Fitting curves and surfaces to point clouds in the presence of obstacles

Simon Flöry

Geometric Modeling and Industrial Geometry Research Group, Vienna University of Technology, Wiedner Hauptstraße 8-10, A-1040 Wien, Austria

Abstract

We consider the problem of fitting B-spline curves and surfaces to point clouds in the presence of obstacles constraining this approximation at the same time. Therefore, we describe the fitting problem as optimization problem and employ an iterative procedure to solve it — the presence of obstacles poses constraints on this minimization process. We examine two families of obstacles: first, the point cloud itself is interpreted as obstacle, e.g. to reconstruct any apparant boundaries of the data set. Second, we define arbitrary regions the fitting must not penetrate. We discuss several numerical aspects of this constrained optimization and present experimental results for B-spline curve and surface fittings in the presence of obstacles.

Key words: Curve fitting, Surface fitting, Obstacles, Constrained optimization, Surface trimming

1 Introduction

The wide spread use of 3D laser scanning technology offers various possibilities to digitize real world objects and further process their virtual models on computers. One of the most common representation methods for the acquired measurement data stores a high number of surface sample coordinates, often denoted to as the *point cloud*. Data processing knows an ever increasing number of ways to deal with such a discrete point set: noise removal, mesh generation or registration to name only a few. A topic of special interest is to reduce the amount of information represented by the numerous elements of such a data set and smooth the point cloud at the same time. For this purpose, a curve or surface is *fitted* to the point set and used to represent the

Email address: floery@geometrie.tuwien.ac.at (Simon Flöry).

measurement data henceforth. Less surprisingly, both curve and surface fitting are closely related and build upon the same basic considerations. Due to their universality, free-form curves and surfaces are a popular choice as fitting entities and we focus on fittings with B-spline curves and surfaces below.

Conventional approaches in curve and surface fitting aim for an approximation of the shape represented by the point cloud. Often, this process is exposed to constraints of abstract nature to ensure mathematical properties of the final solution. In our work, we let real world obstacles influence the fitting by means of restricting the solution space for the final result. Within this context we are able to address several challenging tasks: boundary reconstruction of point clouds, computation of fittings avoiding certain forbidden regions, computation of hulls for moving objects and surface trimming to name only a few. In achieving these goals, the main contributions of this work comprise the formulation of linear constraints induced by real world obstacles that integrate well with existing fitting algorithms. In detail, we consider the point cloud itself as obstacle and subsets of the plane or space the final fitting must avoid. Moreover, we discuss numerical issues of the arising constrained optimization problem (suitable classes of algorithms and primal vs. dual solution).

The remaining parts of our paper are organized as follows. After a review of related literature we describe the general fitting problem and develop constraints once for a boundary approximation of the point set and second for more general obstacles. Subsequent to a short excursion to relevant topics of constrained optimization theory, several examples show applications of the proposed algorithm and discuss a handful of numerical properties.

1.1 Related Work

Piecewise parametric functions have been used in curve fitting of point clouds for a long time. In one of the first works on this topic, (Cox, 1971) employs a least-squares method, an error metric still popular today, to fit scattered data. A major challenge in curve fitting, if compared to a univariate function approximation, is to define those samples on the fitting entity the approximation error is measured in. If the data points are ordered, the chordal length method or the centripetal method (see e.g. (Hoschek and Lasser, 1993)) could be used for *parametrization*. (Hoschek, 1988) tackles the problem of unordered target data with an iterative method of intrinsic parametrization and approximates the foot point computation by a first order term. Approximations of higher order (Saux and Daniel, 2003; Hu and Wallner, 2005) or an accurate foot point computation (Hoschek and Lasser, 1993) in each step are subject of discussion as well. Once these foot points have been obtained, the distance function to the curve is approximated. The most popular technique computes the squared dis-

tance from the data points to the foot points (Plass and Stone, 1983; Hoschek, 1988; Goshtasby, 2000). (Blake and Isard, 1998) improve these approaches by means of convergence properties and utilizes the squared distance from the data point to the tangent plane in the foot point as error term. Recently, (Wang et al., 2006) give an thorough overview of existing curve fitting methods and describe an algorithm based on a second order approximation of the squared distance function to solve the curve fitting problem.

The problem of surface fitting has been addressed for years as well (see e.g. (Hayes and Halliday, 1974)) and has drawn a certain amount of attention in recent years. (Dierckx, 1993) give a comprehensive overview of curve and surface fitting techniques (scattered data fitting, mesh fitting and data smoothing) and is a good starting point for references to the older literature. (Dietz, 1995) extend the existing least-squares approximations and discuss the use of additional smoothing and regularization terms. (Greiner and Hormann, 1996) consider the surface reconstruction with hierarchical tensor product B-spline surfaces and (Diebel et al., 2006) apply a Bayesian method for probable surface reconstruction. The literature on this topic is extensive and we want to round up the review and refer to two recent publications (Weiss et al., 2002; Wang et al., 2006) and the references therein.

Constrained curve and surface design is a widely investigated topic in computer aided geometric design. Given a set of data points, side conditions have been imposed on interpolation methods with cubic splines and rational cubics to handle obstacles (Opfer and Oberle, 1988; Meek et al., 2003). The constrained approximation of point sets with Bezier or B-spline curves (cf. (Rogers, 1989; Bercovier and Jacobi, 1994)) may have various aims in mind, such as increasing the quality of the final fitting, simplifying it or ensuring certain geometric properties of the solution. Especially the latter task is addressed in detail in (Hoschek and Kaklis, 1996) where properties such as convexity and monotony of a final approximation are discussed. Certain shape preserving criterias or restrictions on the class of surfaces employed for approximation pose constraints to fitting problems as well, e.g. in reconstructions of developable surfaces (Pottmann and Wallner (1999)). In Reverse Engineering, it is of special interest to reconstruct geometric primitives from measured data. In (Benkó et al., 2002), various constraints are considered for multiple fittings of geometric objects, which are handled in a modified Newton iteration. Further constrained methods in Reverse Engineering are surveyed in (Várady and Martin, 2002) and (Fisher, 2004). Besides these theoretical constraints, our special focus is on side conditions derived from real world obstacles. (Myles and Peters, 2005) construct splines of prescribed smoothness obliged to stay in a channel with piecewise linear boundaries. In (Liu et al., 2005), a combination of surface fitting and registration based on a squared distance minimization algorithm is discussed and applied to constrained reverse engineering of CAD models. Especially for registration, (Huang et al., 2006)

describe the local, simultaneous, penetration free alignment of multiple data sets as constrained optimization problem. (Lin and Wang, 2002) tightly border planar point clouds resembling curves by B-spline interpolation of their boundary points. In (Flöry and Hofer, 2008), curve fittings are constrained to happen on a smooth parametric surface. This main constraint is extended by curve boundary approximations that represent simplified cases of what we will discuss in the following. Furthermore, it is a common problem in motion planning to deal with obstacles, see (Latombe, 1991).

2 Constrained curve and surface fitting

Let $P = \{\mathbf{p}_k \in \mathbb{R}^d : k = 1, \ldots, n\}$ denote a set of unordered points in the plane ($d = 2$) or Euclidean three space ($d = 3$). Fitting a curve in the plane and fitting a surface in space to such a point cloud are closely related topics. For the ease of discussion, we will concentrate on the curve case in the following and point out differences to the surface case where necessary.

2.1 General curve fitting

We choose to carry out the fitting (or *approximation*) of P with a parametric curve $\mathbf{x} : \mathbb{R} \to \mathbb{R}^2, u \to \mathbf{x}(u)$. For now, we assume that $\mathbf{x}$ is closed and does not exhibit any self-intersections (we are going to show ways to deal with more general approximating curves in Sec. 2.2 and 4). Furthermore, let $d(\mathbf{x}, \mathbf{p})$ denote the signed distance of a point $\mathbf{p} \in \mathbb{R}^2$ to $\mathbf{x}$. Then, the curve $\mathbf{x}$ is said to fit P in a least-squares sense if it minimizes the sum of squared distances from each element $\mathbf{p}_k \in P$ to $\mathbf{x}$,

$$\sum_{k=1}^{n} d^2(\mathbf{x}, \mathbf{p}_k) + \omega \cdot r. \tag{1}$$

Here, r resembles a regularization term weighted by a factor ω, on which we will give motivation and details later on (see Sec. 4). Please note that by minimizing the squared distance function we avoid any handling of the distance function's sign.

Common approaches tackle the *optimization* problem of Equ. (1) in an iterative way. For the current location of the fitting curve $\mathbf{x}$, the squared distance function d^2 from $\mathbf{x}$ is described approximately. Usually, this is done for each data point (or the corresponding closest point on $\mathbf{x}$); an updated position of $\mathbf{x}$ is obtained by minimizing the sum over all these approximated distance functions. Details on approximations of the squared distance function are dis-

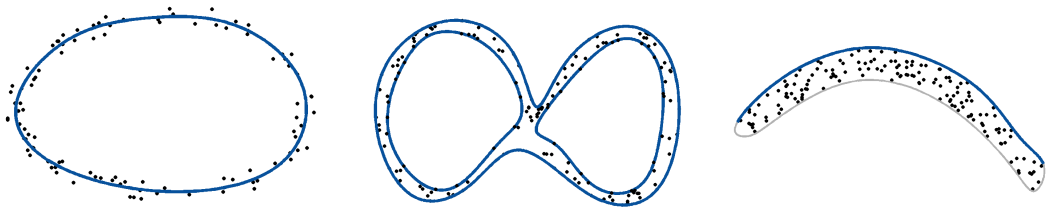

Fig. 1. (Left) A classical least-squares curve fitting results in a balance of residues. (Middle) A point cloud resembling a self-intersecting shape is — due to its discrete nature — interpreted as not self-intersecting but with several interior regions. (Right) Open curve shape approximation.

cussed in the literature, for example (Wang et al., 2006) gave a thourough overview recently.

So far, the fitting problem and its roughly outlined solution have been described for any general parametric curve. However, we want to get more specific about the employed curve type. As stated in the introduction we will use B-spline curves as fitting curves. Assuming that knots and degree of such a B-spline curve are fixed, we write $\mathbf{x}(u) = \sum_{i=1}^{m} N_i(u)\mathbf{d}_i$, where $N_i(u)$ are the B-spline basis functions and $\mathbf{d}_i \in \mathbb{R}^2$ describe the curve's control points. Accordingly, we denote by $\mathbf{x}_c(u) = \sum_{i=1}^{m} N_i(u)(\mathbf{d}_i + \mathbf{c}_i)$ any updated position of $\mathbf{x}$. For convenience, we will summarize the displacements $\mathbf{c}_i$ into a single displacement vector $\mathbf{c}$. With this notation in mind, we are able to describe a general fitting algorithm.

(1) Assign each data point $\mathbf{p}_k$ a parameter value u_k such that $\mathbf{x}(u_k)$ is the closest point of $\mathbf{p}_k$ on $\mathbf{x}(u)$.
(2) Describe the current fitting error of $\mathbf{x}$ w.r.t. P. Therefore, approximate the squared distance function of $\mathbf{x}$ to P in the foot points $\mathbf{x}(u_k)$.
(3) Compute the displacements $\mathbf{c}$ of the fitting curve's control points by minimizing this approximation error. If the fitting is of satisfactory quality, stop, or continue at step 1, otherwise.

Commonly used stopping criterias ask for the number of iterations exceeding a pre-defined value or demand for the averaged squared distances from P to $\mathbf{x}$ to fall below a user-defined threshold.

2.2 Boundary approximation

Considering the point cloud P, a fitting as outlined above will yield a final approximation following the shape of the point cloud by means of a balance in residues. Loosely speaking, if we assume that the point cloud exhibits some width, the curve will pass through the middle of the point cloud stripe, such that there are points to both sides of $\mathbf{x}$ (see Fig. 1, left). In the following, we

are interested in achieving an opposite effect and aim at approximating any apparant boundaries of the point cloud.

Let $\mathbf{x}$ be a closed parametric curve (in the plane) or a surface (in 3D) without self intersections and $d(\mathbf{x}, \mathbf{p})$ the signed distance function to $\mathbf{x}$ defined in such a way that $d(\mathbf{x}, \mathbf{p}) < 0$ holds for all points $\mathbf{p} \in D$, whereas D denotes the domain bounded by $\mathbf{x}$. By considering the sign of the distance function, we are able to define a boundary reconstruction of a point cloud: $\mathbf{x}$ approximates the *outer* or *inner boundary* of P if $\mathbf{x}$ minimizes Equ. (1) and $d(\mathbf{x}, \mathbf{p}_k) < 0$ or $d(\mathbf{x}, \mathbf{p}_k) > 0$ holds for all $\mathbf{p}_k \in P$.

In the following we will explicitely exclude fittings with self intersecting curves or surfaces. This doesn't pose any limitations to our succeeding considerations if we consider that the notion of a self intersecting point cloud is hard to grab. Given that a point cloud is a discrete set of data points, such a point cloud may be seen as a non self-intersecting point set at the same time, exhibiting a well defined outer boundary and an inner boundary consisting of several non connected components (see Fig. 1, middle). Please note, that for some point clouds, there might be no meaningful inner boundary at all (see Fig. 1, right, and Fig. 8, bottom right).

For fittings of point clouds resembling an open curve shape, the above definition doesn't make sense as for an open fitting curve no inner or outer region can be defined. Nevertheless, such an open shape might be seen as special case of a point cloud with no inner boundary. Thus, we approximate the outer boundary of such a point cloud with an artificially closed approximation curve and restrict the parameter space of the final fitting curve $\mathbf{x}$ to those parts of the boundary we wanted to approximate (see Fig. 1, right).

We have seen that the problem of curve fitting can be turned into an optimization problem that is solved with an iterative procedure. Approximating the boundary of a point cloud imposes constraints on this optimization problem, e.g. the outer boundary of a point set is reconstructed by that curve minimizing the constrained optimization problem,

$$\begin{aligned} &\text{minimize} \quad \sum_{k=1}^{n} d^2(\mathbf{x}, \mathbf{p}_k) + \omega \cdot r \\ &\text{subject to} \quad d(\mathbf{x}, \mathbf{p}_k) < 0 \quad \forall \mathbf{p}_k \in P. \end{aligned} \tag{2}$$

For a solution of the unconstrained minimization problem we approximated the squared distance function by quadratic functions. Following the same idea, we will now derive suitable approximative side conditions from the definition of boundaries introduced above. For an approximation of the outer boundary, $d(\mathbf{x}, \mathbf{p}_k) < 0$ must hold for all $\mathbf{p}_k \in P$. $d(\mathbf{x}, \mathbf{p}_k)$ describes the signed distance from $\mathbf{p}_k$ to $\mathbf{x}$, which defines a foot point $\mathbf{x}(u_k)$ such that $|d(\mathbf{x}, \mathbf{p}_k)| = \|\mathbf{x}(u_k) - \mathbf{p}_k\|$. By computing the first order Taylor approximation of d in that foot point

$\mathbf{x}(u_k)$, we obtain

$$d_k(\mathbf{x}, \mathbf{p}) \approx d(\mathbf{x}(u_k)) + \nabla_p d(\mathbf{x}(u_k))^T \cdot (\mathbf{p} - \mathbf{x}(u_k)).$$

As $d(\mathbf{x}(u_k)) = 0$ holds and $\nabla_p d$ coincides with the outward normal vector $\mathbf{n}(u_k)$ in $\mathbf{x}(u_k)$, we approximate the constraint $d(\mathbf{x}, \mathbf{p}_k) < 0$ for a data point $\mathbf{p}_k$ by

$$(\mathbf{p}_k - \mathbf{x}(u_k))^T \cdot \mathbf{n}(u_k) < 0, \tag{3}$$

which basically means that we replace $\mathbf{x}$ by its tangent in the foot point and require $\mathbf{p}_k$ to be located on the opposite side of the tangent the normal points to.

In the course of optimization, $\mathbf{x}$ will get displaced to better approximate the point cloud. Thus, the foot point $\mathbf{x}(u_k)$ and the normal $\mathbf{n}(u_k)$ will change as they depend on the current shape of the fitting curve, given by the current control points (summarized in a single vector $\mathbf{d}$). Precisely speaking, for our choice of B-splines as approximating curves, $\mathbf{x}(u_k)$ and $\mathbf{n}(u_k)$, and thus the whole left hand side of Equ. (3),

$$l(\mathbf{d}) = (\mathbf{p}_k - \mathbf{x}(u_k(\mathbf{d}), \mathbf{d}))^T \cdot \mathbf{n}(u_k(\mathbf{d}), \mathbf{d}),$$

depends on the control points in a highly non-linear way. We are going to consider two ways to deal with this dependancy. Our first approach will simply regard the foot points as fixed and thus ignore this relation at all,

$$(\mathbf{p}_k - \mathbf{x}_c(u_k^0))^T \cdot \mathbf{n}(u_k^0) < 0. \tag{4}$$

Here, superscript index 0 labels values at the beginning of the current iteration.

Another way to deal with this dependancy is to linearize $l(\mathbf{d})$,

$$l(\mathbf{d}) \approx l(\mathbf{d}^0) + \nabla_d l(\mathbf{d}^0)^T \cdot (\mathbf{d} - \mathbf{d}^0).$$

For the computation of the gradient $\nabla_d l$, partial derivation of $l(\mathbf{d})$ with respect to $\mathbf{d}_i$ yields

$$\frac{\partial l}{\partial \mathbf{d}_i}(\mathbf{d}) = -N_i(u_k(\mathbf{d})) \cdot \mathbf{n} + \frac{\partial \mathbf{n}}{\partial \mathbf{d}_i} \cdot (\mathbf{p}_k - \mathbf{x}).$$

Here, N_i describe the B-spline basis function again. Let J denotes the rotation by $\pi/2$ turning $\dot{\mathbf{x}}(u_k)$ into the outward oriented normal vector. Then, $\frac{\partial \mathbf{n}}{\partial \mathbf{d}_i} = J\ddot{\mathbf{x}} \cdot \left(\frac{\partial u_k}{\partial \mathbf{d}_i}\right)^T + N_i \cdot \mathbf{1}$ with $\mathbf{1} \in \mathbb{R}^{2\times 2}$. The partial derivatives of $u_k(\mathbf{d})$ are defined in an implicit way by the condition that in a closest point $\mathbf{x}(u_k)$ the tangent $\dot{\mathbf{x}}(u_k)$ is perpendicular to the connecting vector from $\mathbf{x}(u_k)$ to $\mathbf{p}_k$,

$$\dot{\mathbf{x}}(u_k)^T \cdot (\mathbf{x}(u_k) - \mathbf{p}_k) = 0.$$

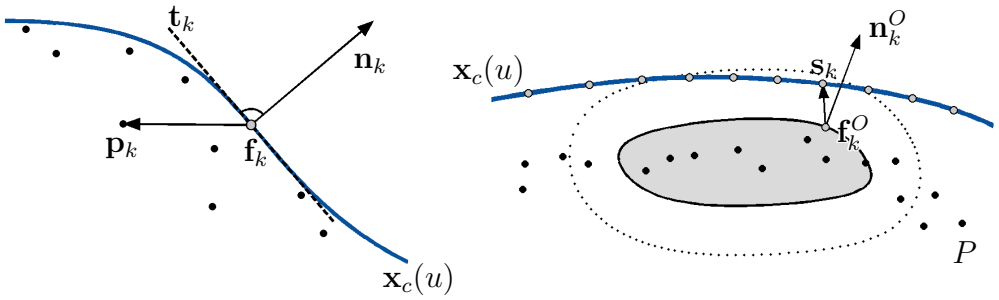

Fig. 2. (Left) At the end of an iteration, the data points lie on the opposite side of the tangent the normal points to. (Right) For smooth bounded general obstacles, all curve samples $\mathbf{s}_k$ within a certain distance (denoted by the dashed line) to an obstacle's boundary are constrained to be located outside any forbidden region.

Derivation of this expression with respect to $\mathbf{d}_i$ yields

$$\frac{\partial u_k}{\partial \mathbf{d}_i}(\mathbf{d}) = -\frac{\dot{N}_i \cdot (\mathbf{x} - \mathbf{p}_k) + N_i \cdot \dot{\mathbf{x}}}{\ddot{\mathbf{x}}^T \cdot (\mathbf{x} - \mathbf{p}_k) + \dot{\mathbf{x}}^T \cdot \dot{\mathbf{x}}}.$$

By requiring the linear approximation of $l(\mathbf{d})$ to be negative (compare Equ. (3)), we obtain for $\mathbf{c} = \mathbf{d} - \mathbf{d}^0$ the linear constraints of the second type,

$$\nabla l(\mathbf{d}^0)^T \cdot \mathbf{c} < -(\mathbf{p}_k - \mathbf{x}(u_k^0))^T \cdot \mathbf{n}(u_k^0). \tag{5}$$

These considerations are generalized to the three dimensional surface case straight forward. For a B-spline surface $\mathbf{x}(u,v) = \sum_{i=1}^{m'} N_i'(u,v)\mathbf{d}_i$, the normal vector in a foot point $\mathbf{x}(u_k, v_k)$ may be computed by the cross product of the derivatives along the parameter lines, $\mathbf{n}(u_k, v_k) = \mathbf{x}_u(u_k, v_k) \times \mathbf{x}_v(u_k, v_k)$. As before, the derivatives of $u(\mathbf{d})$ and $v(\mathbf{d})$ with respect to $\mathbf{d}_j$ for the partial derivatives of $l(\mathbf{d})$ are defined implicitely by the conditions for a foot point of the shortest distance between a sample $\mathbf{p}_k$ and a surface $\mathbf{x}$, which read in the surface case as

$$\mathbf{x}_u^T(u_k, v_k) \cdot (\mathbf{p}_k - \mathbf{x}(u_k, v_k)) = 0 \quad \text{and} \quad \mathbf{x}_v^T(u_k, v_k) \cdot (\mathbf{p}_k - \mathbf{x}(u_k, v_k)) = 0.$$

For the ease of reading, we omit the final expression for $\frac{\partial l}{\partial \mathbf{d}_i}$ and $\frac{\partial \mathbf{n}}{\partial \mathbf{d}_i}$ which are obtained similar to the curve case.

2.3 General obstacles

Going beyond the boundary approximation of point clouds, we might consider subsets $\{O_j\} \subseteq \mathbb{R}^2$ the final approximating curve $\mathbf{x}$ of a point cloud P is obliged to avoid. Given that each region O_j is bounded by a smooth curve $\mathbf{o}_j$, a feasable fitting in the presence of such obstacles can be defined in a

way closely related to the previous section on boundary reconstruction. If we assume that the distance function to $\mathbf{o}_j$ is negative for points inside an obstacle, a curve $\mathbf{x}$ does not penetrate any O_j if $d(\mathbf{o}_j, \mathbf{s}) > 0$ holds for all $\mathbf{s} \in \mathbf{x}$. Contrary to the previous considerations for boundary approximation, this definition naturally holds for closed and open curves, with and without self intersections.

For applications, we will weaken this condition. We discretize the curve $\mathbf{x}$ in dense samples $S = \{\mathbf{s}_k\}$ and ask for $d(\mathbf{s}_k) > 0$ for all $\mathbf{s}_k \in S$. Then, a curve fitting in the presence of general obstacles yields constraints of the same type as for the boundary case,

$$(\mathbf{f}_k^O - \mathbf{s}_k))^T \cdot \mathbf{n}_k^O > 0, \tag{6}$$

where $\mathbf{f}_k^O$ is the foot point of the shortest distance from a sample point $\mathbf{s}_k$ to the closest obstacle O_k and $\mathbf{n}_k^O$ the normal in that foot point. In an adapted sense, the same considerations for the dependancy of any samples, foot points and normals to the current location of $\mathbf{x}$ apply.

3 Constrained Optimization

So far, we described an iterative solution for the curve fitting problem along with approximative linear constraints induced by specific real world obstacles. In this section, we are going to restate the boundary approximation problem of Equ. (7) with above results and show ways to solve it. As noted in Sec. 2.1, the sum of squared distances in the objective function can be approximated as sum over quadratic terms $Q_k(\mathbf{c})$. Then, by making use of the approximative side conditions of Sec.2.2, we may write our constrained optimization problem in the form,

$$\begin{aligned} &\text{minimize} \quad \sum_{k=1}^{n} Q_k(\mathbf{c}) + \omega \cdot r(\mathbf{c}) \\ &\text{subject to} \quad \text{Equ. (4) or Equ. (5).} \end{aligned} \tag{7}$$

This minimization problem is solved at each iteration of the algorithm described in Sec. 2.1 to obtain an updated position of the approximating curve $\mathbf{x}$. As we are going to choose a regularization term $r(\mathbf{c})$ quadratic in the unknown displacments $\mathbf{c}$ (see Sec. 4), the objective function of Equ. (7) is quadratic in $\mathbf{c}$. If there were no constraints, a minimum would be found by solving a system of linear equations. However, we face constraints linear in $\mathbf{c}$.

There are basically two families of algorithms solving quadratic programming problems with linear constraints (see e.g. (Nocedal and Wright, 1999)). *Active set methods* have been the most important technique for a long time. They

deal with the constraints by estimating and continuously updating a set of currently active side conditions. The generalization of *Interior point methods* to quadratic programs, which in contrast avoid the boundary of the feasible region, provided a second powerful method for the solution of quadratic programs. While Active set methods are, in general, better suited for smaller scaled problems, Interior point methods are widely used for larger scaled optimization tasks.

Considering that the dimension of $\mathbf{c}$ in Equ. (7) is only two or three times the number of control points of $\mathbf{x}$ this optimization problem is rather small scaled from the viewpoint of optimization. However, the number of constraints may equal the possibly big number of elements in the point cloud. Therefore it might be favorable to solve the dual problem as it reduces the complexity of the constraints while increasing the dimension of the problem at the same time.

If we write the *primal* quadratic program of Equ. (7) in a more general form,

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\mathbf{c}^T H\mathbf{c} + \mathbf{t}^T\mathbf{c} \qquad H \in \mathbb{R}^{3m\times 3m},\ \mathbf{c},\mathbf{t} \in \mathbb{R}^{3m} \\ \text{subject to} & A\mathbf{c} - b \leq 0 \qquad A \in \mathbb{R}^{n\times 3m},\ \mathbf{b} \in \mathbb{R}^n, \end{array} \tag{8}$$

its Lagrange function is given by

$$L(\mathbf{c},\lambda) = \tfrac{1}{2}\mathbf{c}^T H\mathbf{c} + \mathbf{t}^T\mathbf{c} + \lambda^T(A\mathbf{c} - \mathbf{b}) \qquad \lambda \in \mathbb{R}^n.$$

The *dual* problem is then defined as finding the maximum of the dual function

$$q(\lambda) := \inf_{\mathbf{c}\in\mathbb{R}^{3m}} L(\mathbf{x},\lambda) = -\tfrac{1}{2}\lambda^T Q\lambda - \mathbf{q}^T\lambda - \tfrac{1}{2}\mathbf{t}^T H^{-1}\mathbf{t},$$

where we write in short $Q = AH^{-1}A^T$ and $\mathbf{q} = AH^{-1}\mathbf{t} + \mathbf{b}$. Therefore, the dual problem of the primal one in Equ. (8) is yet another quadratic program,

$$\begin{array}{ll} \text{maximize} & -\frac{1}{2}\lambda^T Q\lambda - \mathbf{q}^T\lambda \\ \text{subject to} & \lambda \geq 0. \end{array} \tag{9}$$

The solution of the primal problem is given as the minimal argument of $L(\mathbf{x},\lambda)$ for λ fixed, thus $\mathbf{c} = -H^{-1}(t + A^T\lambda)$. As for our optimization task strong duality holds, it could be favorable to focus on the dual problem. In the following section, we will examine experimentally which method shows best performance for our task of curve and surface fittings in the presence of obstacles.

4 Experiments and results

Now, as we have set up a theoretical framework for curve and surface fittings in the presence of obstacles, we want to apply these considerations to

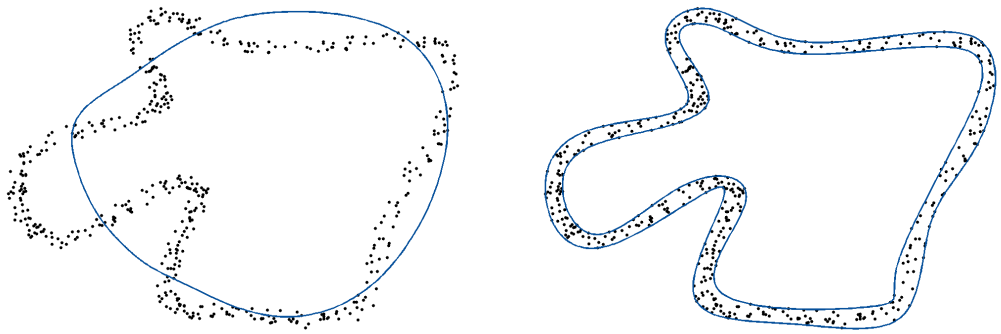

Fig. 3. Reconstruction of a point cloud's boundary: initial setup (left) and final approximation after 10 iterations (right).

a handful of examples. We describe various experiments showing the method in use — in particular, we will address any remaining open questions such as in how far the two proposed constraints of Equ. (4) and Equ. (5) show different performance and what is the best optimization technique to tackle the emerging constrained optimization problems. As basic fitting algorithm, we employ the method of *Squared Distance Minimization*, which builds upon second order Taylor approximation of the squared distance function in the foot points $\mathbf{x}(u_k)$ of the shortest distances from the elements of P to $\mathbf{x}$ (for a detailed description of this method, see (Wang et al., 2006)).

Another issue that needs to be addressed are details on the regularization term in Equ. (7). As an optimal solution in a mathematical sense does not necessarily mean a visually appealing solution (e.g. strong oscillations might occur), a simplified measure for the bending energy,

$$r(\mathbf{c}) = \int \|\ddot{\mathbf{x}}_c(u)\|^2 \mathrm{d}u,$$

is added to the problem's objective function. Please note that this expression is quadratic in $\mathbf{c}$. Details on the weighting factor ω (which is halved after each iteration step) can be found along with other relevant key figures of the following examples in Tab. 1. The algorithms for curve fittings were implemented in a Matlab environment whereas the surface fitting code was written in C/C++. All results were obtained on a *AMD Sempron Processor 3100+* computer.

Example 1 The first example is a simple application of the described algorithm and approximates the boundaries of a set of data points in the plane. The point cloud in this experiment was, as those of several succeeding examples, artifically created. For this purpose, an auxilary B-spline curve was sampled in a user specified number of points which were then displaced in normal direction by distances drawn from a Gaussian distribution. In Example 1, the approximating B-spline curve was cubic and counted 13 control points. By

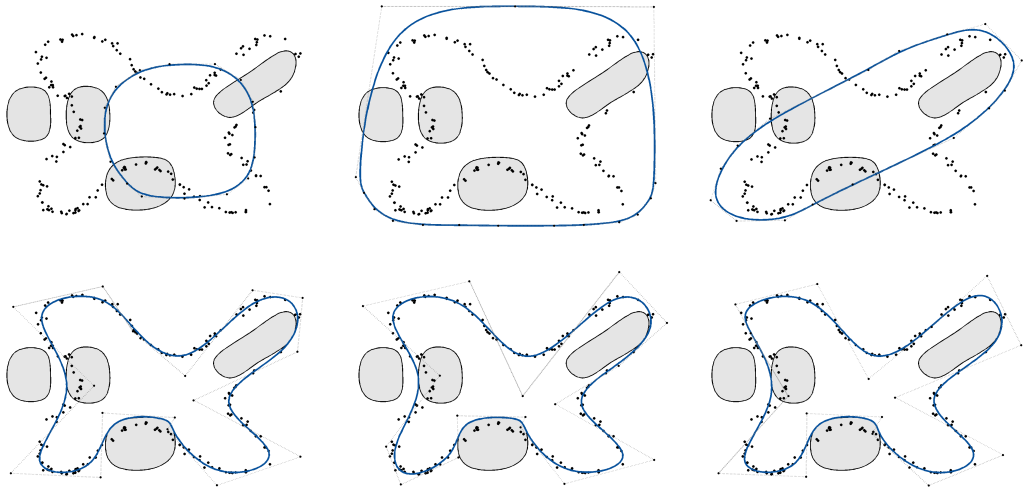

Fig. 4. Fitting a cubic B-spline curve to a point cloud in the presence of four general obstacles. For three different initial setups (top row), the corresponding final approximations after 25 iterations (bottom row) are shown.

using the second type linear constraints of Equ. (5), the inner and the outer boundary of the point cloud were approximated with an Active set method; both results were summarized in a single plot in Fig. 3 (right).

Example 2 The second example exhibits a curve fitting constrained by four smooth bounded forbidden regions. In a preprocessing step, an approximated distance field from the obstacles' boundaries was computed (applying the sweeping algorithm described in Tsai (2002)), which took 15.6 seconds on a $[0,1]\times[0,1]$ grid with a resolution of 0.01. At each iteration, the distance values of this discretization were used to determine those sample points $\mathbf{s}_k \in \mathbf{x}_c(u)$ close to a forbidden region ($\min_j d(\mathbf{s}_k, O_j) < 0.04$). For these samples, linear constraints of the second type were added to the optimization problem. Fig. 4 shows three approximations of the same setup (with respect to point cloud and obstacles) but with distinct differing initial positions of the approximating B-spline curve. As the results indicate, are the fittings barely influenced by the starting configuration.

For most of the experiments presented here, the linear constraints have been deactivated in the first 3 to 5 iterations (the exact number is given in brackets in the column holding the number of iterations in Tab. 1). This way, the fitting curve first approximates the point cloud before taking any obstacles into account - a strategy improving the stability of the method significantly. Please note, that the final solution of an approximation avoiding certain regions depends on the actual location of the fitting curve in the moment when the constraints are actived for the first time and is thus not unique. Obstacles inside the domain bounded by a closed approximating curve will remain

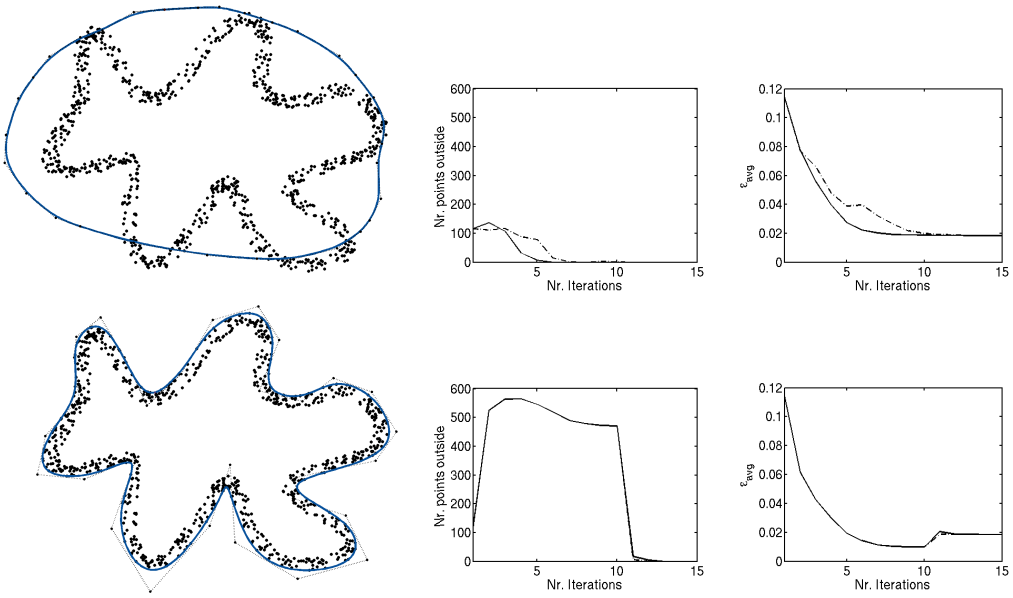

Fig. 5. (Left column) A boundary approximation with constraints obtained by linearizing any dependancy on the foot points. (Top row) If applying solely constrained iterations, this method (solid line) performs better than that ignoring any relation (dashed line). (Bottom row) Unconstrained iterations in the beginning cancel any advantage.

inside, the same applies to obstacles outside.

Example 3 In Sec. 2.2 we derived two types of linear constraints for a point cloud's boundary approximation. If the side conditions are active from the very beginning, the constraints of the second type obtained by linearization let us expect faster convergence as the curve doesn't fit the point set's shape well and thus causes relevant changes of the foot points. In contrast, if we first fit the point cloud with some unconstrained iterations roughly, using the more sophisticated constraints may not pay off as most of the curve's displacement happens in normal direction and thus leads to only minimal changes in the foot points. In Fig. 5, we compare both types of constraints in the course of an outer boundary approximation of a point cloud. The inital setup was the same for all optimizations, as well as the final fittings were of similar quality. If the side conditions are active from the beginning, the linearized constraints (solid line) outperfrom the constant constraints (dashed line), both in terms of decrease of points in the forbidden region and in terms of the average approximation error,

$$\epsilon_{avg} = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{x}(u_k) - \mathbf{p}_k\|.$$

However, if we apply 10 unconstrained iterations steps first, both approaches show similar performance, with the constant type performing even slightly better in the example at hand. Considering that the additional computation

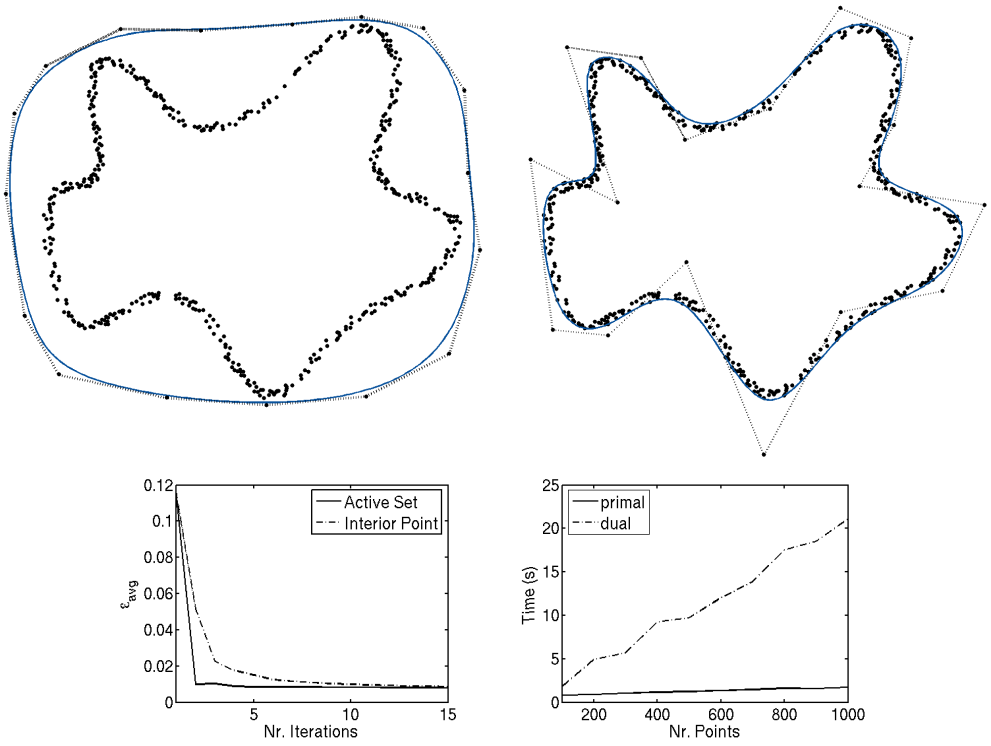

Fig. 6. (Top) Boundary approximation of a point cloud counting 500 elements by solving the dual problem. (Bottom left) An Active set method shows faster convergence than an Interior point method. (Bottom right) Solving the dual problem takes more and more time for a growing number of points.

time for the second type of constraints is minimal, one might favor the second approach applying a fitting without side conditions first. This is because the costs of an unconstrained optimization step are considerably lower than that of a constrained minimization.

The condition that the approximating curve is not self-intersecting proofed to be important for the theoretical work in Sec. 2. If we choose the initial position of the curve manually, this requirement can be fulfilled easily. If we obtain it in an automatic way, such as in parts for this example, we can avoid self-intersections in the course of optimization by choosing a high initial regularization weight ω and decrease it carefully.

Example 4 Given the boundary approximation problem of Fig. 6 we discuss two further numerical aspects of unconstrained optimization. First, as Active set methods are said to perform better than Interior point methods for small and medium scaled problems (such as our fitting problems), we want to examine which one of the two algorithm classes suits our purposes best. Though we are aware of the fact that a thorough investigation of this task is out of scope of our work, we compared Matlab's standard solver for medium

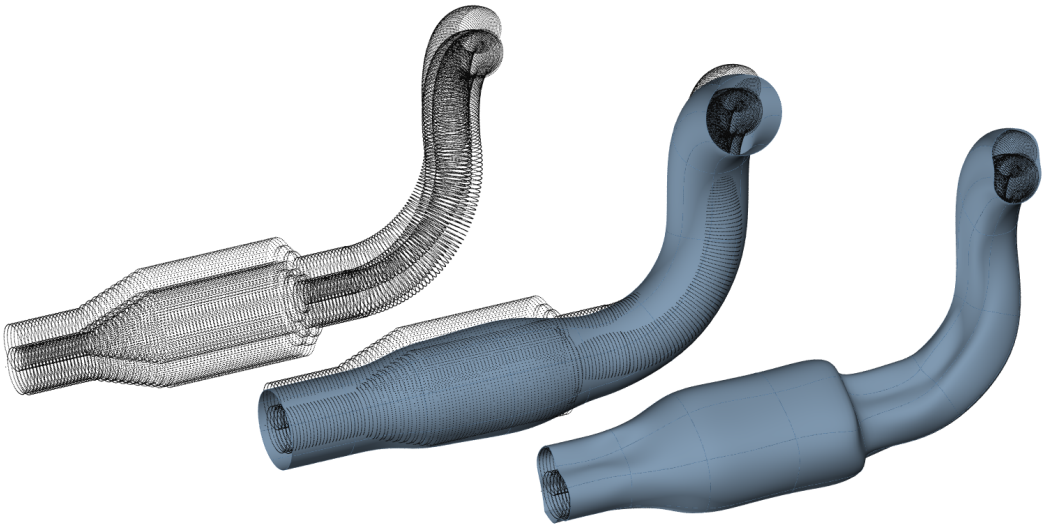

Fig. 7. Boundary approximation of a point cloud in 3D. A CAD model is sampled, exposed to simulated shakings (left) and approximated by a B-spline surface closed in one parameter direction (right).

scaled problems (an Active set method, cf. (Gill et al., 1981)) with a Matlab implementation of a recently proposed Interior point method (Absil and Tits, 2007). Fig. 6 (bottom, left) visualizes the convergence speed of both algorithms (with the second type of boundary constraints active from the beginning) over a period of 15 iterations. This comparison confirms advantages of the Active set method.

For a second numerical issue we notice that the dimension of the unknown displacement vector $\mathbf{c}$ is small, whereas the number of linear constraints grows with the size of the point cloud. Thus, as outlined in Sec. 3, it may be worth to address the dual problem which is larger scaled but shows more simple constraints that can be tackled with specialized algorithms. In our framework, we applied a solver for quadratic programs with only bound constraints (Neumaier, 1998). However, we couldn't observe any advantage in addressing the dual problem. We see two reasons for this: once, the increase in dimension is quite large when changing from the primal to the dual problem. Second, the dual problem shows worse condition than its primal counterpart.

Example 5 In another example, we want to illustrate the surface approximation of a point cloud's outer boundary. Within this context, we are going to address a common problem in engineering. Consider the mechanical parts of an operating machine, thus exposed to vibrations. If we are in need of a hull for such an object, we have to take into account all possible positions and wrap them with a surface. For our example, we use the CAD model of a pipe (see Fig. 7) and sample its surface in approximately 3000 points. Then, we simulate

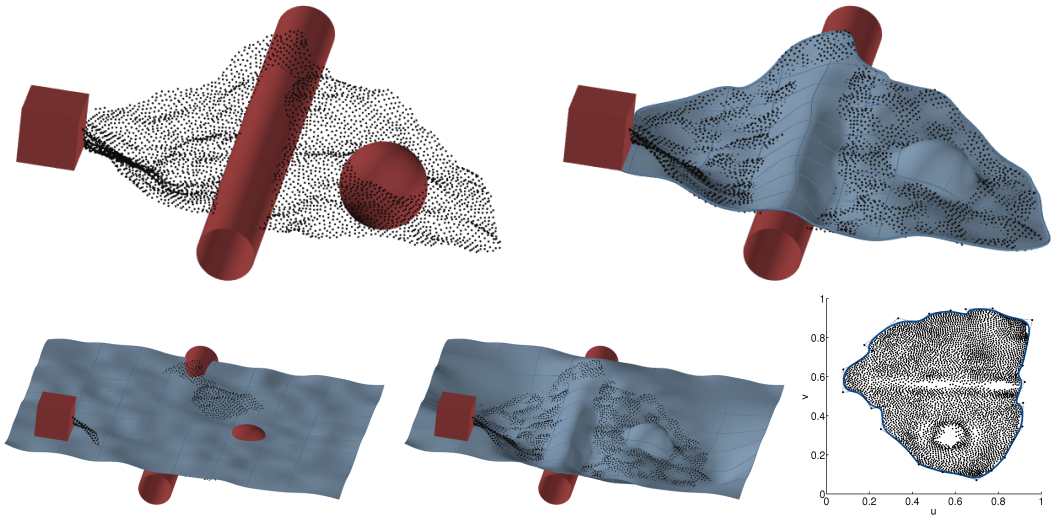

Fig. 8. (Top) Initial and final setup of a surface fitting in the presence of general obstacles. (Bottom) Some intermediate results: untrimmed initial surface, untrimmed final surface and parameter domain boundary approximation for trimming.

the shaking of operation by applying some random translations to this point cloud, whereas translations in positive direction of two coordinate axis may be observed more frequently. We summarize the relocated data points into a single point set and compute an approximation of its outer boundary. The B-spline surface is closed with respect to one parameter and counted 15×22 control points. The final result was trimmed such that the parameter domain ranges from 0 to 1 in the closed parameter direction and was bounded by straight lines in the other direction. In order to deal with the large number of points (and thus constraints), only data points within a distance of 0.05 to the approximating surface entered the optimization. Please note that we employed this strategy of including only a subset of P for constraining the optimization only in this example. In all other experiments, each $\mathbf{p}_k \in P$ contributed a side condition to the minimization.

Example 6 In this example, the point cloud results from digitizing an object with a 3D laser scanner. We place three geometric primitives (a sphere, a cylinder and a cube) close to the data set and ask for a B-spline surface reconstruction avoiding these objects. For this purpose, we first fit a plane to the point cloud, define the initial position of the control points on a rectangular grid thereon and distort these in normal direction. In order to prevent a shrinking of the approximating surface, we let it overlap the data set significantly and fix the outer frame of control points in the subsequent optimization. Then, we employ the algorithm of Sec. 2.3 to compute a surface fitting not penetrating the obstacles. Contrary to Example 2, we didn't discretize the distance to the obstacles on a grid but sampled the obstacles boundary and added constraints for the foot points of those samples on the approximating

Example	Nr. points in P	Iterations (constr.)	ω	Time (s)
1	500	10(3)	0.0001	3.0
2	200	25(5)	0.0001	14.5
3	1000	15(0)	0.001	16.4, 16.8
		15(10)	0.001	13, 13.2
4	$100 \ldots 1000$	15(0)	0.005	9.8, see also Fig. 6
5	60000	15(5)	0.04	277.3
6	5000	15(5)	0.01	208.5

Table 1
Configuration parameters and runtime information for the examples of Sec. 4.

surface. For a proper visualization, we trim the fitting B-spline surface to the effective extent of the point cloud. Therefore, we determine the foot points of the point cloud's elements on the final surface and consider the corresponding parameters in (u, v)-parameter space. There, the parameters form a point cloud by themselves; the outer border of this point cloud is a good estimate for the boundary of the trimmed surface. We approximate the extent of the parameters with a B-spline curve with the methods of Sec. 2.2 and let the resulting fitting curve define the boundary of parameters contributing to the trimmed surface. Fig. 8 visualizes the final result of the constrained, trimmed surface fitting.

At this point, we want to refer one more time to Tab. 1, in which we summarize information on the examples' point clouds and their approximations. The second column holds the number of elements of the point cloud. For Example 4, the visualized boundary approximation and comparison of Active set and Interior point method was done for 500 data points, while the performance of both methods for a growing number of points was measured for point clouds with sizes ranging from 100 to 1000 points. The third column gives the total number of iterations to obtain the final solution, the numbers in brackets give the share of any unconstrained iterations steps. For the timing values of Example 3, approximations with side conditions ignoring any dependancy on the foot points come first. In general, if an example comprises more than a single approximation but Tab. 1 includes only a single value, the measured numbers were identical (iterations) or very similar (time).

5 Conclusions and future research

We presented an algorithm to fit a curve (in 2D) or a surface (in 3D) to a set of unordered points in the presence of obstacles. As obstacles, we considered

once the point cloud itself as constraint. Second, we restricted the approximation to a subset of the plane or space. By adapting the idea of common fitting approaches that interpret the fitting problem as optimization problem solved in an iterative way, we derived constraints to this optimization from the obstacles under consideration. Basically, we presented two types of constraints, both dealing with the dependancy of the constraints on the changing foot points in a different way. While the unconstrained fitting problem yields minimization of a quadratic objective function, the introduced side considitions are linear in the unknown displacement of the curve. We discussed some questions concerning the solution of such a quadratic program with linear constraints and illustrated the way the algorithm works in various examples.

We see several directions our work on constrained fittings may be continued. The fact that we keep knot vector and number of control points of the approximating B-spline curve or surface fixed poses certain limitations on the adaptiveness of the method. Thus it is of interest in how far existing solutions for the unconstrained case (cf. (Yang et al., 2004)) can be integrated into our constrained approximation framework to increase the flexibility of our algorithm. The error term describing the current approximation error poses further research challenges we hope to address in the future. Most approaches so far employ approximations of the squared distance function, thus obtaining solutions in a least-squares sense. Least-squares fitting is known to be sensitive to outliers, a problem that could be overcome by an approximation in the L_1 norm, leading to non-smooth optimization problems.

Today, most point clouds forming the input to fitting problems originate from 3D laser scanners. Such scanners, as every physical measurement device, exhibit distinct noise characteristics that can be determined in experiments and described in a mathematical way. Knowledge about this noise can enter the approximation process such that not the curve or surface with smallest distance to the point cloud is computed but the *most probable* curve or surface, given the characteristic noise. We hope to address these issues in future research.

Acknowledgements

I would like to thank Helmut Pottmann for his valuable comments and suggestions throughout the development of this work and the anonymous reviewer for their efforts to improve the paper. This research was supported by the Austrian Science Fund (FWF) as part of the projects P16002-N05 and P18865-N13 and a Neumaier fellowship.

References

Absil, P.-A., Tits, A. L., 2007. Newton-KKT interior-point methods for indefinite quadratic programming. Comput. Optim. Appl. 36 (1), 5–41.

Benkó, P., Kósa, G., Várady, T., Andorb, L., Martin, R., 2002. Constrained fitting in reverse engineering. Comput. Aided Geom. Des. 19 (3), 173–205.

Bercovier, M., Jacobi, A., 1994. Minimization, constraints and composite Bézier curves. Comput. Aided Geom. Des. 11 (5), 533–563.

Blake, A., Isard, M., 1998. Active Contours. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Cox, M. G., 1971. Curve fitting with piecewise polynomials. IMA J Appl Math 8 (1), 36–52.

Diebel, J. R., Thrun, S., Brünig, M., 2006. A Bayesian method for probable surface reconstruction and decimation. ACM Trans. Graphics 25 (1), 39–59.

Dierckx, P., 1993. Curve and surface fitting with splines. Oxford University Press, Inc., New York, NY, USA.

Dietz, U., February 1995. Erzeugung glatter Flächen aus Meßpunkten. Tech. Rep. 1717, Preprint Series, Department of Mathematics, Darmstadt TU.

Fisher, R. B., 2004. Applying knowledge to reverse engineering problems. Computer-Aided Design 36 (6), 501–510.

Flöry, S., Hofer, M., 2008. Constrained curve fitting on manifolds. Computer-Aided Design 40 (1), 25–34.

Gill, P. E., Murray, W., Wright, M. H., June 1981. Practical Optimization. Elsevier.

Goshtasby, A. A., 2000. Grouping and parameterizing irregularly spaced points for curve fitting. ACM Trans. Graphics 19 (3), 185–203.

Greiner, G., Hormann, K., 1996. Interpolating and approximating scattered 3D-data with hierarchical tensor product splines. In: Mehaute, A. L., Rabut, C., Schumaker, L. L. (Eds.), Surface Fitting and Multiresolutional Methods. Vanderbilt University Press, pp. 163–172.

Hayes, J. G., Halliday, J., 1974. The least-squares fitting of cubic spline surfaces to general data sets. IMA J Appl Math 14 (1), 89–103.

Hoschek, J., 1988. Intrinsic parametrization for approximation. Comput. Aided Geom. Des. 5 (1), 27–31.

Hoschek, J., Kaklis, P. (Eds.), 1996. Advanced Course on FAIRSHAPE. B. G. Teubner.

Hoschek, J., Lasser, D., 1993. Fundamentals of Computer Aided Geometric Design. AK Peters.

Hu, S.-M., Wallner, J., 2005. A second order algorithm for orthogonal projection onto curves and surfaces. Comput. Aided Geom. Des. 22 (3), 251–260.

Huang, Q.-X., Flöry, S., Gelfand, N., Hofer, M., Pottmann, H., 2006. Reassembling fractured objects by geometric matching. ACM Trans. Graphics 25 (3), 569–578, Proc. SIGGRAPH 2006.

Latombe, J.-C., 1991. Robot Motion Planning. Kluwer Academic Publishers, Norwell, MA, USA.

Lin, H., Wang, G., 2002. Interval b-spline curve evaluation bounding point cloud. In: PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications. IEEE Computer Society, Washington, DC, USA, p. 424.

Liu, Y., Pottmann, H., Wang, W., August 2005. Constrained 3D shape reconstruction using a combination of surface fitting and registration. Tech. Rep. 144, Geometry Preprint Series, Vienna Univ. of Technology.

Meek, D. S., Ong, B. H., Walton, D. J., 2003. Constrained interpolation with rational cubics. Computer-Aided Design 20 (5), 253–275.

Myles, A., Peters, J., 2005. Threading splines through 3D channels. Computer-Aided Design 37 (2), 139–148.

Neumaier, A., 1998. Minq - general definite and bound constrained indefinite quadratic programming.
URL `http://www.mat.univie.ac.at/~neum/software/minq/`

Nocedal, J., Wright, S. J., 1999. Numerical Optimization. Springer.

Opfer, G., Oberle, H. J., 1988. The derivation of cubic splines with obstacles by methods of optimization and optimal control. Numer. Math. 52, 17–31.

Plass, M., Stone, M., 1983. Curve-fitting with piecewise parametric cubics. In: SIGGRAPH '83: Proc. of the 10th annual conference on Computer graphics and interactive techniques. ACM Press, New York, NY, USA, pp. 229–239.

Pottmann, H., Hofer, M., 2003. Geometry of the squared distance function to curves and surfaces. In: Hege, H.-C., Polthier, K. (Eds.), Visualization and Mathematics III. Springer, pp. 223–244.

Pottmann, H., Leopoldseder, S., 2003. A concept for parametric surface fitting which avoids the parametrization problem. Comput. Aided Geom. Design 20 (6), 343–362.

Pottmann, H., Wallner, J., 1999. Approximation algorithms for developable surfaces. Comput. Aided Geom. Design 16 (10), 539–556.

Rogers, D. F., 1989. Constrained B-spline curve and surface fitting. Comput. Aided Des. 21 (10), 641–648.

Saux, E., Daniel, M., 2003. An improved Hoschek intrinsic parametrization. Comput. Aided Geom. Des. 20 (8-9), 513–521.

Tsai, Y. R., 2002. Rapid and accurate computation of the distance function using grids. J. Comput. Phys. 178 (1), 175–195.

Várady, T., Martin, R. R., 2002. Reverse engineering. In: Handbook of Computer Aided Geometric Design. Elsevier, pp. 651–681.

Wang, W., Pottmann, H., Liu, Y., 2006. Fitting B-spline curves to point clouds by squared distance minimization. ACM Trans. Graphics 25 (2), 214–238.

Weiss, V., Andor, L., Renner, G., Várady, T., 2002. Advanced surface fitting techniques. Comput. Aided Geom. Des. 19 (1), 19–42.

Yang, H., Wang, W., Sun, J.-G., 2004. Control point adjustment for b-spline curve approximation. Computer-Aided Design 36 (7), 639–652.