Construction of Surfaces by Shape Preserving Approximation of Contour Data

Bert Jüttler

University of Dundee

Abstract: We present a two–stage approach for the construction of surfaces from contour data. At first the contour curves are found using an algorithm for shape preserving least–square approximation of planar data by polynomial parametric spline curves. The obtained curves are then interpolated by tensor–product B–spline surfaces. The interpolation scheme used in the second step preserves the signs of the sectional curvature of the contours.

1 Introduction

Methods for the construction of surfaces from contour data are required in several applications of Computer Aided Geometric Design. These applications include the design of ship hulls, or also the reconstruction of bone surfaces in medical imaging. References to related literature can be found in the survey articles [13, 14]. As one of the possible approaches, such methods can be composed from two parts. At first, the contour curves of the desired surface are generated from the data. The contours are then interpolated in order to obtain the desired surface. We are particularly interested in methods which preserve the shape of the data as far as possible.
An algorithm for shape preserving least–square approximation of planar data by polynomial parametric spline curves is discussed in the first part of the present paper. It can be used for constructing the contour curves of the surface from the given data. Whereas shape preserving interpolation by parametric curves has been discussed in a number of papers (see the references cited in [9, pp. 103, 105, 114], see [7] for space curves), the corresponding approximation problem has not been considered so far. Even in the case of functional data, only the method proposed by Dierckx [2, 3] seems to be available. In Dierckx' method least–square approximation by cubic spline functions is formulated as a quadratic programming problem. Our approach generalizes this idea to the case of polynomial parametric spline curves. The desired shape of the approximating spline curve is specified with the help of a reference curve which is used in order to generate linear sufficient shape constraints. The control points of the approximating curve result by solving a quadratic programming problem.
After the contour curves have been obtained, the desired surface is to be found by interpolating them in the second step. In order to preserve the shape of the given data we use the notion of sectional curvature preserving interpolation which has been introduced by Kaklis and Ginnis [12]. Their approach to achieve this property is based on piecewise polynomial surfaces of non–uniform degree, whereby the degrees of the segments act as tension parameters. In contrast with this, we guarantee the desired shape with the help of appropriate

linear constraints. The control points of the interpolating surface are then found by solving a simple optimization problem.

2 Approximation of planar data

In this section we describe a method for approximating a sequence of points by a B–spline curve satisfying some shape constraints. More precisely, the desired inflection points and the curvature signs of the spline segments in between can be specified. In order to be brief, we give an outline of the basic ideas of our approach only. For more details of the approximation scheme the reader is referred to [10].

2.1 Problem statement

Let a sequence of $P+1$ points $(\mathbf{p}_i)_{i=0,\dots,P}$ in the plane $\mathbb{R}^2$ with associated parameter values $(t_i)_{i=0,\dots,P}$ satisfying $t_i < t_{i+1}$ be given. If the parameters t_i are unknown yet, then they can be estimated from the data, cf. [9, pp. 201/202]. Additionally, the $K+1$ knots $(\tau_i)_{i=0,\dots,K}$ of the approximating spline curve are assumed to be known ($\tau_i < \tau_{i+1}$, $\tau_0 = t_0$, $\tau_K = t_P$). Some of the knots are marked as the desired inflections of the spline curve. They are denoted by $(\tau_{w(i)})_{i=1,\dots,W-1}$ where $w(i) < w(i+1)$. Additionally we set $w(0) = 0$ and $w(W) = K$. Finally, the curvature signs $\sigma_i \in \{+1, -1\}$ of the spline segment $t \in [\tau_{w(i)}, \tau_{w(i+1)}]$ have to be specified. Of course, the curvature signs of adjacent segments should be different: $\sigma_i \cdot \sigma_{i+1} = -1$.
The given data is to be approximated by a C^l B–spline curve ($l = 1, 2$) of degree d ($d > l$),

$$\mathbf{x}(t) = \sum_{j=0}^{D} \mathbf{d}_j \cdot N_j^d(t) \qquad t \in [\tau_0, \tau_K] \tag{1}$$

($D = d \cdot k - (k-1) \cdot l$) with the unknown control points $(\mathbf{d}_j)_{j=0,\dots,D}$ in $\mathbb{R}^2$. The B–spline basis functions $(N_j^d(t))_{j=0,\dots,D}$ are defined over the knot vector

$$(\underbrace{\tau_0, \dots, \tau_0}_{d+1 \text{ times}}, \underbrace{\tau_1, \dots, \tau_1}_{d-l \text{ times}}, \underbrace{\tau_2, \dots, \tau_2}_{d-l \text{ times}}, \dots \underbrace{\tau_{K-1}, \dots, \tau_{K-1}}_{d-l \text{ times}}, \underbrace{\tau_K, \dots, \tau_K}_{d+1 \text{ times}}). \tag{2}$$

For more information concerning B–spline curves we refer the reader to the textbooks [4, 9]. The construction described below will ensure that the approximating spline curve (1) possesses inflection points only for $t = \tau_{w(i)}$ ($i = 1, \dots, W-1$). The curvature sign of the spline segments $t \in [\tau_{w(i)}, \tau_{w(i+1)}]$ in between will be equal to the specified value σ_i ($i = 1, \dots, W$).

2.2 The method

The construction of the spline curve consists of three steps. At first, we construct a reference curve which possesses the specified inflection points and curvature signs. With the help of this curve we then generate an appropriate set of linear constraints ensuring the desired shape properties. Finally, the control points of the approximating curve are found by solving a constrained optimization problem.

(i) Construction of the reference curve

The reference curve $\mathbf{y}(t)$ is chosen as a C^l B–spline curve of degree $l+1$ which possesses the desired shape. Two possible construction of the reference curve are described in [10].
For the first construction we prescribe the direction vectors of the legs of the B–spline control polygon, whereas the lengths are found by minimizing the least–square sum. The directions of the legs which correspond to spline segments having constant curvature signs result from uniform rotations, This ensures the desired shape properties.
For the second construction we use a B–spline curve which is defined over an appropriate subset of the knot sequence $(\tau_i)_{i=0,\dots,K}$. The curve is found by least–square approximation of the data. Its knots are chosen such that no undesired inflections occur. This is achieved by systematically deleting knots from the original knot sequence.
Whereas the first construction is guaranteed to yield a curve which has the desired shape properties, the second construction only ensures that no undesired inflections occur. But in most cases of practical interest the result of the second construction can be expected to have also the desired curvature signs. The second construction generally yields a better approximation of the data than the first one, and its result is more suitable for the following constructions.
In the case of C^2 spline curves ($l=2$), the construction of the reference curve additionally ensures that $\ddot{\mathbf{x}}(\tau_{w(i)}) = \vec{\mathbf{0}}$ holds for all inflection knots, $i=1,\dots,W-1$. This is necessary for the generation of the shape constraints in the second step.
In order to find a representation which is defined over the specified knot vector (2), the degree of the reference curve is elevated from $l+1$ to d.

(ii) Generation of linear shape constraints

The generation of the linear constraints ensuring the desired curvature distribution is based on the following observation:

Lemma. *Consider a Bézier curve $\mathbf{y}(t)$ of degree d with control points $(\mathbf{b}_i)_{i=0,\dots,d}$ in $\mathbf{R}^2$ (see [9, p. 120]). If two vectors $\mathbf{u},\mathbf{v} \in \mathrm{I\!R}^2 \setminus \{\vec{\mathbf{0}}\}$ exist such that the $4d-2$ inequalities* [1]

$$\begin{array}{lll} [\vec{\mathbf{u}}, \Delta^1\mathbf{b}_i] \geq 0 & [\Delta^1\mathbf{b}_i, \vec{\mathbf{v}}] \geq 0 & (i=0,\dots,d-1) \\ [\vec{\mathbf{v}}, \Delta^2\mathbf{b}_i] \geq 0 \quad (\textit{resp.} \ \leq 0) & [\Delta^2\mathbf{b}_i, -\vec{\mathbf{u}}] \geq 0 \quad (\textit{resp.} \ \leq 0) & (i=0,\dots,d-2) \end{array} \tag{3}$$

for the first and second differences vectors $\Delta^1\mathbf{b}_i = \mathbf{b}_{i+1} - \mathbf{b}_i$ and $\Delta^2\mathbf{b}_i = \Delta^1\mathbf{b}_{i+1} - \Delta^1\mathbf{b}_i$ of the control points hold, then the curve segment $t \in [0,1]$ is convex and the curvature is non–negative (resp. non–positive).

The conditions of the Lemma ensure that the wedges spanned by the first and second difference vectors of the control points can be separated as shown in Figure 1. The first and second derivatives of the curve are contained in these wedges, hence the curve has non–negative (resp. non–positive) curvature.
As shown in [10], any Bézier curve with non–positive (resp. non–negative) curvature can be subdivided into a finite number of segments such that the convexity of the segments can be guaranteed with the help of the above linear inequalities. This also applies to curves with vanishing curvature at the segment end points $t_0 = 0$ or 1, provided that the second

[1]The abbreviation $[.,.]$ denotes the exterior product $[\vec{\mathbf{s}}, \vec{\mathbf{t}}] = s_1 \cdot t_2 - s_2 \cdot t_1$ of two vectors $\vec{\mathbf{s}}, \vec{\mathbf{t}} \in \mathrm{I\!R}^2$.

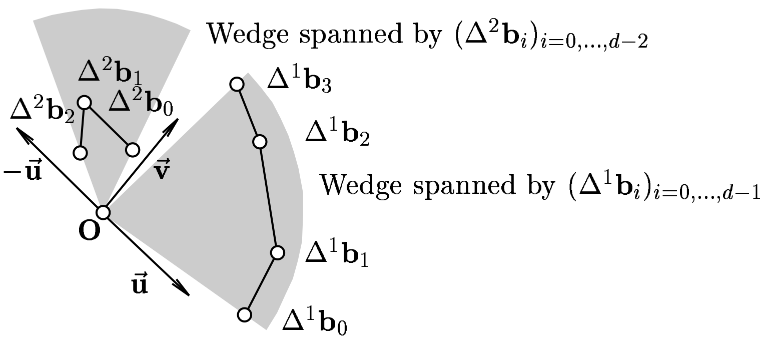


Fig. 1 Convexity conditions for a quartic Bézier curve.

derivative at the segment end points vanishes, $\ddot{\mathbf{y}}(t_0) = \vec{\mathbf{0}}$, and $[\dot{\mathbf{y}}(t_0), \dddot{\mathbf{y}}(t_0)] > 0$ (resp. < 0) holds. This is important for the generation of convexity constraints in the presence of desired inflection points.
Based on these observations, an algorithm has been formulated in [10] which generates a set of linear sufficient convexity constraints with the help of the reference curve. It yields two (possibly empty) sets of inequalities $\mathcal{I}$ and equalities $\mathcal{E}$ for the components of the unknown control points $(\mathbf{d}_i)_{i=0,\ldots,D}$. In order to generate these constraints, the reference curve is subdivided into an appropriate number of subsegments, and then the vectors $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$ occurring in (3) are computed from the bounds of the wedges spanned by the first and second differences of the resulting control points.

(*iii*) Computing the curve

The unknown control points of the approximating B–spline curve (1) are found by minimizing the least–square sum

$$F(\mathbf{d}_0, \ldots, \mathbf{d}_D) = \sum_{i=0}^{P} \|\mathbf{x}(t_i) - \mathbf{p}_i\|^2 \tag{4}$$

subject to the linear shape constraints $\mathcal{I}$ and $\mathcal{E}$. This quadratic programming problem (see [1]) is solved using the method of active set as described in [5]. The control points of the reference curve serve as initial point for the optimization.
As most algorithms for solving quadratic programming problems have difficulties with degenerate situations, it is of crucial importance to avoid redundancies of the sets of constraints as far as possible, see [10].
In most applications, the number of data points can be expected to be much bigger than the number D of B–spline control points, and also the distribution of the parameters t_i over the spline segments will be more or less uniform. Then the quadratic part of the function (4) is positively definite, and therefore the solution of the quadratic programming problem will be unique. Otherwise it is possible to add a “tension term” (cf. [9, p. 98] in order to achieve regularity.
If the approximating curve obtained from the quadratic programming is not satisfying, then the whole procedure can be iterated. New linear shape constraints can be generated by using the first curve as the new reference curve. This should yield constraints which are more suitable for the approximation of the given data.

2.3 Examples

In this section we will illustrate the capabilities of our method by an example. We computed two sets each consisting of 8 cubic C^2 B–spline curves (defined over appropriate knot vectors) which approximate the 296 data points (marked by crosses) of Figure 2. The

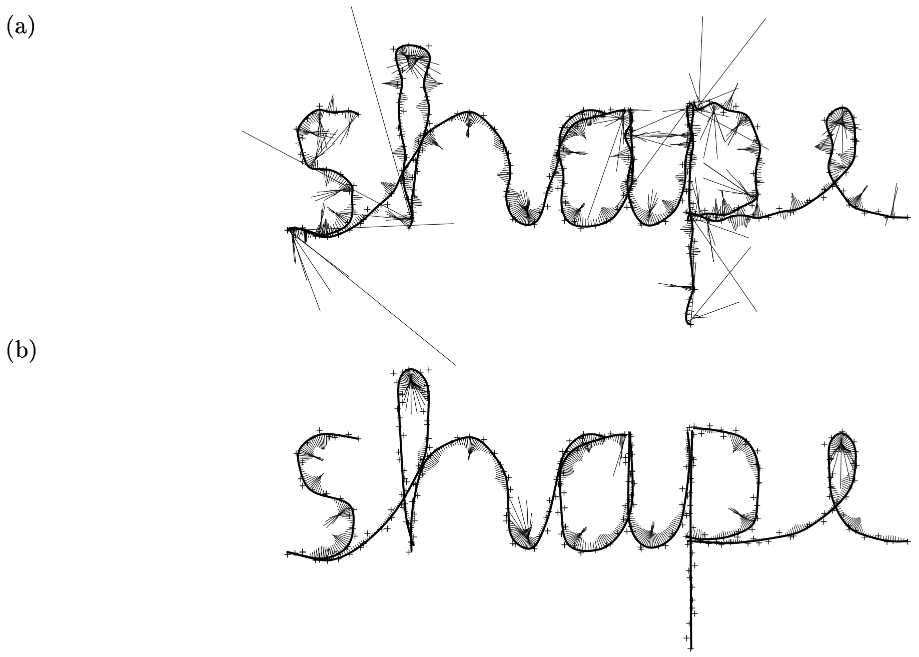


Fig. 2 Shape–preserving least–square approximation. The approximating curves have been computed (a) without constraints, (b) with shape constraints.

curves in Figure 2a have been computed without any shape constraints. In contrast with this, the curves in Figure 2b have been obtained after two iterations of the above–described method. In order to analyze the shape of the curves, the normals of the spline curves have been drawn (thin black lines), whereby their magnitudes have been chosen proportional to the curvature at the curve points.

3 Interpolation of the contour curves

Now we outline a method for constructing a surface by sectional curvature–preserving interpolation of a given set of contour curves. A more detailed description of the scheme will be presented in [11].

3.1 Problem statement

Let a sequence of $C+1$ open contours $(\mathbf{x}_i(t))_{i=0,\ldots,C}$ in $\mathbb{R}^2$ with associated z–coordinates (heights) $z_0 < z_1 < \ldots < z_C$ be given. Each contour is described by a C^l B–spline curve $(l = 1, 2)$

$$\mathbf{x}_i(t) = \sum_{j=0}^{D_i} \mathbf{d}_{i,j} \cdot N^d_{i,j}(t) \qquad t \in [0,1], \quad i = 0, \ldots, C, \tag{5}$$

of degree d with D_i+1 control points $(\mathbf{d}_{i,j})_{j=0,\dots,D_i}$ in $\mathbb{R}^2$. The B–spline basis functions $(N^d_{i,j}(t))_{j=0,\dots,D_i}$ of the i–th contour are defined over a certain knot vector $\mathcal{T}_i$. The knot vectors may be different, but they are assumed to have the $(d+1)$–fold boundary knots 0 and 1. The construction described below is based on the following three assumptions.

(1) The *matching problem* (cf. [14]) has already been solved; points on adjacent contours $\mathbf{x}_i(t)$, $\mathbf{x}_{i+1}(t)$ which share the parameter value t correspond to each other.

(2) If a contour curves possesses inflections, then the parameters of the inflection points are also knots of the spline curve. Moreover, in the C^2–case $(l=2)$ we have $\ddot{\mathbf{x}}_i(t_{\text{infl}})=\vec{\mathbf{0}}$ at these points. (Note that this assumption is automatically fulfilled if the contour curves have been generated using the algorithm described in the first part of the paper.) No higher order flat points of the contour curves are assumed to exist.

(3) The shape of the contour curves corresponds with the shape of the control polygons. More precisely, if the curvature of a segment $t\in[t_a,t_b]$ of one of the contour curves $\mathbf{x}_i(t)$ has only non–negative (non–positive) values, then also the polygon formed by all control points acting on this segment has only non–negative (non–positive) angles between adjacent difference vectors of control points. Additionally, these angles are smaller [2] than $\pi/(d-1)$. The validity of this assumption can always be achieved by inserting additional knots.

We want to construct a C^l spline surface $\mathbf{y}(z,t)$ with $(t,z)\in[0,1]\times[z_0,z_C]$ which interpolates the given contour curves, i.e.,

$$\mathbf{y}(z_i,t)\equiv\mathbf{x}_i(t) \qquad \text{holds for} \qquad i=0,\dots,C. \tag{6}$$

The interpolating surface is to preserve the shape of the given contour curves. Consider two segments $t\in[t_a,t_b]\subseteq[0,1]$ of adjacent contours $\mathbf{y}(z_i,t)=\mathbf{x}_i(t)$ and $\mathbf{y}(z_{i+1},t)=\mathbf{x}_{i+1}(t)$. If the curvature of both segments has only non–negative (resp. non–positive) values, then also the curvature of any interpolating contour segment $\mathbf{y}(z^*,t)$ in between $(z^*\in[z_i,z_{i+1}]$ is to be non–negative (resp. non–positive). If this assertion is true for all corresponding segments of adjacent contours, then the interpolating surface is said to preserve the sectional curvature of the data. This notion has been introduced by Kaklis and Ginnis [12].

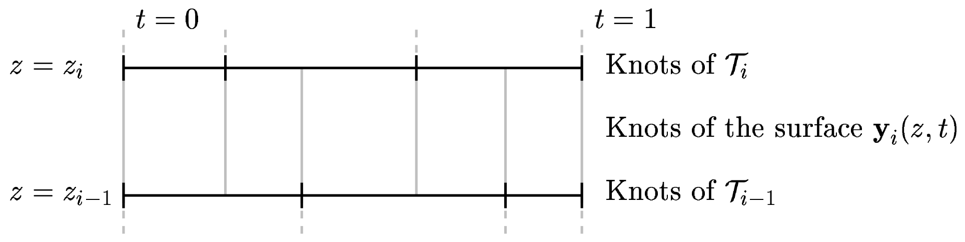
3.2 Surface definition and continuity constraints

The interpolating surface is composed from tensor–product B–spline surfaces of degree (d,n). Whereas d denotes the degree of the contour curves, the degree of the parameter lines $t=\text{const}$ is equal to $n\geq 2\cdot l+2$. For $z\in[z_{i-1},z_i]$, the interpolating surface is described by

$$\mathbf{y}_i(z,t)=\sum_{j=0}^{\hat{D}_i}\hat{\mathbf{d}}_{i,j}(z)\cdot\hat{N}^d_{i,j}(t), \qquad (z,t)\in[z_{i-1},z_i]\times[0,1], \qquad i=1,\dots,C. \tag{7}$$

The B–spline basis functions $(\hat{N}^d_{i,j}(t))_{j=0,\dots,\hat{D}_i}$ are defined over the union of the knot vectors $\mathcal{T}_{i-1}$ and $\mathcal{T}_I$, see Figure 3. The $\hat{D}_i+1$ control points of $(\hat{\mathbf{d}}_{i,j}(z))_{j=0,\dots,\hat{D}_i}$ of the interpolating

[2]Under this assumption, the number of inflections of the curve is bounded by the number of inflections of the control polygon, see [6]. Of course, this also applies to any segment of the curves.



$t = 0$
$t = 1$
$z = z_i$
$z = z_{i-1}$
Knots of $\mathcal{T}_i$
Knots of the surface $\mathbf{y}_i(z,t)$
Knots of $\mathcal{T}_{i-1}$

Fig. 3 The knots of interpolating surface.

contours depend on the height z. They run along Bezier curves of degree n with parameter z,

$$\hat{\mathbf{d}}_{i,j}(z) = \sum_{k=0}^{n} \mathbf{c}_{i,j,k} \cdot B_k^n(\tfrac{z-z_{i-1}}{z_i - z_{i-1}}) \qquad z \in [z_{i-1}, z_i],\ j = 0, \dots, \hat{D}_i,\ i = 1, \dots, C. \tag{8}$$

Their control points $\mathbf{c}_{i,j,k} \in \mathrm{I\!R}^2$ are unknown yet, whereas $B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i}$ are the Bernstein polynomials.
From the interpolation conditions (6) we immediately obtain the boundary control points $(\mathbf{c}_{i,j,0})_{j=0,\dots,\hat{D}_i}$ and $(\mathbf{c}_{i,j,n})_{j=0,\dots,\hat{D}_i}$ of the surface defined by (7) and (8) with the help of the knot insertion algorithm, cf. [9]. In addition, the surface has to fulfill the continuity constraints

$$\left(\frac{\partial}{\partial z}\right)^{\lambda} \mathbf{y}_i(z,t) \Bigg|_{z=z_i} \equiv \left(\frac{\partial}{\partial z}\right)^{\lambda} \mathbf{y}_{i+1}(z,t) \Bigg|_{z=z_i} \qquad \lambda = 1, \dots, l,\ t \in [0,1], \tag{9}$$

$i = 1, \dots, C-1$. The continuity constraints leads to a set of linear equations involving the control points $(\mathbf{c}_{i,j,r})_{j=0,\dots,\hat{D}_i}$ and $(\mathbf{c}_{i+1,j,n-r})_{j=0,\dots,\hat{D}_{i+1}}$, $r = 0, \dots, l$. Note that the knot vectors of adjacent spline surfaces $\mathbf{y}_i(z,t)$ may be different. Thus, the continuity constraints include certain not–a–knot–type conditions for the control points $\mathbf{c}_{i,j,k}$. Resulting from $n \geq 2 \cdot l + 2$, each coefficient $\mathbf{c}_{i,j,k}$ is subject to at most one set of constraints (9).

3.3 The method

The construction of the interpolating surface consists of three steps. At first we generate sufficient linear constraints for preserving the sectional curvature of the contour curves. Then we choose an appropriate quadratic objective function. In the last step we find an initial point for the optimization and construct the control points of the surface by solving a quadratic programming problem.

(i) Linear sufficient shape constraints

We consider one segment $\mathbf{y}_i(z,t)$ of the interpolating surface. Resulting from the assumptions formulated in (3), it is sufficient to consider the shape of the control polygons of the adjacent contour curves $\mathbf{x}_{i-1}(t) = \mathbf{y}_i(z_{i-1}, t)$ and $\mathbf{x}_i(t) = \mathbf{y}_i(z_i, t)$. If three successive contour control points $\hat{\mathbf{d}}_{i,j}(z)$, $\hat{\mathbf{d}}_{i,j+1}(z)$, $\hat{\mathbf{d}}_{i,j+2}(z)$ (cf. Figure 4) satisfy the condition

$$\Big[\, \hat{\mathbf{d}}_{i,j+1}(z^*) - \hat{\mathbf{d}}_{i,j}(z^*),\ \hat{\mathbf{d}}_{i,j+2}(z^*) - \hat{\mathbf{d}}_{i,j+1}(z^*) \,\Big] \geq 0 \qquad \text{(resp.} \quad \leq 0) \tag{10}$$

for $z^* = z_{i-1}$ and $z^* = z_i$, then we want to ensure that this inequality is true for all $z^* \in [z_{i-1}, z_i]$. (Note that the value of the left–hand side of the inequality for $z^* = z_{i-1}$ and

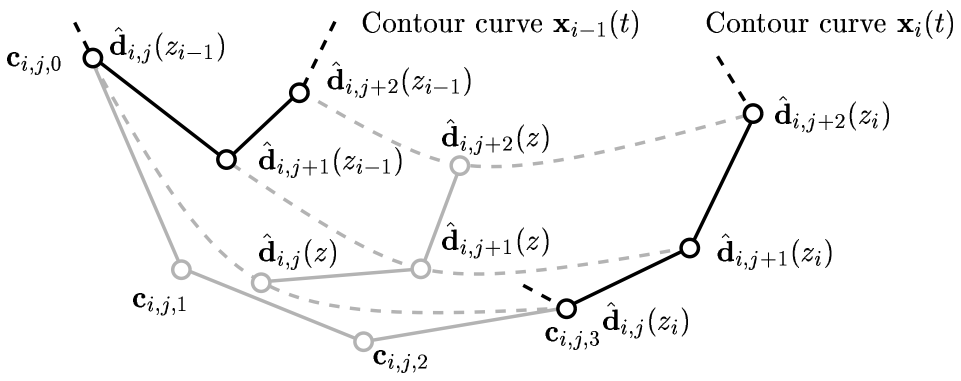

Contour curve $\mathbf{x}_{i-1}(t)$
Contour curve $\mathbf{x}_i(t)$
$\mathbf{c}_{i,j,0}$
$\hat{\mathbf{d}}_{i,j}(z_{i-1})$
$\hat{\mathbf{d}}_{i,j+2}(z_{i-1})$
$\hat{\mathbf{d}}_{i,j+1}(z_{i-1})$
$\hat{\mathbf{d}}_{i,j+2}(z)$
$\hat{\mathbf{d}}_{i,j+2}(z_i)$
$\hat{\mathbf{d}}_{i,j}(z)$
$\hat{\mathbf{d}}_{i,j+1}(z)$
$\hat{\mathbf{d}}_{i,j+1}(z_i)$
$\mathbf{c}_{i,j,1}$
$\mathbf{c}_{i,j,3}$
$\hat{\mathbf{d}}_{i,j}(z_i)$
$\mathbf{c}_{i,j,2}$

Fig. 4 Three adjacent contour control points ($n = 3$).

$z^* = z_i$ results from the interpolation of the given contours, cf. (6).) In addition to (10), the contour control points are to fulfill

$$-\tfrac{\pi}{d-1} \leq \sphericalangle \Big(\hat{\mathbf{d}}_{i,j+1}(z) - \hat{\mathbf{d}}_{i,j}(z),\ \hat{\mathbf{d}}_{i,j+2}(z) - \hat{\mathbf{d}}_{i,j+1}(z) \Big) \leq \tfrac{\pi}{d-1} \text{ for } z \in [z_{i-1}, z_i] \quad (11)$$

because then the number of inflections of the contour curves is bounded by that of the control polygon, see [6]. Similar to the construction outlined in Section 2.2(*ii*), it is possible to derive linear sufficient constraints ensuring the inequalities (10) and (11), see [11].

(*ii*) **An objective function**

In addition to the constraints ensuring the desired shape of the contours we need an appropriate objective function in order to compute the unknown control points $\mathbf{c}_{i,j,k}$ of the interpolating surface. We want to achieve that the transition between control polygons of adjacent contours is as smooth as possible. So we construct the objective function by summing up the terms

$$\sum_{r=0}^{S} \left\| \left[\hat{\mathbf{d}}_{i,j+1}(\tfrac{S-r}{S} \cdot z_{i-1} + \tfrac{r}{S} \cdot z_i) - \hat{\mathbf{d}}_{i,j}(\tfrac{S-r}{S} \cdot z_{i-1} + \tfrac{r}{S} \cdot z_i) \right] - \mathbf{q}_{i,j,r} \right\|^2 \quad (12)$$

for each pair of adjacent contour control points $(\hat{\mathbf{d}}_{i,j}, \hat{\mathbf{d}}_{i,j+1})_{i=0,\ldots,\hat{D}_i-1}$ $(i = 1, \ldots, C)$, whereby the $S + 1$ (S large) points $(\mathbf{q}_{i,j,r})_{r=0,\ldots,S}$ are sampled equidistantly from the spiral with center $\mathbf{O}$ which interpolates the difference vectors $\hat{\mathbf{d}}_{i,j+1}(z^*) - \hat{\mathbf{d}}_{i,j+1}(z^*)$ of adjacent contour control points for $z^* = z_{i-1}$ and $z^* = z_i$. For the trajectories of the first and last control point we add a "tension term" (cf. [9, p. 98]) like

$$\int_{z_{i-1}}^{z_i} \| \frac{\mathrm{d}^2}{\mathrm{d}z^2} \, \hat{\mathbf{d}}_{i,j_0}(z) \|^2 \, \mathrm{d}z \qquad (j_0 = 0, \hat{D}_i) \quad (13)$$

to the objective function. Note that without this term the objective function would depend only on difference vectors $\mathbf{c}_{i,j+1,k} - \mathbf{c}_{i,j,k}$ of control points, and hence the minimum would not be unique. The objective function is a quadratic functions of the components of the unknown control points $\mathbf{c}_{i,j,k}$.

(iii) Computing the control points

The control points $\mathbf{c}_{i,j,k}$ of the interpolating surface are found by minimizing the quadratic objective function subject to the linear shape constraints obtained from step (i). This quadratic programming problem is again solved with the help of the method of active set as described in [5]. The initial solution is found using the simplex algorithm. It is advantageous to start with an initial solution which is close to the minimum of the unconstrained objective function, see [11].
The linear shape constraints obtained from the first step are not guaranteed to provide feasible solutions. In most cases of practical interest the feasible region (in the linear space of control points $\mathbf{c}_{i,j,k}$) will be non–empty. Moreover it can be shown that feasible solutions always exist if the degree n of the parameter lines $t = \text{const}$ is chosen high enough [11].

3.4 An example

As an example we show in Figure 5 the interpolation of four given contour curves with equidistant heights $z_0, \ldots, z_3$. The four given contours are described by quadratic B–spline curves defined over different vectors. The degree of the three interpolating C^1 B–spline surface patches $\mathbf{y}_1(z,t), \ldots, \mathbf{y}_3(z,t)$ is equal to $(2,3)$.

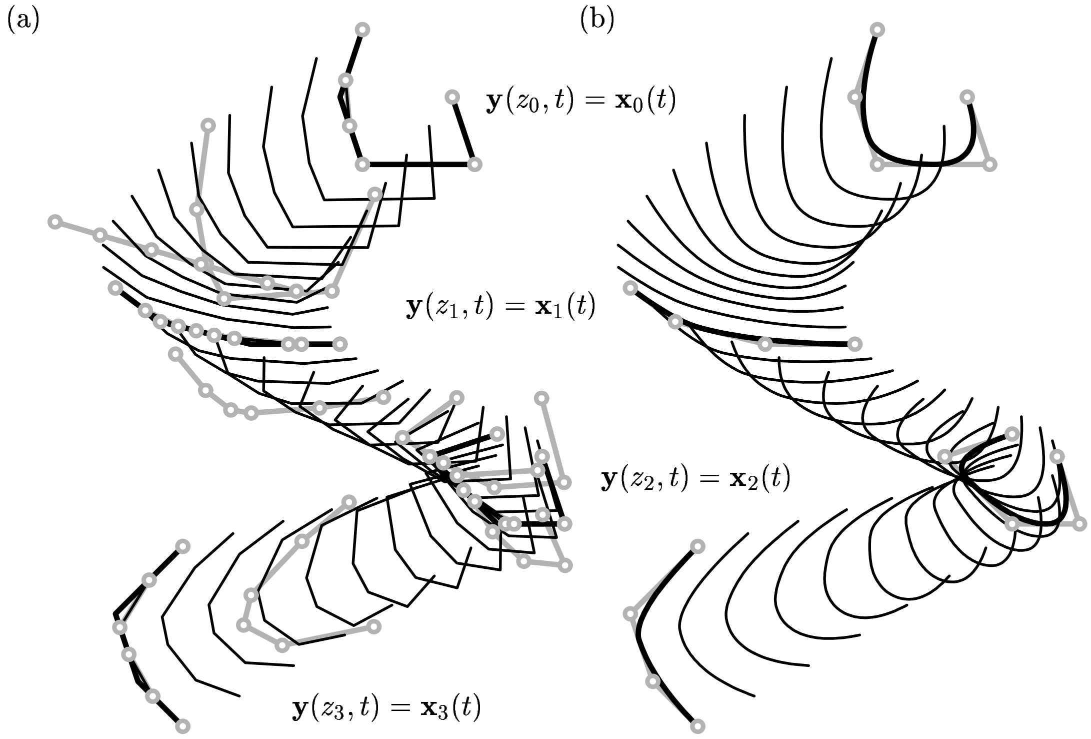

(a)
(b)
$\mathbf{y}(z_0,t) = \mathbf{x}_0(t)$
$\mathbf{y}(z_1,t) = \mathbf{x}_1(t)$
$\mathbf{y}(z_2,t) = \mathbf{x}_2(t)$
$\mathbf{y}(z_3,t) = \mathbf{x}_3(t)$

Fig. 5 Sectional curvature–preserving interpolation of four contours. The interpolating control polygons (a) and the contour curves (b).

The left figure shows the control polygons $\hat{\mathbf{d}}_{i,j}(z)$ of the interpolating contours and the control points $\mathbf{c}_{i,j,k}$ of their trajectories (in grey). The interpolating contour curves and the control polygons of the four given contours (in grey) have been drawn in Figure 5b. The thicker black lines in both figures indicate the given data.

References

[1] Boot, J. C. G., *Quadratic Programming.* Rand McNally, Chicago 1964.

[2] Dierckx, P., *An algorithm for cubic spline fitting with convexity* constraints. Computing 24 (1980), 349–371.

[3] Dierckx, P., *Curve and Surface Fitting with Splines.* Clarendon Press, Oxford 1993, 119–134.

[4] Farin, G., *Curves and Surfaces for Computer Aided Geometric Design.* Academic Press, Boston 1993.

[5] Fletcher, R., *Practical Methods of Optimization.* John Wiley & Sons, Chichester 1991 (2nd ed.).

[6] Goodman, T. N. T., *Inflections on curves in two and three dimensions.* Computer Aided Geometric Design 8 (1991), 37–50.

[7] Goodman, T. N. T., and Ong, B. H., *Shape preserving interpolation by curves in three dimensions.* These proceedings.

[8] Goodman, T. N. T., and Unsworth, K., *Shape preserving interpolation by curvature continuous parametric curves.* Computer Aided Geometric Design 5 (1988), 323–340.

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

[10] Jüttler, B., *Shape preserving least–square approximation by polynomial parametric spline curves.* University of Dundee, Applied Analysis Report 965 (1996), submitted to Computer Aided Geometric Design.

[11] Jüttler, B., *Sectional curvature preserving approximation of contour lines.* in preparation.

[12] Kaklis, P. D., and Ginnis, A. I., *Sectional–Curvature Preserving Skinning Surfaces.* Technical Report, National Technical University of Athens, Dept. of Naval Architecture and Marine Engineering, 1994.

[13] Schumaker, L. L., *Reconstructing 3D objects from cross–sections.* in: Dahmen, W., Gasca, M., and Micchelli, C. A. (eds.), *Computation of Curves and Surfaces.* Kluwer, Dordrecht 1990, 275–309.

[14] Unsworth, K., *Recent developments in surface reconstruction from planar cross-sections.* University of Dundee, Computer Science Report 94/03 (1994), to appear in the proceedings of the Conference on CAGD held in Penang, July 1994 (to be published as a volume of the Annals of Numerical Mathematics).