Constrained Variational Triangular Surface Design

Remco C. Veltkamp *† Wieger Wesselink *
Huub van de Wetering *

* Eindhoven University of Technology, Dept. Computing Science, Eindhoven
† CWI, Dept. Interactive Systems, Amsterdam

Abstract

In Computer Aided Design, esthetic demands often require that surfaces be designed as smooth as possible while functional requirements constrain the surface to interpolate certain data. This paper introduces new constrained optimization techniques for triangular surfaces. The surfaces are specified through interpolation constraints consisting of points, curves, and normals. Non-linear optimization techniques are used to minimize an energy functional in order to make the surface smooth.

Keywords: CAD, surface modeling, constrained optimization.

1. Introduction

Variational surfaces minimize a functional that is typically modeling the energy of the surface. A functional is a quantity that depends on a function (in this case the surface) rather than on a continuous variable as in the case of a function. The calculus of variations is about finding the function (surface) that results in the functional becoming a minimum. Many different models of the surface energy may be used [HS91]. Section 2.2 discusses some of these.

A recent development in surface construction is the combination of modeling through interpolation constraints and variational optimization. Such constrained optimization approaches result in surfaces that interpolate prescribed data, such as the position of a point on the surface, or the normal of the surface at a point, while minimizing some surface energy [CG91] [Kal93] [MS92] [WW92]. This results in a powerful way of modeling: certain aspects of the shape of the surface are exactly prescribed, and the rest of the surface adopts a 'natural' shape. Section 2.4 discusses some types of interpolation constraints, and the way they are used by others.

Tensor product surfaces are widely used in surface design. Indeed, many surfaces lend themselves naturally to a tensor product representation, but surfaces exist where this is not the case. The theory of tensor product patches requires that all data have a rectangular geometry and that parameterizations of opposite curves be similar. This is the case for some surfaces, e.g. the trunk lid of a car, but not for others, e.g. the interior car body panels [Far86]. For those more complicated surfaces no distinct rectangular structure is apparent, and one should have to invest considerable time and effort to impose

*Correspondence address: Eindhoven University of Technology, Dept. Computing Science, P.O. Box 513, 5600 MB Eindhoven, The Netherlands. E-mail: {remco,wieger,wstahw}@win.tue.nl

a rectangular structure on such surfaces. Also for surface reconstruction from empirical data a rectangular structure is often absent and a triangular scheme is used. This paper introduces new constrained optimization techniques for triangular surfaces in Section 3. The computational methods used are given in Section 4.

We will use a triangular surface scheme in the Bézier formulation. The Bézier formulation is convenient to describe other polynomial schemes as well as to develop new schemes. (Bézier curves and surfaces were independently developed by de Casteljau at the Citroën and by Bézier at the Renault automobile company, but de Casteljau's development was never published, so that this curve and surface scheme was named after Bézier.) The Bézier formulation is often used because of the geometrical significance of the coefficients. The Bézier representation is given in Section 2.1.

2. Constraints and variational surfaces

The problem that we address in this paper is the following:

> Given a set of interpolation constraints, construct a triangular Bézier surface that satisfies the constraints. Given a surface energy functional, change that surface so as to minimize the functional, while satisfying the constraints.

Before introducing new techniques for constrained variational surface design in Section 3, we present the triangular Bézier surface formulation that we use, explain the use of energy functionals, and give a number of frequently used constraints.

2.1. Triangular Bézier patches

Analogous to the univariate Bernstein polynomials over an interval, the Bernstein polynomials of degree n over a non-degenerate triangle (A, B, C) are defined by

$$B^n_{i,j,k}(u,v,w) = \frac{n!}{i!j!k!} u^i v^j w^k, \;\; i+j+k=n, \;\; i,j,k \in \mathbb{N},$$

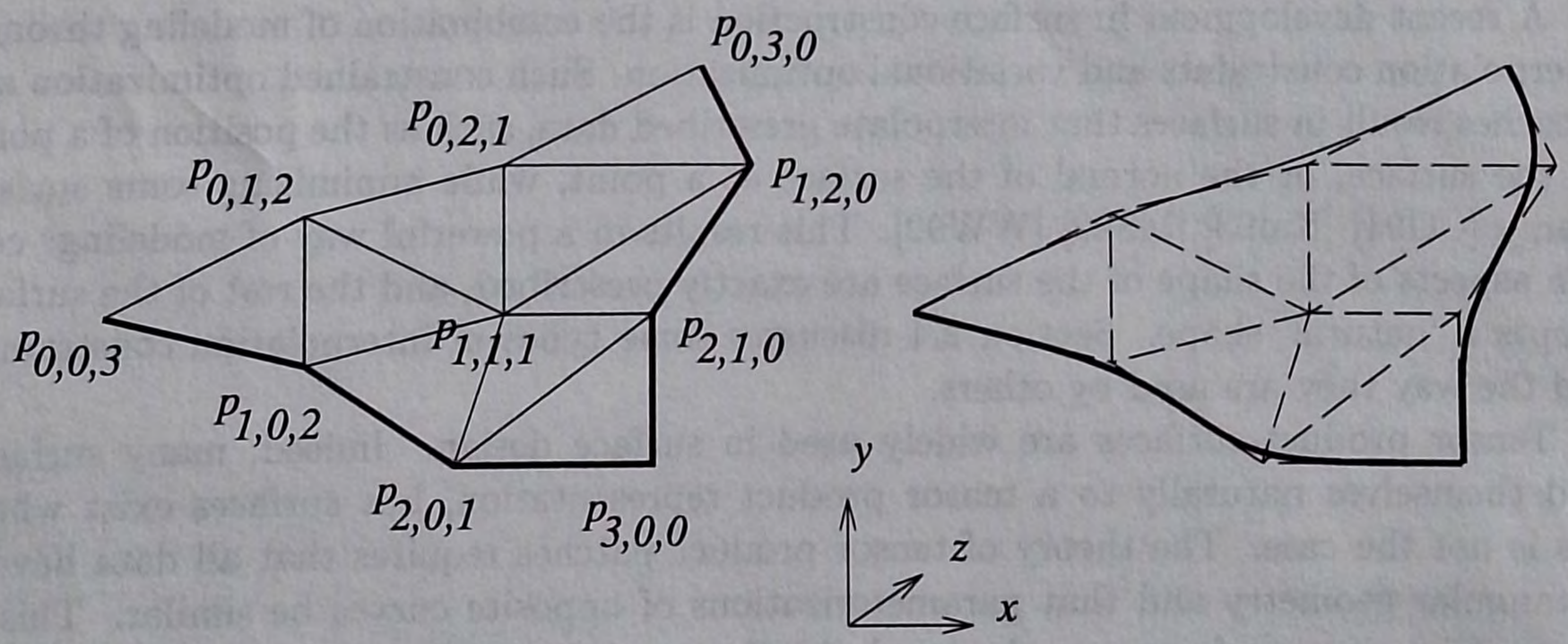


Figure 1: Left: Bézier control polyhedron. Right: corresponding cubic Bézier patch.

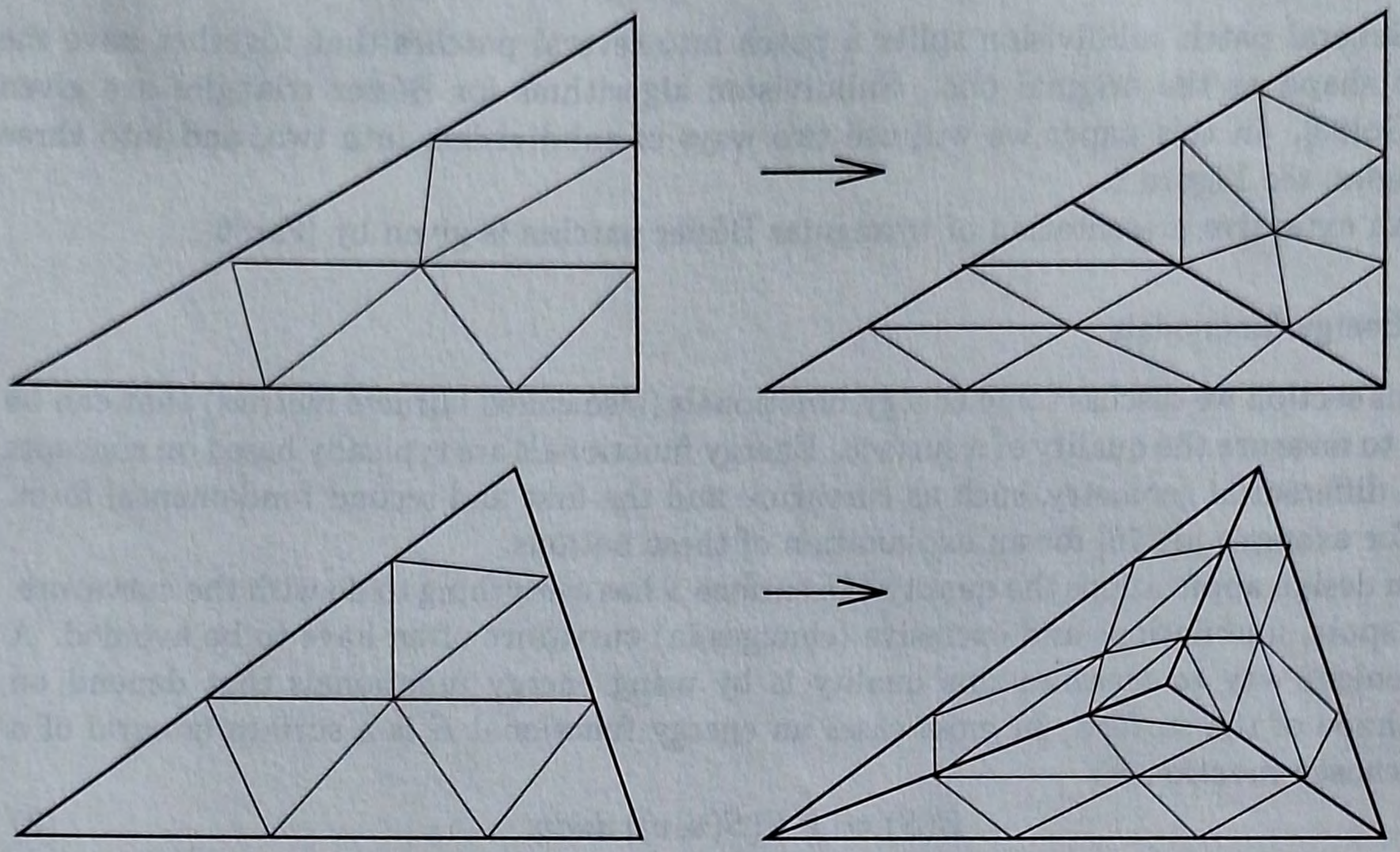


Figure 2: Two ways of patch subdivision used in this paper (here illustrated for cubic patches).

where u, v, and w, with $u + v + w = 1$, are barycentric coordinates with respect to (A, B, C). A parametric Bézier triangle is defined as follows:

$$T(u, v, w) = \sum_{i+j+k=n} p_{i,j,k} B^n_{i,j,k}(u, v, w), \tag{1}$$

where $p_{i,j,k}$ are points in space, called control points, forming an open control polyhedron. In the context of Bézier polyhedra, only $p_{n,0,0}$, $p_{0,n,0}$, and $p_{0,0,n}$ are called vertices. See Figure 1 for a control polyhedron and the corresponding *cubic* Bézier patch ($n = 3$).

If two patches join along a boundary curve, then the control points at the corresponding edges of the two control polyhedra should coincide. A continuous piecewise Bézier surface consists of a collection of abutting Bézier triangles.

The boundary of a patch consists of three univariate Bézier curves. For example for $w = 0$ we get $v = 1 - u$, and thus

$$T(u, v, 0) = B(u) = \sum_{i=0}^{n} p_{i,n-i,0} B^n_i(u),$$

with

$$B^n_i(u) = \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i}, \quad i \in \mathbb{N}.$$

So, a boundary curve of a patch is defined by the control points on the corresponding boundary of the control polyhedron.

The difference between two barycentric coordinates defines a direction d in the parameter space. The derivative vector of patch T in the direction $d = (d_1, d_2, d_3)$, $d_1 + d_2 + d_3 = 0$, is given by

$$\nabla_d T(u, v, w) = n \sum_{i+j+k=n-1} (d_1 p_{i+1,j,k} + d_2 p_{i,j+1,k} + d_3 p_{i,j,k+1}) B^{n-1}_{i,j,k}(u, v, w). \tag{2}$$

General patch subdivision splits a patch into several patches that together have the same shape as the original one. Subdivision algorithms for Bézier triangles are given by [Gol83]. In this paper we will use two ways of subdivision, into two, and into three triangles, see Figure 2.

An extensive presentation of triangular Bézier patches is given by [Far86].

2.2. Energy functionals

In this section we discuss some energy functionals (also called fairness metrics) that can be used to measure the quality of a surface. Energy functionals are typically based on concepts from differential geometry, such as curvature and the first and second fundamental form. See for example [dC76] for an explanation of these notions.

In design applications the quality of a surface S has everything to do with the curvature. Flat spots, undulations and excessive (changes in) curvature often have to be avoided. A convenient way to measure this quality is by using energy functionals that depend on the shape of the surface. In most cases an energy functional E is a surface integral of a well-chosen function F:

$$E(S) = \int F(S(u,v))\, dudv. \tag{3}$$

The general problem is stated as follows: find a surface S such that the functional (3) achieves a minimum over the parameter domain for which S is defined. (Note that a surface is a bivariate function, depending on u and v; the triangular Bézier patch as formulated above also depends on only two parameters, since $w = 1 - u - v$.) There are basically two types of energy functionals: the first type is based on geometric properties, and the second is based on physical properties.

[RR91] describes three of those 'geometric' functionals: the flattening, rounding and rolling metric. They depend on the Gaussian curvature K and the mean curvature H of the design surface S (see [dC76]). In all cases the energy functional E is defined as the area of a surface O that is derived from the original S:

$$E(S) = \int_O 1\, dO$$

For the flattening metric this surface O is defined as

$$O(u,v) = K(u,v)N(u,v),$$

where $N(u,v)$ is the unit normal vector to the surface $S(u,v)$. In [RR91] it is shown that the flattening metric penalizes large Gaussian curvature K, extreme changes in curvature along the lines of curvature, and large values of principal curvature. Minimizing with respect to this metric tends towards developable surfaces. (A developable surface is one that can be 'unrolled' onto a plane without stretching or distorting; cones and cylinders are good examples.)

For the rounding metric O is defined as

$$O(u,v) = S(u,v) + (H(u,v)/K(u,v))\, N(u,v),$$

and for the rolling metric

$$O(u,v) = \left(K(u,v) + H(u,v)^2\right) N(u,v)$$

The rounding metric penalizes unwanted flat regions and undulations in the surface and tends to spherical surfaces. The rolling metric tends to produce a cylindrical or conical shape.

The 'physical' functionals are mostly based on bend (or strain) and stretch energy. Stretch energy involves first order derivatives, and bend energy involves second order derivatives. Sometimes even higher order derivatives are used (see [HS91]). In [TPBF87] the following functionals are defined for stretch energy $E_{stretch}$ and bend energy E_{bend}:

$$E_{stretch}(S) = \int_S \|G\|^2 \, dS, \tag{4}$$

$$E_{bend}(S) = \int_S \|B\|^2 \, dS, \tag{5}$$

where B and G are the first and second fundamental forms of the surface S using the natural parameterization (see [dC76]), and $\|.\|$ is a suitably chosen matrix norm. Another common formulation of the bend energy is (see e.g. [LP88])

$$E_{bend}(S) = \int_S (k_1^2 + k_2^2) \, dS, \tag{6}$$

where k_1 and k_2 are the principal curvatures of the surface S. Naturally the bend- and stretch energy functionals are used to measure how much a surface is bent or stretched. The bend energy functional tends to distribute local bending over a large area, while the stretch energy functional tends to minimize the area of the surface.

2.3. Minimizing energy functionals

The above energy functionals all suffer from the fact that they are highly nonlinear in the surface's control points. This makes it computationally very difficult to determine the spline surface that minimizes such a functional. In order to simplify the minimization it is common practice to replace these functionals with approximations. The functional (6) is often approximated by (see e.g. [LP88]):

$$E_{bend}(S) = \int_\Omega (k_1^2 + k_2^2) \, dudv, \tag{7}$$

where Ω is the domain of the surface S in parameter space. A drawback of this approach is that (7) is parameterization dependent. It is only a good approximation of (6) if the Jacobian determinant $\left|\frac{\partial S}{\partial u} \times \frac{\partial S}{\partial v}\right|$ doesn't differ too much from 1. Formula (7) can be approximated even further by integrating only along some constant parameter lines, see [LP88].

In [CG91], [Kal93] and [WW92] the following approximations are given to the stretch energy (4) and the bend energy (5):

$$E_{stretch}(S) = \int_\Omega \left(\frac{\partial S}{\partial u}\right)^2 + 2\frac{\partial S}{\partial u} \cdot \frac{\partial S}{\partial v} + \left(\frac{\partial S}{\partial v}\right)^2 \, dudv, \tag{8}$$

$$E_{bend}(S) = \int_\Omega \left(\frac{\partial^2 S}{\partial u^2}\right)^2 + 2\left(\frac{\partial^2 S}{\partial u \partial v}\right)^2 + \left(\frac{\partial^2 S}{\partial v^2}\right)^2 \, dudv \tag{9}$$

(sometimes with some additional parameters). Both approximations depend on the parameterization.

In Section 3 we will use spline surfaces composed of triangular Bézier patches, each patch T_l being defined by Equation 1 as the weighted sum of the control points of the patch. The energy of the whole surface is the sum of the energies of the triangular patches:

$$E(S) = \sum_l E(T_l).$$

Let us denote the vector of control points of T_l with X_l. An important property of the expressions (8) and (9), applied to a single patch T_l, is that they can be written as a quadratic form in the control points of the patch:

$$E(T_l) = X_l^T A_{patch} X_l.$$

It is easily verified that the coefficients a_{ij} of the matrix A_{patch} are

$$a_{ij} = \int_\Omega \frac{\partial B_i}{\partial u}\frac{\partial B_j}{\partial u} + 2\frac{\partial B_i}{\partial u}\frac{\partial B_j}{\partial v} + \frac{\partial B_i}{\partial v}\frac{\partial B_j}{\partial v}\, dudv \tag{10}$$

for the stretch energy and

$$a_{ij} = \int_\Omega \frac{\partial^2 B_i}{\partial u^2}\frac{\partial^2 B_j}{\partial u^2} + 2\frac{\partial^2 B_i}{\partial u\partial v}\frac{\partial^2 B_j}{\partial u\partial v} + \frac{\partial^2 B_i}{\partial v^2}\frac{\partial^2 B_j}{\partial v^2}\, dudv. \tag{11}$$

for the bend energy.

Since patches that are adjacent in S have control points in common, we use a vector X of all distinct control points of S. The energy of the whole surface is then

$$E(S) = \sum_l X_l^T A_{patch} X_l = X^T A X, \tag{12}$$

where the coefficients of A are sums of coefficients of A_{patch}.

Because quadratic minimization problems can be solved very efficiently (see also Section 4) the approximations (8) and (9) are very appropriate for interactive applications. For our purposes we use a linear combination of (8) and (9) as energy functional.

2.4. Interpolation constraints

In this section we investigate some useful types of interpolation constraints and the way some of these are handled by others. Suppose we are given an initial surface $S_0(u, v)$, and we need to compute from this a new surface $S(u, v)$ that minimizes a given energy functional while satisfying a collection of interpolation constraints.

Point interpolation

The surface $S(u, v)$ interpolates a given point P if there exist parameter values u_0 and v_0 such that $S(u_0, v_0) = P$. For typical surface schemes such as Bézier and B-spline surfaces, this equation is linear in the surface's control points, but not in u_0 and v_0. It is therefore common practice to apply some heuristic for assigning values u_0 and v_0 to P.

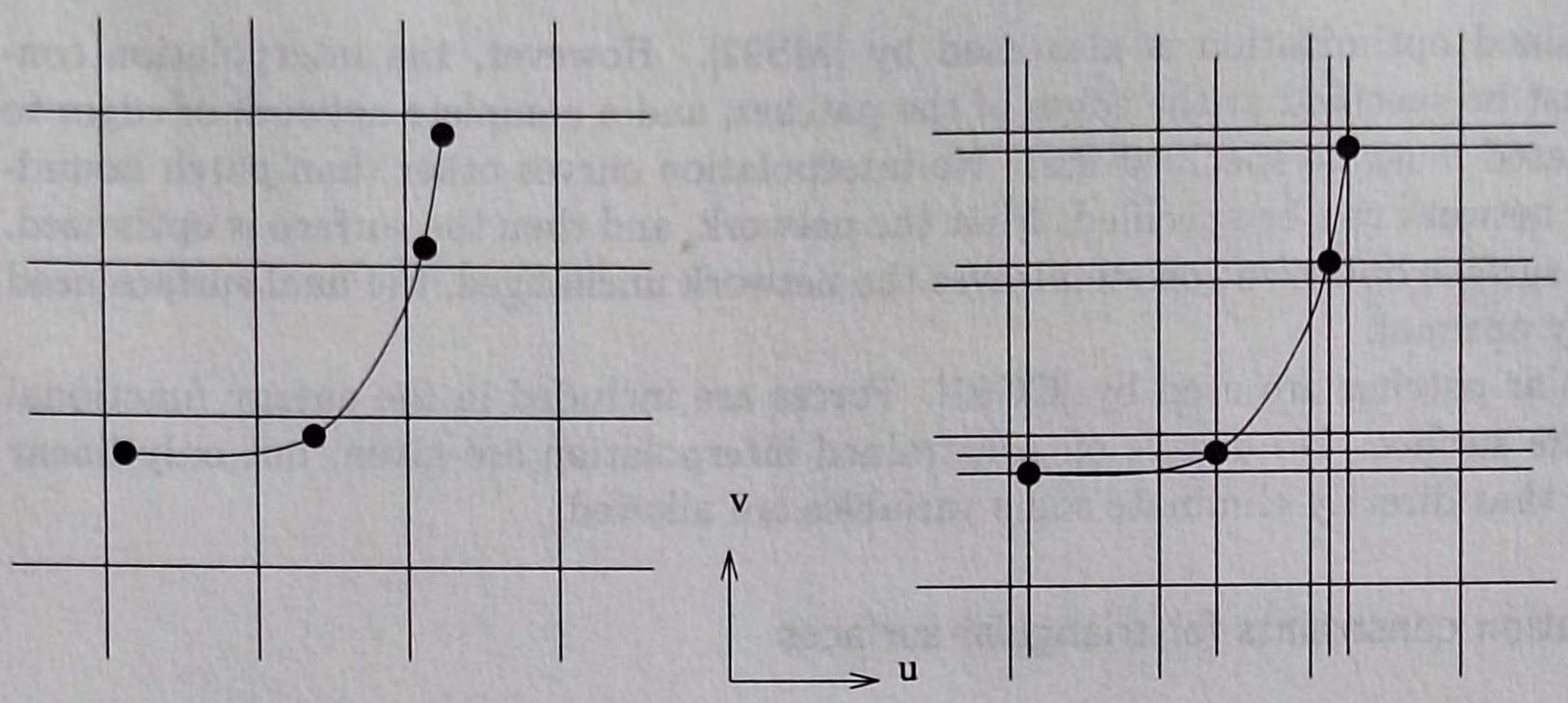


Figure 3: Refinement of a tensor product surface.

Curve interpolation

If surface $S(u,v)$ interpolates a polynomial curve $C(t)$, then there is a curve in the parameter space of S that maps onto $C(t)$, say curve $(u(t),v(t))$. Since $S(u(t),v(t)) = C(t)$ is a polynomial in t, $u(t)$ and $v(t)$ are also chosen to be polynomial. If the surface is of degree m in u and v, and u and v are of degree n in t, then curve $S(u(t),v(t))$ is of degree $2nm$ in t. Suppose $C(t)$ is of degree k and let $d = \max\{k, 2nm\}$. If $d+1$ points on $C(t)$ and $S(u(t),v(t))$ coincide, then the curves are equal.

Now suppose we want to construct a surface $S(u,v)$ that interpolates a given curve $C(t)$, i.e. the control points need to be set. Then the following procedure can be taken. Choose a mapping $u(t)$ and $v(t)$, with $n \geq k/2m$. Taking $2nm+1$ parameter values t_i and corresponding values u_i and v_i gives the equations $S(u_i,v_i) = C(t_i)$, which are the constraints on the control points of S so as to let the surface interpolate curve C. Actually, no explicit formulations of $u(t)$ and $v(t)$ are needed, but n must be known and the u_i and v_i must lie on a degree n polynomial.

Normal interpolation

If $S(u,v)$ must have a normal vector N at a point P for some $S(u_0,v_0) = P$, then two surface partial derivatives of different direction should be perpendicular to the normal vector. This is equivalent to the constraint that the two dot products of the normal vector and the derivatives are zero, which gives linear equations in the control points of the surface.

2.5. Related work

Tensor product B-splines are used in [Kal93]. In this case $S_0(u,v)$ is defined piecewise on ranges $[u_i,u_{i+1}] \times [v_i,v_{i+1}]$. The interpolated curve $C(t)$ is assumed to be a B-spline curve, piecewise defined on intervals $[t_i,t_{i+1}]$. Since it may be impossible for a single rectangular surface patch R_0 of S_0 to interpolate several curve segments $C(t)$, $t_i \leq t \leq t_{i+1}$, R_0 is refined at $u(t_i)$ and $v(t_i)$, see Figure 3. A major drawback of tensor product surfaces now becomes apparent: the subdivision of a single patch has a global effect in that it forces to split all patches of a whole row. The effect is that many tiny patches result far away from the place where extra patches are needed.

Constrained optimization is also used by [MS92]. However, the interpolation constraints must be specified at the edges of the patches, and a complete network of edges to be interpolated must be specified first. No interpolation curves other than patch boundaries in the network can be specified. First the network, and then the surface is optimized. Since their surface optimization step leaves the network unchanged, the final surface need not be truly optimal.

Triangular patches are used by [CG91]. Forces are included in the energy functional to model the surface. No details on constrained interpolation are given, but only linear constraints that directly eliminate some variables are allowed.

3. Interpolation constraints for triangular surfaces

In this section we introduce new techniques for constrained optimization of triangular surfaces. We use a piecewise surface S of triangular Bézier patches, each patch T_l being defined by Equation 1 as the weighted sum of the control points of the patch:

$$T_l(u,v,w) = \sum_{i+j+k=n} p_{i,j,k} B^n_{i,j,k}(u,v,w),$$

such that the surface satisfies given interpolation constraints, and minimizes the energy in Formulae (8) and (9). In this section we formulate the constraints in terms of the control points. Let us again denote the vector of control points of T_l with X_l. The objective is to generate linear constraint equations on the control points, i.e. a set of equations $F_l X_l = G_l$, and to provide good starting configurations for the minimization process. Since patches that are adjacent in S have control points in common, we use a vector X of all distinct control points of S, and gather all constraints into a global set of equations $F\,X = G$. The vector of control points X contains the variables that must be solved in the constrained minimization task: minimize $X^T A\,X$ (Equation 12) subject to the constraints $F\,X = G$. The way the control points are computed so as to minimize the energy functional while satisfying the constraints is discussed in Section 4.

3.1. Point interpolation

The point interpolation constraint equation $S(u_0, v_0) = P$ in terms of the Bézier control points is

$$\sum_{i+j+k=n} p_{i,j,k} B^n_{i,j,k}(u_0,v_0,w_0) = P,$$

where $u_0 + v_0 + w_0 = 1$. To make this constraint linear in the control points, we have to assign values to u_0 and v_0. The obvious heuristic choice for u_0 and v_0 is the value for which an initial surface $S_0(u_0, v_0)$ is closest to P, as done for tensor product splines by [Kal93]. It is shown for curves, however, that better results are obtained (i.e. a lower energy of the final curve) if the curve is made to approximate the interpolation point before assigning values to u_0 and v_0 [KGM91].

We therefore construct from S_0 a surface S_1 that approximates the interpolation point. First we determine the patch T_0 of S_0 that contains the point closest to P. Then we determine control point p_{i_0,j_0,k_0} of T_0 closest to P, and place it at the position of P. The control points at distance one from p_{i_0,j_0,k_0} in the control polyhedron, i.e. $p_{i,j,k}$ with $i = i_0 \pm 1$, $j = j_0 \pm 1$, $k = k_0 \pm 1$, $i+j+k = n$, are translated by the vector $(P - p_{i,j,k})/2$; their new position is half way their old position and P. If any of these control points lies

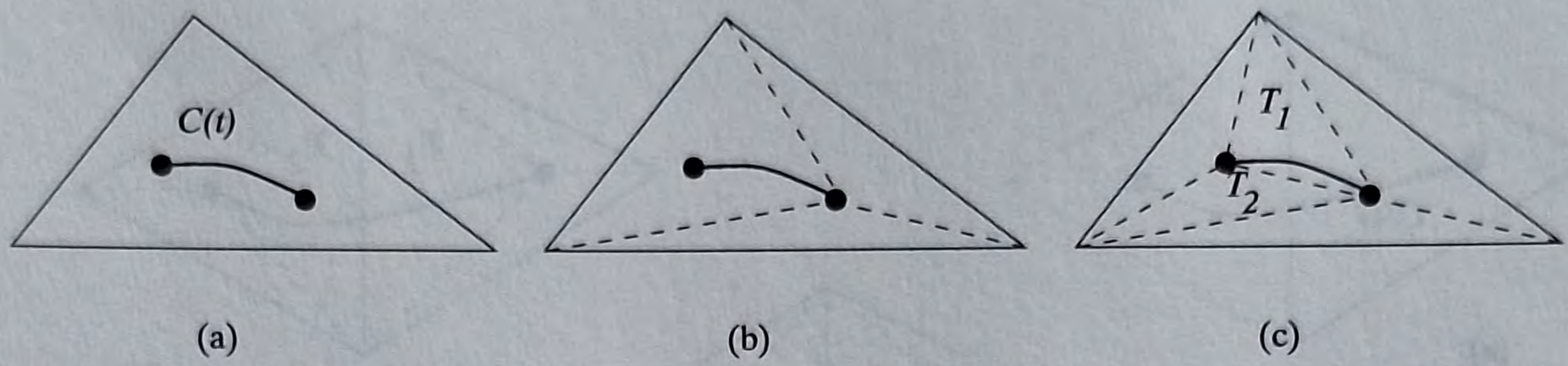


Figure 4: Two endpoints in on patch.

on the boundary of the patch's control polyhedron, the corresponding control point of the neighboring polyhedron is also translated. Hence, if p_{i_0,j_0,k_0} is a vertex, then the total number of control points moved is one plus the number of incident patches. If p_{i_0,j_0,k_0} is not a vertex, the total number is seven. The surface S_1 thus obtained may arbitrarily bad approximate point P, but mimics the final surface in many cases. Finally, u_0 and v_0 are assigned the values for which $S_1(u_0, v_0)$ is closest to P.

Using this method, multiple point interpolation constraints for points close together affect each other. Even worse, there may not be sufficient degrees of freedom to satisfy all constraints. Therefore, each patch may satisfy only one point interpolation constraint. If a second one is specified, the patch is split at the parameter values half way the two.

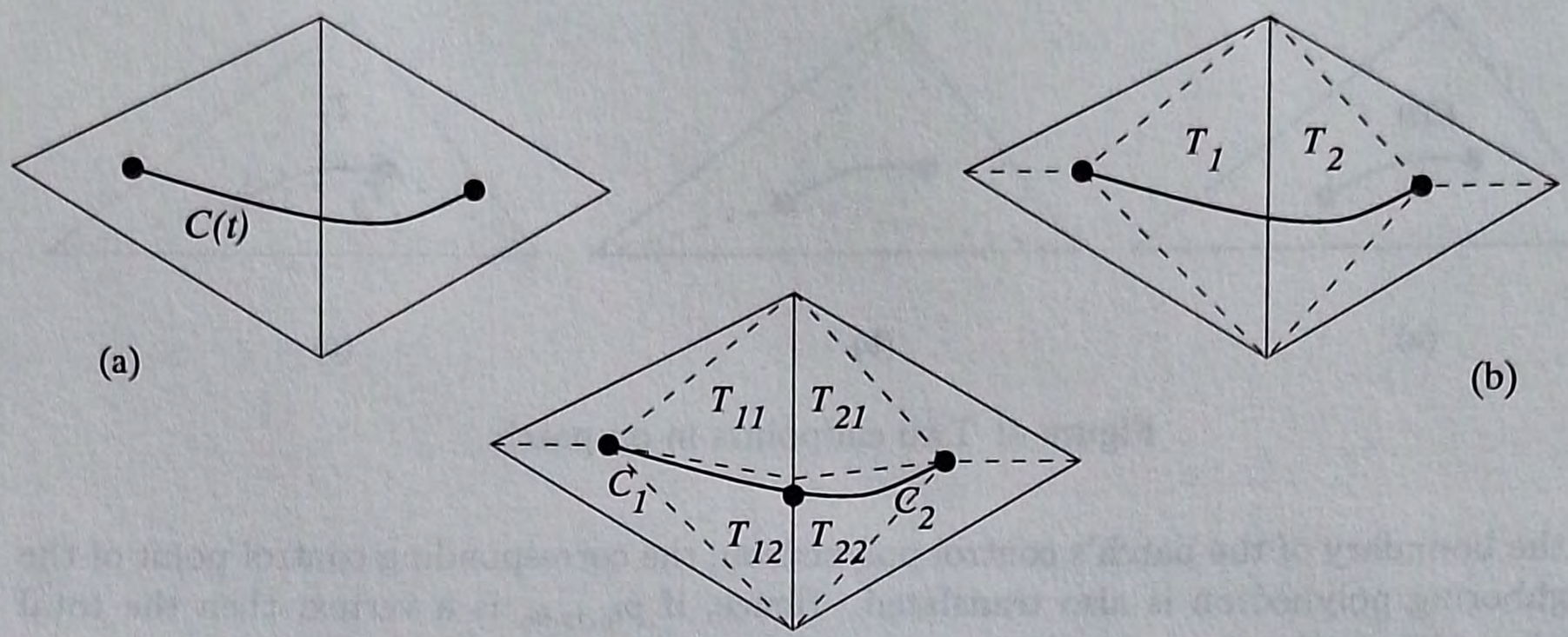
3.2. Curve interpolation

The curve to be interpolated is an arbitrary piecewise polynomial $C(t)$, piecewise defined on intervals $[t_i, t_{i+1}]$. C may for example be a piecewise Bézier curve or a B-spline curve. Since it may be impossible for a single surface patch T to interpolate several segments $C(t)$, $t_i \leq t \leq t_{i+1}$, some appropriate patches T_0 of the initial surface S_0 are subdivided. Constrained optimization with respect to the energy functionals (8) and (9) can be achieved by an iterative process, whose efficiency heavily depends on the starting values, see Section 4. We therefore opt for a two step procedure. First we make the surface satisfy the constraints by subdividing some of the patches and suitably setting control points. Later we optimize the surface with respect to the energy functional while still satisfying the constraints.

Let $C(t)$ be a single polynomial curve of degree k, and assume without loss of generality that this curve is in Bézier form. We distinguish three cases: the two endpoints of $C(t)$ lie in one patch, in two adjacent patches, or in two non-adjacent patches.

Case 1: the two endpoints of $C(t)$ lie in one patch. In this case we split the patch in five triangles by successive subdivision as illustrated in Figure 4. The parameter values at which to split the patch are determined as described in Section 3.1. The control points associated with the common boundary of the triangles T_1 and T_2 are now set equal to those of $C(t)$.

Case 2: the two endpoints $E_1 = C(0)$ and $E_2 = C(1)$ lie in two adjacent patches. We determine values u_i, v_i, w_i at which the patches should interpolate E_i, $i = 1, 2$ as described in Section 3.1. Next we subdivide both patches at these parameter values into three triangles as illustrated in Figure 5. Now let $D(s)$ be the Bézier curve that is the common boundary of T_1 and T_2 (see the figure). Then we determine values t_0 of t and s_0 of s such that the distance between $C(t_0)$ and $D(s_0)$ is minimal, by the numerical method described in [RB90]. $C(t)$ is subdivided at $C(t_0)$ into two curves C_1 and C_2, and the T_i



(c)

Figure 5: Two endpoints in two adjacent patches.

are subdivided at $D(s_0)$ into two triangles T_{i1} and T_{i2}, as shown in Figure 5. The control points of the common boundary of T_{i1} and T_{i2} are set to the control points of C_i, $i = 1, 2$.

Case 3: the endpoints of C lie in two non-adjacent patches. First we determine the two triangles corresponding to these endpoints as in Section 3.1, then we find a sequence of patches T_j, $j = 2, \ldots, k-1$ between those two by a walking procedure, cf. [GS85]. Next we subdivide $C(t)$ into curves C_j similar to case 2, as illustrated in Figure 6. Each curve C_j is split at $C_j(\frac{1}{2})$, $j = 2, \ldots, k-1$, and triangles T_1, T_2, ..., T_{k-1}, and T_k are subdivided with respect to $C_1(0)$, $C_2(\frac{1}{2})$, ..., $C_{k-1}(\frac{1}{2})$, and $C_k(1)$ respectively, as in Section 3.1. We now have a sequence of pairs of triangles as in Figure 5b and follow the procedure of case 2 from there on.

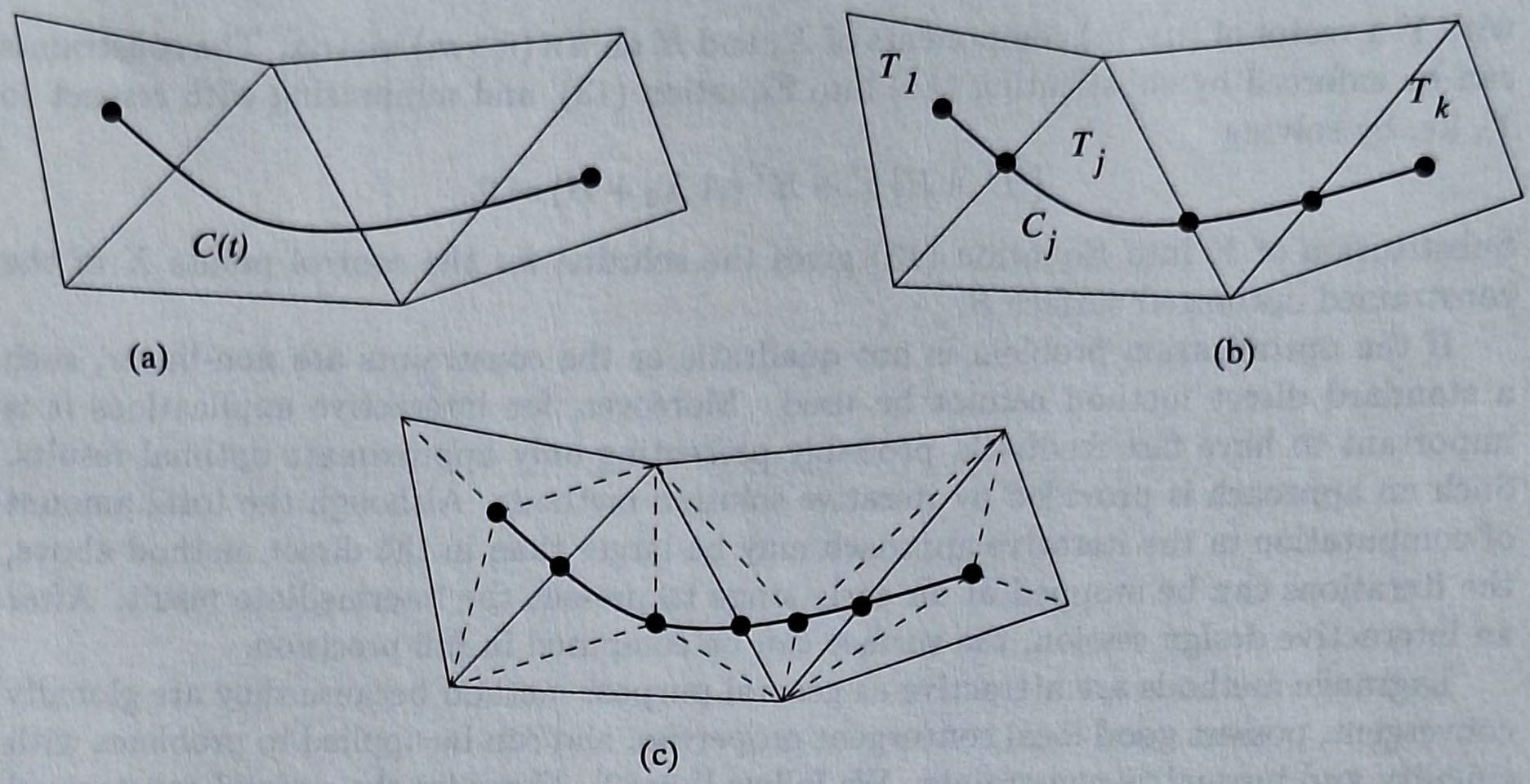
In all three cases the patches must be of sufficiently high degree to set the control points of patch boundaries equal to those of the interpolation curve. Specifically, the degree of the patches may not be lower than the curve's degree.

Note further that the figures show only a schematic situation. The curve to be interpolated does not already lie on patches of S_0; these patches must be deformed to interpolate the curve. Even if the curve behaves wildly it will be fully interpolated, provided that the degree of the patches is sufficiently high.

3.3. Normal interpolation

Suppose that the surface normal N must be interpolated at point $T(u_0, v_0, w_0)$. Let us take two derivative vectors having different directions, for example the directions $d = (0,1,0) - (1,0,0) = (-1,1,0)$ and $c = (0,0,1) - (1,0,0) = (-1,0,1)$. Using Equation (2) we get $\nabla_d T = n(L - K)$ and $\nabla_c T = n(M - K)$, where

$$
\begin{aligned}
K &= \sum_{i+j+k=n-1} p_{i+1,j,k} B^{n-1}_{i,j,k}(u_0, v_0, w_0), \\
L &= \sum_{i+j+k=n-1} p_{i,j+1,k} B^{n-1}_{i,j,k}(u_0, v_0, w_0), \\
M &= \sum_{i+j+k=n-1} p_{i,j,k+1} B^{n-1}_{i,j,k}(u_0, v_0, w_0).
\end{aligned}
$$



(a) (b) (c)

Figure 6: Two endpoints in two non-adjacent patches.

The constraint that the partial derivatives be perpendicular to N is equivalent to $N \cdot (L - K) = 0$ and $N \cdot (M - K) = 0$. For prescribed N, these equations are linear in the control points $p_{i,j,k}$.

4. Computational methods

Sufficient detail to implement the constrained optimization methods for variational surfaces reported in the literature is rarely provided. For a quadratic programming problem with linear equality constraints, the equations forming the standard solution are linear, and thus can be solved by standard procedures.

If the energy function is quadratic in the control points, then unconstrained optimization can be written in matrix form as

$$\min\{X^T A\, X + 2\, B^T\, X + D\}, \tag{13}$$

where X is the vector of all n Bézier control points of the surface, and A is an $n \times n$ matrix. Provided that A is symmetric (which it is in our case), the solution to (13) is given by the linear equation

$$A\, X + B = 0,$$

which can be solved by standard techniques [PFTV88].

If all constraints are linear in the control points, they can be gathered in matrix form as

$$F\, X = G, \tag{14}$$

with F an $m \times n$ constraint matrix and G a vector of size m. For the constraints to be non-conflicting, at least the number of linearly independent constraints must be smaller than or equal to n. By Gaussian elimination and backward substitution, cf. [PFTV88], Equation (14) can be written as $F'\, X = G'$ with elements f'_{ij} of F' such that $f'_{ij} = 0$ for $i > j$, which can be transformed into

$$X = H\, Y + X_0, \tag{15}$$

with Y a vector of $(n-m)$ components of X, and H an $n \times (n-m)$ matrix. The constraints can be enforced by substituting (15) into Equation (13), and minimizing with respect to Y, i.e. by solving

$$[H^T A\, H]\, Y + H^T (A\, X_0 + B) = 0.$$

Substitution of Y into Equation (15) gives the solution for the control points X of the constrained optimized surface S.

If the optimization problem is not quadratic or the constraints are non-linear, such a standard direct method cannot be used. Moreover, for interactive applications it is important to have fast feedback, probably presenting only approximate optimal results. Such an approach is provided by iterative solution methods. Although the total amount of computation in the iterative approach may be larger than in the direct method above, the iterations can be stopped at an early stage to present the intermediate result. After an interactive design session, the surface can be computed in full precision.

Lagrange methods are attractive as general purpose method because they are globally convergent, possess good local convergent properties, and can be applied to problems with equality and inequality constraints. We follow [Lue89]. Consider the general constrained optimization problem

$$\begin{aligned} &\min f(X) \\ \text{subject to} \\ &h(X) = 0, \end{aligned}$$

where X is again the vector of all Bézier control points of the surface. It is convenient to introduce the so-called Lagrangian, defined as $l(X, \lambda) = f(X) + \lambda^T h(X)$, where λ is a vector of auxiliary variables, the Lagrange multipliers. The necessary conditions for a solution of this problem are given by

$$\begin{aligned} \nabla l(X, \lambda) &= 0 \\ h(X) &= 0, \end{aligned}$$

Newton's method solves this system by solving the linearized version iteratively. Given X_k and λ_k, the new solution $X_{k+1} = X_k + d_k$, $\lambda_{k+1} = \lambda_k + y_k$ is determined from the Newton equations

$$\begin{aligned} \nabla l(X_k, \lambda_k)^T + L(X_k, \lambda_k) d_k + \nabla h(X_k)^T y_k &= 0 \\ h(X_k) + \nabla h(X_k) d_k &= 0, \end{aligned}$$

where $L(X_k, \lambda_k)$ is the Hessian of the Lagrangian $l(X_K, \lambda_k)$.

The Newton equations have a unique solution. If the linearized system is nonsingular at the solution, and if the initial values of X_k and λ_k are sufficiently close to the solution, the method will converge with order two. It is therefore important to have a good starting configuration, which is provided by the techniques presented in Section 3.

5. Concluding remarks

This paper has introduced constrained optimization techniques for triangular surfaces. The surfaces are specified through interpolation constraints consisting of points, curves,

and normals. Non-linear optimization techniques are used to minimize an energy functional in order to make the surface smooth. We have given explicit constraints in terms of the surface's control points.

The use of triangular patches rather than the ubiquitous tensor product surfaces provides additional surface modeling power. One major advantage of our techniques of constrained interpolation by means of subdividing triangular patches is that the effect is only local. Parts of the surface remote from the part that interpolates prescribed data is are not unnecessarily subdivided. By contrast, the subdivision of tensor product surfaces has a global effect.

Another advantage of the methods introduced in this paper is that they provide good starting configurations for an iterative optimization process. Such an iterative computational scheme is suitable for interactive use, because intermediate results provide fast feedback. A good starting position is essential for many iterative procedures in order to arrive close to the optimal solution at an early stage in the iteration.

Standard tools and our own methods are currently put together into a prototype system for variational surface design.

Sufficient algorithmic and implementational information is provided so that an experienced practitioner can reproduce our methods.

References

[BF83] Wolfgang Böhm and Gerald Farin. Letter to the editor. *Computer Aided Design*, 15(5), 260 – 261, September 1983.

[CG91] George Celniker and Dave Gossard. Deformable curve and surface finite-elements for free-form shape design. *Computer Graphics*, 25(4), July 1991.

[dC76] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, 1976.

[Far86] Gerald Farin. Triangular Bernstein-Bézier patches. *Computer Aided Geometric Design*, 3(2), 83 – 127, 1986.

[Gol83] Ronald N. Goldman. Subdivision algorithms for Bézier triangles. *Computer Aided Design*, 15(3), 159 – 166. See also [BF83], May 1983.

[GS85] Leonidas Guibas and Jorge Stolfi. Primitives for the manipulation of general subdivisions and the computation of voronoi diagrams. *ACM Transactions on Graphics*, 4(2), 74 – 123, April 1985.

[HS91] Hans Hagen and G. Schulze. Variational principles in curve and surface design. In Hans Hagen and Roller, editors, *Geometric Modelling – Methods and Applications*. Springer-Verlag, 1991.

[Kal93] Michael Kallay. Constrained optimization in surface design. In *Preliminary proceedings IFIP TC5/WG5.10 Working Conference on Modeling in Computer Graphics, Methods and Applications*, 1993.

[KGM91] Dilip R. Khandekar, S. Gopalsamy, and S. P. Mudur. Mathematical techniques for the interactive design of B-spline curves and surfaces using form constraints. In B. Falcidieno, I. Herman, and C. Pienovi, editors, *Preliminary proceedings of the Eurographics Workshop on Computer Graphics and Mathematics, Genova, Italy*, 132 – 149, October 1991.

[LP88] N.J. Lott and D.I. Pullin. Method for fairing b-spline surfaces. *CAD*, 20(10), 597–604, December 1988.

[Lue89] David G. Luenberger. *Linear and Non-Linear Programming.* Addison-Wesley, 2nd edition, 1989.

[MS92] Henry Moreton and Carlo H. Séquin. Functional optimization for fair surface design. *Computer Graphics*, 26(2), July 1992.

[PFTV88] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C, The Art of Scientific Computing.* Cambridge University Press, 1988.

[RB90] John R. Rankin and Jonathan Burns. Coordinate frames and geometric approximation in graphics object oriented programming. In Edwin Blake and Peter Wißkirchen, editors, *Advances in Object-Oriented Graphics I (Proceedings of the Eurographics Workshop on Object-Oriented Graphics, 1990)*, Eurographic-Seminars Series, 131 – 148. Springer-Verlag, 1990.

[RR91] T. Rando and J.A. Roulier. Designing faired parametric surfaces. *CAD*, 23(7), 492–497, September 1991.

[TPBF87] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. *ACM Computer Graphics*, 21(4), 205–214, July 1987.

[WW92] William Welch and Andrew Witkin. Variational surface modeling. *ACM Computer Graphics*, 26(2), 157–166, July 1992.