniques.

4.3 Approximation of a triangulated point cloud

Example 4

In the last example the model surface M is a triangulated point cloud, see Fig. 9, left. The data has been obtained by scanning an architectural design, namely a clay model of a tower. In the lower left part of the model you may note a small gap in the model. This is a local artifact where the laser based data capturing failed. Topologically the surface M is a cylindrical patch. This surface shall be approximated by a B-spline surface which is closed in u-parameter direction, see Fig. 9, right. The initial position of the active B-

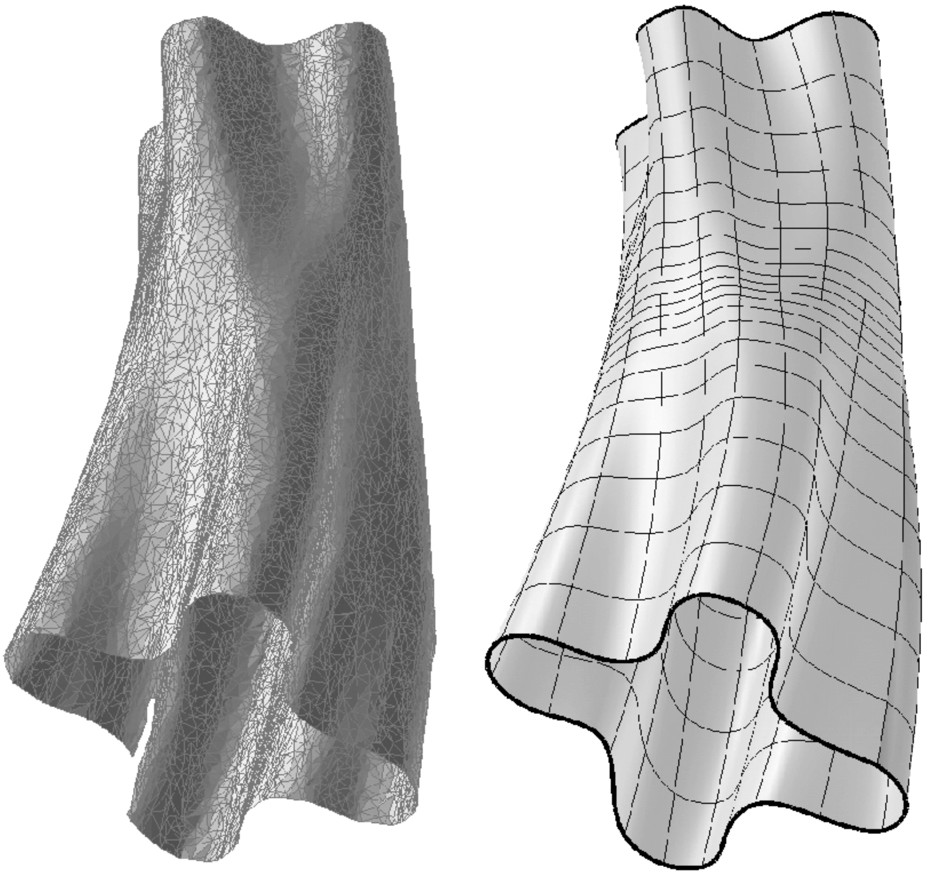

Fig. 9. Approximation of a model surface (triangulated point cloud, left) with a closed tensor product B-spline surface (right).

spline surface was chosen as in Fig. 10, left. The B-spline surface is bicubic, i.e., degree $(3,3)$, with 24×6 control points.

As a boundary condition, two planes ε_0 and ε_1 have been chosen where the closed boundary curves $v = v_0$ and $v = v_1$ of the active surface patch are lying in. During the iterative surface approximation procedure, the control points corresponding to these two boundary curves are only allowed to move within ε_0 and ε_1, respectively. This side condition is linear in the unknown displacement vectors of the control points. All other control points have their full three degrees of freedom. The number of sample points $\mathbf{s}_k$ on the active B-spline surface was increased from iteration to iteration, beginning with 500 sample points and ending with 3000.

Although the model surface and its squared distance function are rather complex, the approximation algorithm converges after only 20 iterations. Fig. 10, left, shows the final result as an overlay of Fig. 9, left and right. The deviations between the triangulated model and the active B-spline surface are given in Fig. 11 for each iteration step. The upper diagram shows the mean

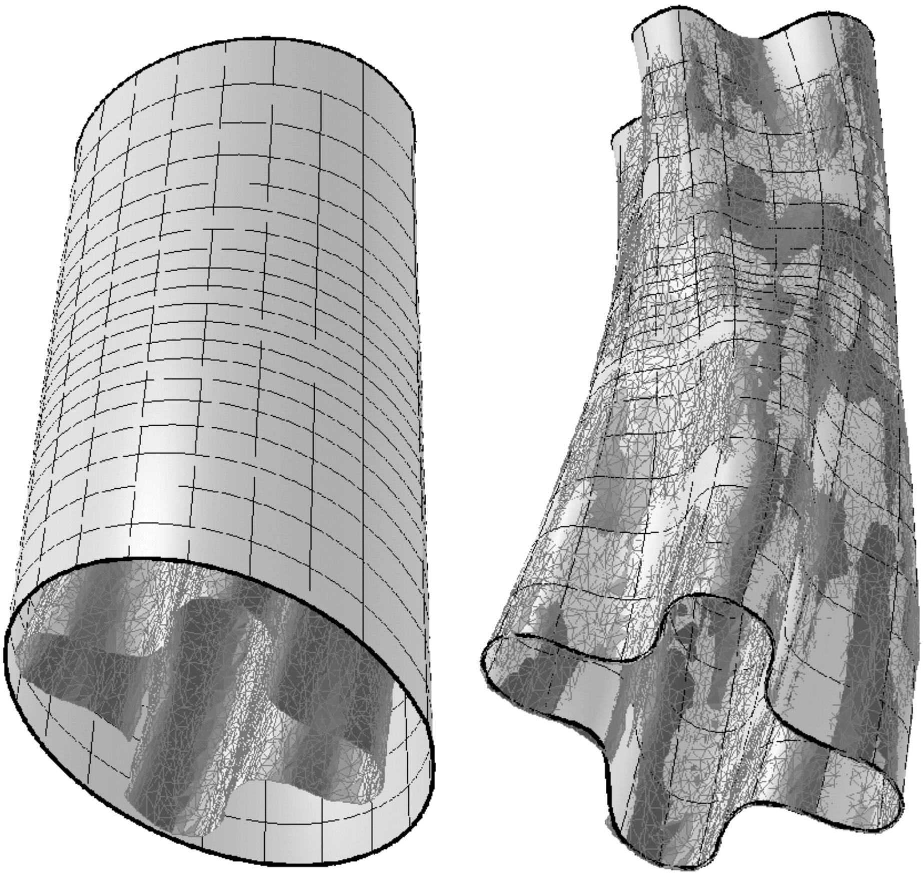

Fig. 10. Approximation of a model surface (triangulated point cloud) with a closed tensor product B-spline surface. Left: initial position of active B-spline. Right: final position of active B-spline after 20 iterations.

squared distance deviation (our method tempts to minimize this quantity in each iteration), the lower diagram gives the maximum squared distance deviation. The triangulated object is scaled to a bounding box of unit size 1. Then a mean squared distance deviation of $2.2 \cdot 10^{-6}$ and a maximum squared distance deviation of $1.5 \cdot 10^{-5}$ is reached after 20 iterations.

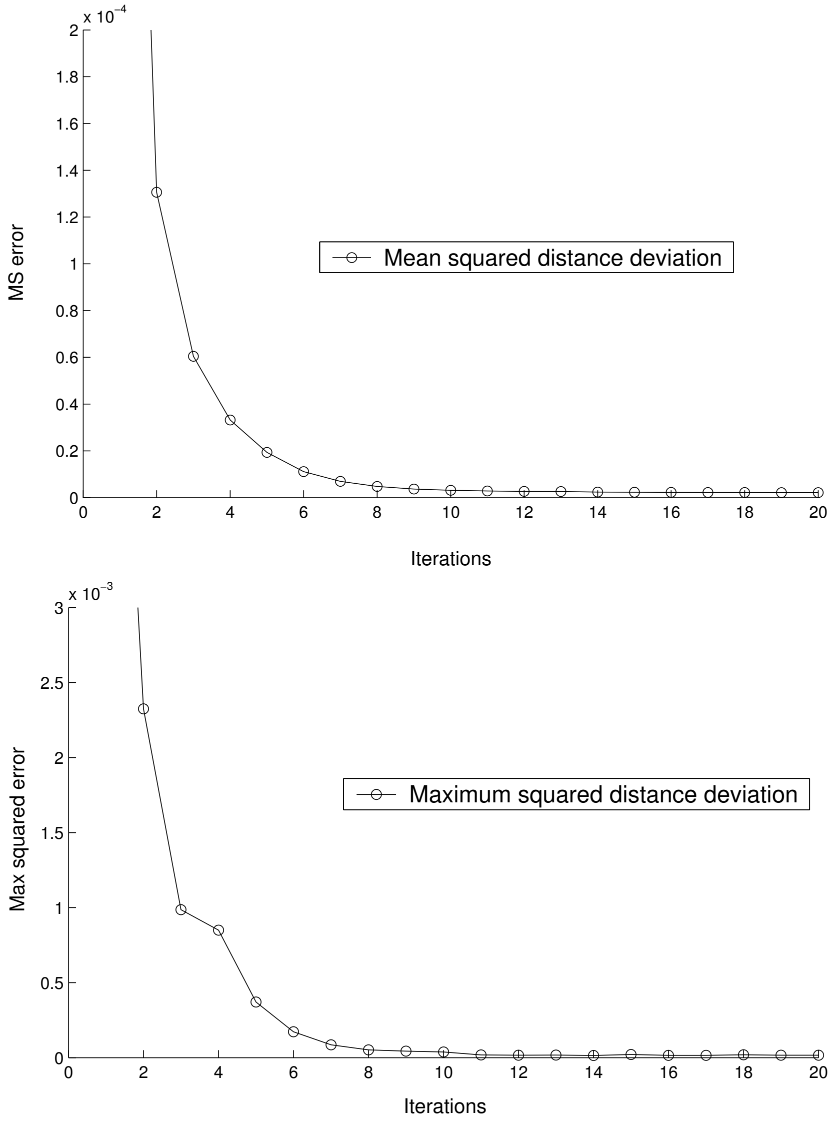

Fig. 11. Mean squared error (upper diagram) and maximum squared error (lower diagram) for the surface approximation of Example 4.

5 Conclusion and future research

We have presented an active contour model for surface approximation which avoids the parametrization problem. According to the nonlinearity of the surface approximation problem, it is an iterative method. It solves in each iteration step a linear system of equations, which arises from the minimization of a quadratic function. The new idea is that in this quadratic function we use local quadratic approximants of the squared distance function to the model shape which shall be approximated. In the present paper, we just outlined the idea and demonstrated its use at hand of some examples. There is a lot of room for extensions and future research directions. Let us mention a few of them:

- We need more research on the initial choice of the control points of the active B-spline surface. Furthermore it is important to adjust the number of control points dynamically, i.e., from iteration to iteration. For the planar curve case this is treated in [50] but the methods presented there are not directly applicable to surface approximation.
- The concept is applicable to *approximation with subdivision surfaces.* One can use an initial shape as in [29], but other choices and an appropriate handling of details require a lot of future research.
- An interesting extension concerns the incorporation of *shape constraints* such as convexity. For example, we can use the sufficient linear convexity conditions which have been derived by B. Jüttler for surface fitting with convex tensor-product splines [20,22]. In our framework, we would then have to solve a quadratic programming problem in each iteration step.
- Another interesting topic is the approximation with *special surface classes.* We have shown this at hand of ruled surfaces. A further important topic is surface approximation with *translational surfaces.* This has applications in architecture and other areas where a simple generation of the designed surface is desired for its practical realization.
- The surface parametrization problem is a correspondence problem. One has to find correspondences between points on the parameter domain and the model shape. Our present approach avoids to deal with this correspondence. Another correspondence problem is the *registration problem* in Computer Vision: Suppose that we have a *CAD model* from which a workpiece has been produced. This workpiece has been scanned with some 3D measurement device (laser scanning, light sectioning, ...) resulting in a *3D data point cloud* from the surface of this workpiece. Thereby, the CAD model shall describe the 'ideal' shape of the object and will be available in a coordinate system that is different to that of the 3D data point set. For the goal of shape inspection it is of interest to find the optimal Euclidean motion (translation and rotation) that aligns, or registers, the point cloud to the CAD model. This makes it possible to check the given workpiece for manufacturing errors and to visualize and classify the deviations.

  A well-known standard algorithm to solve such a registration problem is the iterative closest point (ICP) algorithm of Besl and McKay [3]. It defines to each point of the current position of the data point set a corresponding point, namely the closest point on the model shape. Then, ICP finds the best match in a least squares sense and iterates this procedure. However, the correspondences used are usually wrong, which increases the number of iterations. Using a motion of the data point set in the squared distance field with help of local quadratic approximants, avoids the correspondence problem and appears as promising direction for real time surface inspection [38].

- A broad area for future research is the investigation of algorithms for the solution of other geometric optimization problems. There, we believe it is important not to use an optimization algorithm as a black box, but adapt an optimization concept (Newton, quasi-Newton, sequential quadratic programming [5],...) in a geometric way to the special problem.

Acknowledgements

We would like to thank the referees for their valuable comments and suggestions to improve this paper. Furthermore, we thank Hongkai Zhao for his help in the implementation of his fast sweeping algorithm. This research has been carried out as part of the project P16002-N05 supported by the Austrian Science Fund (FWF). Additionally, this work was supported, in part, by the innovative project '3D technology' of Vienna University of Technology.

References

[1] Bajaj, C., Bernardini, F., Xu, G., Automatic reconstruction of surfaces and scalar fields from 3D scans. SIGGRAPH'95 Proceedings (1995), 193–198.

[2] Barles, G., Soner, H.M., Souganidis, P.E., Front propagation and phase field theory, SIAM J. Control and Optimization **31** (1993), 439–469.

[3] Besl, P. J., McKay, N. D., A method for registration of 3D shapes, IEEE Trans. Pattern Anal. and Machine Intell. **14** (1992), 239–256.

[4] Blake, A., Isard, M., *Active Contours*, Springer, 1998.

[5] Boggs, P. T., Tolle, J. W., Sequential quadratic programming, Acta Numerica (1995), 1-52.

[6] Boissonnat, J.D., Cazals, F., Smooth shape reconstruction via natural neighbor interpolation of distance functions, In: Proc. 16th Annu. ACM Sympos. Comput. Geom. (2000), 223–232.

[7] Brunnet, G., Hagen, H., Santarelli, P., Variational design of curves and surfaces, Surveys on Mathematics for Industry **3**, 1–27.

[8] Caselles, V., Kimmel, R., Sapiro, G., Geodesic active contours, Intl. J. Computer Vision **22** (1997), 61–79.

[9] Caselles, V., Kimmel, R., Sapiro, G., Sbert, C.: Minimal surfaces: A geometric three-dimensional segmentation approach, Numerische Mathematik **77** (1997), 423–451.

[10] Chen, H.-Y., Pottmann, H., Approximation by ruled surfaces, J. Comput. Applied Math. **102** (1999), 143–156.

[11] Choi, H.I, Choi, S.W., Moon, H.P.: Mathematical theory of medial axis transform. Pacific J. Math. **181** (1997), 57–88.

[12] Choi, H.I., Choi, S.W., Moon, H.P.: New algorithm for medial axis transform of plane domain. Graphical Models and Image Processing **59** (1997), 463–483.

[13] Desbrun, M., Cani-Gascuel, M.-P., Active implicit surface for animation, Graphics Interface (1998), 143–150.

[14] Dietz, U., Geometrie-Rekonstruktion aus Meßpunktwolken mit glatten B-Spline-Flächen. Diss., TU Darmstadt, 1998.

[15] Faugeras, O., Gomes, J., Dynamic shapes of arbitrary dimension: the vector distance functions, In: R. Martin, ed., The Mathematics of Surfaces IX, Springer, 2000, pp. 227–262.

[16] Frisken, S.F., Perry, R.N., Rockwood, A.P., Jones, T.R., Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics, Proceedings of SIGGRAPH 2000, pp. 249–254, 2000.

[17] Gomes, J., Faugeras, O., Reconciling distance functions and level sets, J. Visual Communication and Image Representation **11** (2000), 209–223.

[18] Greiner, G., Variational design and fairing of spline surfaces, Computer Graphics Forum **13** (1994), 143–154.

[19] Hoschek, J. and Lasser, D., *Fundamentals of Computer Aided Geometric Design*, A. K. Peters, Wellesley, MA., 1993.

[20] Hoschek, J., Jüttler, B., Techniques for fair and shape preserving surface fitting with tensor-product B-splines, In: Shape Preserving Representations in Computer-Aided Geometric Design (J.M. Peña, ed.), Nova Science Publ., Commak, New York, 1999.

[21] Hoschek, J., Schwanecke, U., Interpolation and approximation with ruled surfaces, In: The Mathematics of Surfaces VIII (R. Cripps, ed.), Information Geometers, 1998, pp. 213–231.

[22] Jüttler, B., Surface fitting using convex tensor-product splines, J. Comput. Appl. Math. **84** (1997), 23–44.

[23] Jüttler, B., Felis, A., Least-squares fitting of algebraic spline surfaces, Advances in Computational Mathematics **17** (2002), 135–152.

[24] Kass, M., Witkin, A., Terzopoulos, D., Snakes: Active contour models, Intern. J. Computer Vision **1** (1988), 321–332.

[25] Kimmel, R., Shaked, D., Kiryati, N., Bruckstein, A., Skeletonization via distance maps and level sets, Computer Vision and Image Understanding **62** (1995) 382–391.

[26] Kobbelt, L., Vorsatz, J., Labsik, U., Seidel, H.-P., A shrink wrapping approach to remeshing polygonal surfaces, Computer Graphics Forum **18** (1999), Eurographics '99 issue, C119–C130.

[27] Lee, Y.S., Koc, B., Ellipse offset approach and inclined zig-zag method for multiaxis roughing of ruled surface pockets, Comp. Aided Design **30** (1998), 957–971.

[28] Ma, W., Kruth, J. P., Parametrization of randomly measured points for the least squares fitting of B-spline curves and surfaces, Computer Aided Design **27** (1995), 663–675.

[29] Ma, W., Zhao, N., Catmull-Clark surface fitting for reverse engineering applications, in: Proc. Geometric Modeling and Processing, Hong Kong, 2000, pp. 274–283.

[30] Maekawa, T., An overview of offset curves and surfaces, Comp. Aided Design **31** (1999), 165–173.

[31] Malladi, R., Sethian, J. A., Vemuri, B. C., Shape modeling with front propagation: A level set approach, IEEE Trans. Pattern Anal. and Machine Intell. **17** (1995), 158–175.

[32] Nikolaidis, N., Pitas, I., *3-D Image Processing Algorithms*, Wiley, 2001.

[33] Osher, S., Fedkiw, R., Level set methods: an overview and some recent results, *J. Comp. Physics* **169** (2001), 463–502.

[34] Osher, S. J., Sethian, J. A., Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulation, Journal of Computational Physics **79** (1988), 12–49.

[35] Patrikalakis, N. M., Maekawa, T., *Shape Interrogation for Computer Aided Design and Manufacturing*, Springer, Berlin, 2002.

[36] Piegl, L, Tiller, W., *The NURBS book*, Springer Verlag, New York, 1995.

[37] Pottmann, H., Hofer, M., Geometry of the squared distance function to curves and surfaces, In: Hege, H.-C. and Polthier, K., eds., Visualization and Mathematics III, Springer, pp. 223–244.

[38] Pottmann, H., Leopoldseder, S., Hofer, M., Registration without ICP, Technical report 91, Institute of Geometry, Vienna University of Technology, February 2002.

[39] Pottmann, H., Leopoldseder, S., Hofer, M., *Approximation with active B-spline curves and surfaces*, Proc. of Pacific Graphics 2002, Beijing, IEEE Computer Society, pp. 8–25.

[40] Pottmann, H., Leopoldseder, S., Zhao, H.K., The d^2-tree: A hierarchical representation of the squared distance function, Technical report 101, Institute of Geometry, Vienna University of Technology, 2003.

[41] Pottmann, H., Wallner, J., *Computational Line Geometry*, Springer, 2001.

[42] Rusinkiewicz, S., Levoy, M., Efficient variants of the ICP algorithm, in Proc. 3rd Int. Conf. on 3D Digital Imaging and Modeling, Quebec, 2001, Springer-Verlag, 2001.

[43] Sapiro, G., *Geometric Partial Differential Equations and Image Analysis*, Cambridge Univ. Press, Cambridge, 2001.

[44] Serra, J., *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.

[45] Serra, J., Soille, J., eds., *Mathematical Morphology and its Applications to Image Processing*, Kluwer, Dordrecht 1994.

[46] Sethian, J. A., *Level Set Methods and Fast Marching Methods*, Cambridge University Press, 1999.

[47] Siddiqi, K., Tannenbaum, A., Zucker, S.W., A Hamiltonian approach to the eikonal equation, Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, 1999, pp. 1–13.

[48] Toriwaki, J., Yokoi, S., Distance transformations and skeletons of digitized pictures with applications, *Progress in Pattern Recognition*, L.N. Kanal and A. Rosenfeld, eds., North Hollang, 1981, pp. 187–264.

[49] Warren, J., Weimer, H, *Subdivision Methods for Geometric Design: A Constructive Approach*, Morgan Kaufmann Series in Computer Graphics, San Francisco, 2001.

[50] Yang, H., Wang, W., Sun, J., *Control Point Adjustment for B-Spline Curve Approximation*, submitted to CAD.

[51] Zhao, H.K., Chan, T., Merriman, B., Osher, S., A variational level set approach to multiphase motion, J. Comp. Physics **127** (1996), 179–195.

[52] Zhao, H.K., Osher, S., Merriman, B., Kang, M., Implicit, non-parametric shape reconstruction from unorganized data using variational level set method, Computer Vision and Image Understanding **80** (2000), 295–314.

[53] Zhao, H.K., Osher, S., Fedkiw, R., Fast surface reconstruction and deformation using the level set method, Proc. IEEE Workshop on Variational and Level Set Methods in Computer Vision, Vancouver, 2001.

[54] Zhao, H.K., Fast Sweeping Method for Eikonal Equations I, submitted to *SIAM Numerical Analysis.*