Available online at www.sciencedirect.com
SCIENCE DIRECT
Computers & Graphics 30 (2006) 581–587
COMPUTERS & GRAPHICS
www.elsevier.com/locate/cag

Technical Section

A smooth, obstacle-avoiding curve

Z. Li[1], D.S. Meek*, D.J. Walton

Department of Computer Science, University of Manitoba, Winnipeg, Canada R3T 2N2

Abstract

An algorithm for finding a smooth, obstacle-avoiding curve in the plane can be quite complicated. The process usually involves finding one or more feasible polyline paths, choosing a desirable path (for example the shortest path), and smoothing the polyline path to give a curve that avoids the obstacles. This paper is concerned with the last stage in the process; it assumes the existence of an obstacle-avoiding polyline path. A method is given to replace that polyline path by a G^2 cubic spline curve that also avoids the obstacles. The advantages of this method are the simplicity of the smooth, obstacle-avoiding curve, and the simplicity of the algorithm that finds the obstacle-avoiding curve.

Keywords: Obstacle-avoiding curve; G^2 continuity

1. Introduction

An algorithm for finding a smooth, obstacle-avoiding curve in the plane can be quite complicated. The process usually involves finding one or more feasible polyline paths. These paths could be derived from a visibility graph. A desirable path, for example the shortest path, is chosen. Finally, the chosen polyline path is smoothed to give a curve that avoids the obstacles (see [1] for a good discussion of the whole process).

The purpose of this paper is to derive a smooth, obstacle-avoiding curve from a given obstacle-avoiding polyline path. Suppose a planar region that contains some obstacles is given. Assume that a polyline path that does not touch any of the obstacles is also given. That polyline path, here called the *guiding polyline*, will be replaced by a smooth curve that avoids the obstacles. The smooth curve has the same number of inflection points as would be suggested by the guiding polyline and the shape of the smooth curve is controlled by the guiding polyline.

A curve is G^1 continuous if it is continuous and has a continuous unit tangent vector (equivalently, the tangent direction is continuous). A curve is G^2 continuous if it is G^1 continuous and its curvature is continuous [2, p. 181]. These definitions give a mathematical measure of smoothness for curves. In this paper, the guiding polyline is replaced by a G^2 cubic spline curve that avoids the obstacles. It is shown that the spline curve can designed to be as close as desired to the guiding polyline and thus a spline curve that avoids the obstacles can always be found. The spline curve used here is a special case of Farin's G^2 cubic spline curve [2, p. 186]. The algorithm for finding the convex hull for a set of points [3, p. 104, 4] is used to increase the efficiency in the treatment of obstacles. The algorithm that tests whether or not a point is inside a convex polygon [3, p. 43] is used for each relevant obstacle point. Finally, explicit formulas or formulas involving the roots of quadratic equations are given for the spline curve that avoids the obstacles.

The problem of path planning has been treated extensively in various fields such as VLSI layout, computational geometry, and robotics [1]. The references quoted below are to work that is similar to the proposed algorithm.

A G^2 boundary-avoiding curve made of rational cubic curves was found in [5] wherein quadratic and quartic equations must be solved. The proposed method solves a similar problem with simpler curves (polynomials instead of rationals) that are easier to find (quadratic instead of

*Corresponding author. Tel.: +1 204 474 8681; fax: +1 204 474 7609.
E-mail address: Dereck_Meek@UManitoba.Ca (D.S. Meek).
[1]Now at: Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200030, China.

0097-8493/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2006.03.003

quartic equations). The curve here is not as "tight" as the curve in [5] in that the one here will seldom touch the obstacles, while the one in [5] will generally touch the boundary in many places.

Some authors have used clothoids to smooth polyline paths. A robot path made of straight lines and clothoid pairs of given maximum curvature is found in [6]. A nonlinear equation is solved to verify the existence of each clothoid pair. The corners of a polyline can be rounded to give G^2 continuity with a symmetric pair of clothoids as in the last case of [7] or as in [8]. The curves are given explicitly, but there is a possibility of adjacent clothoid pairs overlapping. Two advantages of clothoids are that their curvature has an upper bound chosen by the path designer and that the clothoid is a spiral with curvature varying linearly with arc length. A disadvantage of clothoids is that the clothoid is a transcendental curve and thus cannot be represented as a NURBS curve [9,10, p. 259]. That means the clothoid cannot easily be incorporated into standard graphics packages. In comparison, the cubic spline produced here is a NURBS curve and it can be expressed as a collection of cubic Bézier curves. The curvature of a cubic Bézier curve is fairly easy to calculate at any point, although it is neither easy to give an a priori bound on the curvature, nor are the cubic Bézier segments guaranteed to be spirals.

The work of [11,12] finds a smooth path in a polygonal channel. In the planar problem, the desired path is bounded on both sides by polylines. The solution of this problem requires linear programming, so it requires much more computation than the proposed method.

The part of the method in [1] that smoothes the path uses a divide-and-conquer iteration and the resulting curve is G^1 continuous. In contrast, the proposed algorithm finds the curve in linear time (once the convex hulls of the obstacles have been found) and the resulting curve is G^2 continuous.

2. Theoretical background

2.1. Notation and definitions

The length of a vector $\mathbf{v}$ is denoted as

$$|\mathbf{v}| = \sqrt{v_x^2 + v_y^2}.$$

The scalar cross product of two vectors $\mathbf{v}$ and $\mathbf{w}$ is

$$\mathbf{v} \times \mathbf{w} = v_x w_y - v_y w_x = |\mathbf{v}||\mathbf{w}| \sin\theta,$$

where θ is the counterclockwise angle from $\mathbf{v}$ to $\mathbf{w}$. Recall that the cross product of two parallel vectors is zero.

The cubic Bézier curve with control points $\{\mathbf{B}_0, \mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3\}$ is [2, p. 44]

$$\mathbf{B}(t) = (1-t)^3\mathbf{B}_0 + 3(1-t)^2 t\mathbf{B}_1 + 3(1-t)t^2\mathbf{B}_2 + t^3\mathbf{B}_3, \quad 0 \leqslant t \leqslant 1. \tag{1}$$

A polyline joining vertices $\mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_n$ is said to be *C-shaped* if the following cross products are all positive or all negative:

$$(\mathbf{B}_i - \mathbf{B}_{i-1}) \times (\mathbf{B}_{i+1} - \mathbf{B}_i), \quad i = 1, 2, \ldots, n-1. \tag{2}$$

A C-shaped polyline is said to be in *standard form* if the above cross products are all positive. In the theoretical discussion below, the guiding polyline will be assumed to be in standard form. Any given C-shaped polyline can be reflected across any axis, if necessary, to put it into standard form.

A C-shaped polyline in standard form joining the vertices $\mathbf{B}_0$, $\mathbf{B}_1$, $\mathbf{B}_2$, $\mathbf{B}_3$ is said to be *self-intersecting* if and only if the following holds (refer to (2)):

$$(\mathbf{B}_0 - \mathbf{B}_1) \times (\mathbf{B}_3 - \mathbf{B}_1) \geqslant 0 \quad \text{and} \quad (\mathbf{B}_0 - \mathbf{B}_2) \times (\mathbf{B}_3 - \mathbf{B}_2) \geqslant 0. \tag{3}$$

2.2. Farin's G^2 cubic spline

A one-parameter family of curves with the parameter u in (0, 1) is obtained from Farin's G^2 cubic spline curve [2, p. 186] by choosing $\alpha_i = u/2$ and $\omega_i = 1 - (u/2)$ for each i. The complete definition of this special spline is given below. Let $\mathbf{d}_i$, $i = 0, 1, \ldots, n$, be $n+1$ distinct guiding-polyline vertices; they are also control points of Farin's spline. Let $\mathbf{B}_i(u)$, $i = 0, 1, \ldots, 3n$, be the $3n+1$ control points of the associated cubic Bézier curves; these Bézier control points depend on the parameter u. Choose the end Bézier control points as $\mathbf{B}_0(u) = \mathbf{d}_0$, $\mathbf{B}_{3n}(u) = \mathbf{d}_n$; choose the interior Bézier control points $\mathbf{B}_{3i+1}(u)$ and $\mathbf{B}_{3i+2}(u)$ on the line segment $\mathbf{d}_i\,\mathbf{d}_{i+1}$ as

$$\mathbf{B}_{3i+1}(u) = \left(1 - \frac{u}{2}\right)\mathbf{d}_i + \frac{u}{2}\mathbf{d}_{i+1},$$
$$\mathbf{B}_{3i+2}(u) = \frac{u}{2}\mathbf{d}_i + \left(1 - \frac{u}{2}\right)\mathbf{d}_{i+1}, \quad i = 0, 1, \ldots, n-1; \tag{4}$$

and choose the interior control points $\mathbf{B}_{3i}(u)$ as

$$\mathbf{B}_{3i}(u) = \frac{1}{2}\mathbf{B}_{3i-1}(u) + \frac{1}{2}\mathbf{B}_{3i+1}(u) = \mathbf{d}_i + \frac{u}{4}(\mathbf{d}_{i-1} - 2\mathbf{d}_i + \mathbf{d}_{i+1}), \quad i = 1, 2, \cdots, n-1. \tag{5}$$

This choice for $\mathbf{B}_{3i}(u)$ gives G^2 continuity at the joint of two adjacent cubic Bézier curves. Fig. 1 illustrates these definitions.

It can be seen that Farin's G^2 cubic spline interpolates the end vertices $\mathbf{d}_0$, $\mathbf{d}_n$, and has beginning and ending tangent vectors that match the first and last legs of the guiding polyline, $\mathbf{d}_0\,\mathbf{d}_1$, and $\mathbf{d}_{n-1}\,\mathbf{d}_n$.

In the present paper, the vertices $\mathbf{d}_i$ are confined to the plane. A notation that provides concise expressions is

$$\mathbf{e}_i = \mathbf{d}_i - \mathbf{d}_{i-1}, \quad i = 1, 2, \ldots, n. \tag{6}$$

In the theoretical discussion below, the vertex $\mathbf{d}_i$ will be retained and the other vertices will be expressed through $\mathbf{d}_i$ and (6).

The vertices $\mathbf{d}_i$ are restricted so that they form a C-shaped polyline. For the theoretical results, this is assumed to be a C-shaped polyline in standard form so that

$$\mathbf{e}_i \times \mathbf{e}_{i+1} > 0, \quad i = 1, 2, \ldots, n-1. \tag{7}$$

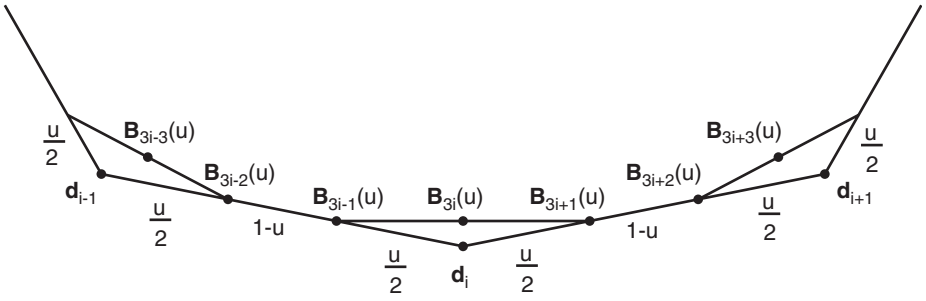

Fig. 1. A special case of Farin's G^2 cubic spline.

The algorithm in Section 3 is constructed such that C-shaped polylines are the only ones that are necessary.

The vertices $\mathbf{d}_i$ are restricted so that the polyline joining any four consecutive vertices is not self-intersecting. This restriction simplifies analysis in Section 2.5 and is not too severe a restriction as the guiding polyline is still allowed to intersect itself over more than four vertices. From (3), the necessary and sufficient condition that the C-shaped polyline in standard form joining the four vertices $\mathbf{d}_{i-1}$, $\mathbf{d}_i$, $\mathbf{d}_{i+1}$, $\mathbf{d}_{i+2}$ does not self-intersect is

$$(\mathbf{d}_{i-1}-\mathbf{d}_i)\times(\mathbf{d}_{i+2}-\mathbf{d}_i)<0 \quad \text{or}$$
$$(\mathbf{d}_{i-1}-\mathbf{d}_{i+1})\times(\mathbf{d}_{i+2}-\mathbf{d}_{i+1})<0.$$

Using (6), this can be written

$$\mathbf{e}_i\times\mathbf{e}_{i+1}+\mathbf{e}_i\times\mathbf{e}_{i+2}>0 \quad \text{or} \quad \mathbf{e}_i\times\mathbf{e}_{i+2}+\mathbf{e}_{i+1}\times\mathbf{e}_{i+2}>0. \tag{8}$$

2.3. The cubic Bézier curve

Farin's cubic spline can be expressed as a collection of cubic Bézier curves. In this subsection, the control points, the control polyline, the arc length, and the curvature of these Bézier curves are examined. It should be noted that the arc length and maximum curvature can be computed for a given u. The more difficult problem of finding a u that gives a desired arc length or desired maximum curvature is not treated here.

The end Bézier curves each have three of their control points collinear while the interior Bézier curves each have four control points of the form (using (4)–(6))

$$\begin{aligned}
\mathbf{B}_{3i}(u) &= \mathbf{d}_i+\frac{u}{4}(\mathbf{e}_{i+1}-\mathbf{e}_i),\\
\mathbf{B}_{3i+1}(u) &= \mathbf{d}_i+\frac{u}{2}\mathbf{e}_{i+1},\\
\mathbf{B}_{3i+2}(u) &= \mathbf{d}_i+\mathbf{e}_{i+1}-\frac{u}{2}\,e_{i+1},\\
\mathbf{B}_{3i+3}(u) &= \mathbf{d}_i+\mathbf{e}_{i+1}+\frac{u}{4}(\mathbf{e}_{i+2}-\mathbf{e}_{i+1}), \quad i=1,2,\ldots,n-2,
\end{aligned} \tag{9}$$

which can be used in (1). A related formula

$$\mathbf{B}_{3i-1}(u)=\mathbf{d}_i-\frac{u}{2}\mathbf{e}_i \tag{10}$$

is used later.

The restrictions on the guiding polyline vertices $\mathbf{d}_i$ result in restrictions on the Bézier control points in (9). Using (9) and (7),

$$\begin{aligned}
&(\mathbf{B}_{3i+1}(u)-\mathbf{B}_{3i}(u))\times(\mathbf{B}_{3i+2}(u)-\mathbf{B}_{3i+1}(u))\\
&\quad=\frac{(1-u)u}{4}\mathbf{e}_i\times\mathbf{e}_{i+1}>0,
\end{aligned}$$

$$\begin{aligned}
&(\mathbf{B}_{3i+2}(u)-\mathbf{B}_{3i+1}(u))\times(\mathbf{B}_{3i+3}(u)-\mathbf{B}_{3i+2}(u))\\
&\quad=\frac{(1-u)u}{4}\mathbf{e}_{i+1}\times\mathbf{e}_{i+2}>0,
\end{aligned}$$

which shows that the Bézier control polyline is C-shaped and is in standard form. Using (9), (7), and (8),

$$\begin{aligned}
&(\mathbf{B}_{3i+1}(u)-\mathbf{B}_{3i}(u))\times(\mathbf{B}_{3i+3}(u)-\mathbf{B}_{3i+2}(u))\\
&\quad=\frac{u^2}{16}[\mathbf{e}_i\times\mathbf{e}_{i+1}+\mathbf{e}_{i+1}\times\mathbf{e}_{i+2}+\mathbf{e}_i\times\mathbf{e}_{i+2}]>0.
\end{aligned}$$

This means that the Bézier control polyline is an "open" C-shape; it does not self-intersect and its convex hull is a quadrilateral.

The distance between $\mathbf{B}_{3i}(u)$ and $\mathbf{d}_i$ is

$$|\mathbf{B}_{3i}(u)-\mathbf{d}_i|=\frac{u}{4}|\mathbf{d}_{i-1}-2\mathbf{d}_i+\mathbf{d}_{i+1}|.$$

As u approaches zero, the control points $\mathbf{B}_{3i-1}(u)$ and $\mathbf{B}_{3i+1}(u)$ move together and $\mathbf{B}_{3i}(u)$ moves in a straight line to $\mathbf{d}_i$. As u approaches one, the control points $\mathbf{B}_{3i-2}(u)$ and $\mathbf{B}_{3i-1}(u)$ move together, the control points $\mathbf{B}_{3i+1}(u)$ and $\mathbf{B}_{3i+2}(u)$ move together, and $\mathbf{B}_{3i}(u)$ moves in a straight line to the point $\frac{1}{4}(\mathbf{d}_{i-1}-2\mathbf{d}_i+\mathbf{d}_{i+1})$ (see Fig. 1). The arc length of a cubic Bézier curve (1) cannot be found explicitly, but it can be found by numerical integration [10, p. 133] in the expression

$$\int_0^1|\mathbf{B}'(t)|\,\mathrm{d}t.$$

The arc length of Farin's spline is the sum of the arc lengths of the component Bézier curves.

The curvature $k(t)$ of a planar cubic Bézier curve (1) when $|\mathbf{B}'(t)|\neq 0$ is

$$k(t)=\frac{\mathbf{B}'(t)\times\mathbf{B}''(t)}{|\mathbf{B}'(t)|^3}.$$

To find the maximum curvature of a cubic Bézier curve, the curvature extrema, which are the zeros of a fifth degree

equation in [0, 1], must be found [13]. Since the control polyline is C-shaped, there can be only three such extrema in [0, 1]. The curvature values at those extrema, at 0, and at 1 allow one to find the maximum curvature. A much simpler calculation in [14] gives a bound on the curvature. That bound is obtained by expressing the numerator and denominator in Bernstein polynomial form and bounding those polynomials. The maximum curvature of Farin's spline is the largest of the maximum curvatures of the component Bézier curves.

The curvature of Farin's spline at $\mathbf{d}_0 = \mathbf{B}_0$ and $\mathbf{d}_n = \mathbf{B}_{3n}$ is zero since both of the triples $\{\mathbf{B}_0, \mathbf{B}_1, \mathbf{B}_2\}$ and $\{\mathbf{B}_{3n-2}, \mathbf{B}_{3n-1}, \mathbf{B}_{3n}\}$ are collinear.

The two cubic Bézier curves that have control points $\{\mathbf{B}_{3i-3}(u), \mathbf{B}_{3i-2}(u), \mathbf{B}_{3i-1}(u), \mathbf{B}_{3i}(u)\}$ and $\{\mathbf{B}_{3i}(u), \mathbf{B}_{3i+1}(u), \mathbf{B}_{3i+2}(u), \mathbf{B}_{3i+3}(u)\}$ are adjacent. The curvature at their joint $\mathbf{B}_{3i}(u)$ is

$$\frac{32(1-u)}{3u^2}\frac{\mathbf{e}_i \times \mathbf{e}_{i+1}}{|\mathbf{e}_i + \mathbf{e}_{i+1}|^3}, \quad i = 1, 2, \ldots, n-1. \tag{11}$$

As u approaches zero, the curvature at $\mathbf{B}_{3i}(u)$ increases without bound. As u approaches one, the curvature at $\mathbf{B}_{3i}(u)$ approaches zero. The derivative of (11) with respect to u is negative for $0 < u < 1$, so the curvature at the joint of two segments decreases monotonely as u increases, $0 < u < 1$.

2.4. The correct side of a point obstacle near a guiding-polyline vertex

Assume that an obstacle is represented by one point $\mathbf{P}$ near $\mathbf{d}_i$ (see Fig. 2). Result 1 shows that the joining legs of two adjacent cubic Bézier curves can always be positioned on the correct side of a point obstacle.

Result 1. If $\mathbf{P}$ is in the triangle with vertices $\{\mathbf{B}_{3i-1}(u_a), \mathbf{d}_i, \mathbf{B}_{3i+1}(u_a)\}$, $0 < u_a < 1$, a value $u = u_P$, $0 < u_P < u_a$ can always be found so that $\mathbf{P}$ is excluded from the interior of the triangle $\{\mathbf{B}_{3i-1}(u_P), \mathbf{d}_i, \mathbf{B}_{3i+1}(u_P)\}$ (see Fig. 2).

Proof. If $\mathbf{P}$ is on the line segment $\mathbf{B}_{3i-1}(u)\ \mathbf{B}_{3i+1}(u)$, then

$$(\mathbf{B}_{3i+1}(u) - \mathbf{B}_{3i-1}(u)) \times (\mathbf{P} - \mathbf{B}_{3i-1}(u)) = 0,$$

or, using (9) and (10),

$$\frac{u}{2}(\mathbf{e}_i + \mathbf{e}_{i+1}) \times \left[\mathbf{P} - \mathbf{d}_i + \frac{u}{2}\mathbf{e}_i\right] = 0.$$

The root $u = 0$ is not of interest, so the required value of u is

$$u_P = 2\frac{(\mathbf{e}_i + \mathbf{e}_{i+1}) \times (\mathbf{P} - \mathbf{d}_i)}{\mathbf{e}_i \times \mathbf{e}_{i+1}}. \tag{12}$$

If $\mathbf{P}$ is inside the triangle $\{\mathbf{B}_{3i-1}(u_a), \mathbf{d}_i, \mathbf{B}_{3i+1}(u_a)\}$, then $\mathbf{P}$ can be expressed as

$$\mathbf{P} = \mathbf{d}_i + \alpha(\mathbf{B}_{3i-1}(u_a) - \mathbf{d}_i) + \beta(\mathbf{B}_{3i+1}(u_a) - \mathbf{d}_i),$$

where α and β are positive with the restriction that $\alpha + \beta < 1$. Using (9) and (10),

$$\mathbf{P} = \mathbf{d}_i - \frac{\alpha u_a}{2}\mathbf{e}_i + \frac{\beta u_a}{2}\mathbf{e}_{i+1},$$

and substituting the above expression into (12) gives

$$u_P = (\alpha + \beta)u_a,$$

from which it is clear that $0 < u_P < u_a$. □

2.5. A cubic Bézier curve that avoids a point obstacle

As u is reduced towards zero, the Bézier control points (9) behave in a very predictable manner (see Fig. 3). Result 2 shows that if there is a point obstacle $\mathbf{P}$ in the convex hull of those Bézier control points, then a smaller u can always be found so that the convex hull excludes $\mathbf{P}$.

Result 2. If the convex hull of $\{\mathbf{B}_{3i}(u_a), \mathbf{B}_{3i+1}(u_a), \mathbf{B}_{3i+2}(u_a), \mathbf{B}_{3i+3}(u_a)\}$, $0 < u_a < 1$, includes a point $\mathbf{P}$, a value $u = u_P$, $0 < u_P < u_a$, can always be found such that $\mathbf{P}$ is excluded from the interior of the convex hull of $\{\mathbf{B}_{3i}(u_P), \mathbf{B}_{3i+1}(u_P), \mathbf{B}_{3i+2}(u_P), \mathbf{B}_{3i+3}(u_P)\}$ (see Fig. 4).

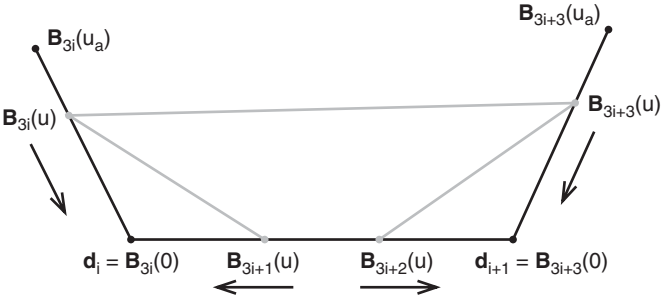

Fig. 3. Control points of a cubic Bézier curve. The arrows indicate the direction of motion as u approaches zero.

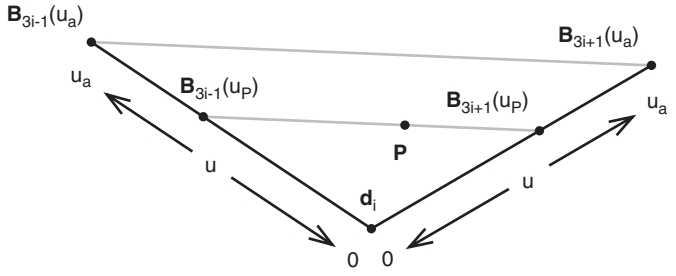

Fig. 2. An obstacle point $\mathbf{P}$ in the triangle with vertices $\{\mathbf{B}_{3i-1}(u_a), \mathbf{d}_i, \mathbf{B}_{3i+1}(u_a)\}$.

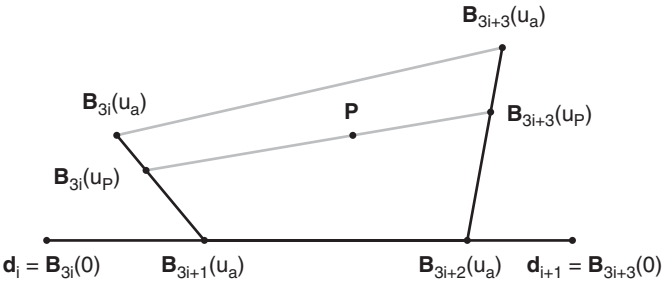

Fig. 4. An obstacle point $\mathbf{P}$ in the convex hull of $\{\mathbf{B}_{3i}(u_a), \mathbf{B}_{3i+1}(u_a), \mathbf{B}_{3i+2}(u_a), \mathbf{B}_{3i+3}(u_a)\}$.

Proof. If $\mathbf{P}$ is on the line segment $\mathbf{B}_{3i}(u)\ \mathbf{B}_{3i+3}(u)$, then $f(u)=0$, where

$$f(u)=[\mathbf{B}_{3i+3}(u)-\mathbf{B}_{3i}(u)]\times[\mathbf{P}-\mathbf{B}_{3i}(u)]. \tag{13}$$

Using (9), $f(u)$ can be written as

$$\begin{aligned} f(u)=&\frac{u^2}{16}[\mathbf{e}_i\times\mathbf{e}_{i+1}+\mathbf{e}_{i+1}\times\mathbf{e}_{i+2}-\mathbf{e}_i\times\mathbf{e}_{i+2}]\\ &+\frac{u}{4}[(\mathbf{e}_i-2\mathbf{e}_{i+1}+\mathbf{e}_{i+2})\times(\mathbf{P}-\mathbf{d}_i)-\mathbf{e}_i\times\mathbf{e}_{i+1}]\\ &+[\mathbf{e}_{i+1}\times(\mathbf{P}-\mathbf{d}_i)]. \end{aligned}$$

From (13), $f(0)>0$ and $f(u_a)<0$, so $f(u)$ changes sign as u varies from zero to u_a. That means there is one root of the quadratic equation $f(u)=0$ in the interval $(0, u_a)$ and that root is u_P.

Since $f(u)<0$ for all $0<u<u_P$, (13) shows that $\mathbf{P}$ is excluded from the interior of the convex hull of $\{\mathbf{B}_{3i}(u_P), \mathbf{B}_{3i+1}(u_P), \mathbf{B}_{3i+2}(u_P), \mathbf{B}_{3i+3}(u_P)\}$. □

The approach taken in this paper to make a cubic Bézier curve avoid an obstacle is to force the convex hull of its control points to exclude the obstacle. The resulting curve will generally be farther away from the obstacle than absolutely necessary. If one wishes to find the true distance from a point obstacle $\mathbf{P}$ to a cubic Bézier curve $\mathbf{B}(t)$, one must solve a quintic equation to find the parameter values t in [0, 1] for which $\mathbf{B}'(t)\cdot(\mathbf{B}(t)-\mathbf{P})=0$. Of the acceptable values, including the end values 0 and 1, one must find the smallest value of $|\mathbf{B}(t)-\mathbf{P}|$. Thus, an alternative to using convex hulls could be to use a numerical minimization method [10, p. 398] to get the u that produces a curve as close to the obstacles as possible. For each trial value of u, a numerical equation-solving method [10, p. 373] would be used determine the smallest distance for that u. The result would be a method which gives more precise results than the proposed method but at a much greater computing cost.

3. The algorithm

The algorithm falls into two parts: the first part is to partition the given guiding polyline into C-shaped sections; the second part is to choose a u-value in each C-shaped section such that the resulting spline avoids the relevant obstacles.

3.1. Partition of the guiding polyline into C-shaped sections

A simple method for partitioning the guiding polyline into C-shaped polyline sections is now described (see Fig. 5). If three adjacent legs of the guiding polyline form an S-shape, choose an inflection point at the midpoint of the centre leg. This inflection point becomes a new vertex in the guiding polyline. The first and last vertices of the guiding polyline and these inflection points divide the guiding polyline into C-shaped sections. This partitioning

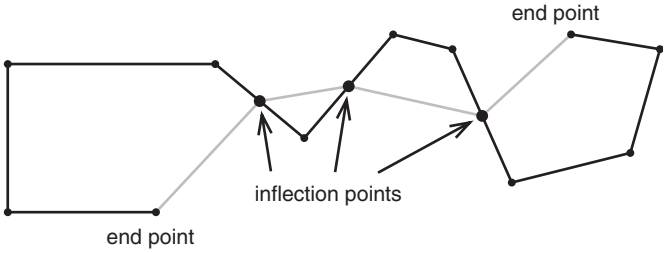

Fig. 5. Partition of a guiding polyline into four C-shaped sections. The original vertices of the guiding polyline are small dots and the inflection points are large dots.

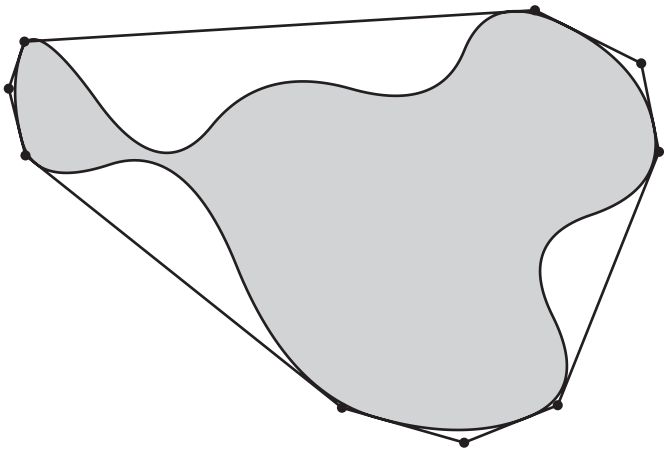

Fig. 6. Bounding a curvilinear obstacle with a convex polygon.

is the justification for restricting the treatment to C-shaped guiding polylines in Section 2.

Partitioning the guiding polyline into C-shaped sections gives a fairly natural arrangement such that there are several adjustable parameters for the resulting curve, one for each C-shaped section. Another advantage of C-shaped sections is that it is particularly easy to verify that a C-shaped curve avoids the obstacles, as will be explained in Section 3.2.

3.2. The treatment of obstacles

A polygonal obstacle can be treated efficiently by representing it by the vertices of its convex hull. Even an obstacle with curved sides can be bounded by a convex polygon (see Fig. 6 and [15]). There is probably no general algorithm to find the "best" bounding convex polygon, but individual cases are often treated easily.

A C-shaped curve that avoids the vertices of the convex hull of an obstacle avoids the obstacle completely (see Fig. 7). This property makes C-shaped curves a desirable type of curve.

Since a Bézier curve falls inside the convex hull of its control points, only the obstacle points that also fall inside that convex hull are relevant to that curve. For example, in Fig. 8, only the vertices $\{\mathbf{O}_1, \mathbf{O}_2, \mathbf{O}_4, \mathbf{O}_5\}$ of the obstacle need be considered. Since the Bézier curve is C-shaped,

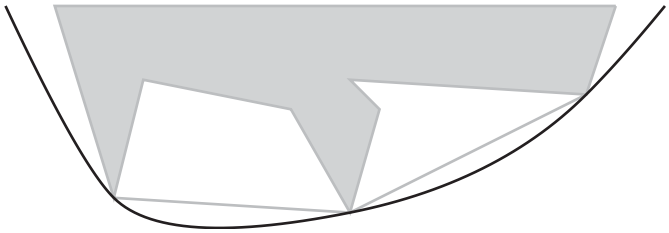

Fig. 7. A C-shaped curve avoiding the convex hull of an obstacle.

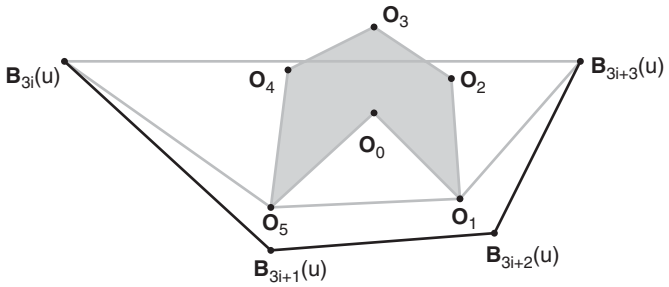

Fig. 8. Bézier control polyline and the convex hull of the obstacle points.

only the obstacle vertices of the convex hull of those obstacle vertices together with $\mathbf{B}_{3i}(u)$ and $\mathbf{B}_{3i+3}(u)$ are relevant. This means only the vertices $\{\mathbf{O}_1, \mathbf{O}_5\}$ of the obstacle are relevant. A similar analysis can be applied if several obstacles fall within the convex hull of the Bézier control points.

3.3. The choice of the parameter u

For each C-shaped section of the guiding polyline, an initial value of u such as $u = 0.8$ can be chosen. Now work through Farin's spline to ensure that the Bézier control polylines are on the correct side of the obstacles. This may require reducing u as described in Section 2.4. Work through Farin's spline again to ensure that there are no obstacles inside the convex hulls of each of the sets of four Bézier control points. This may require reducing u further as described in Section 2.5. The final u is the value after all the reductions, if there are any. Farin's spline with this u parameter is a C-shaped G^2 curve that avoids the obstacles.

4. Examples

Example 1 is shown in Fig. 9. The seven original points of the guiding polyline are $\{(50, 10)^T, (90, 60)^T, (130, 30)^T, (250, 80)^T, (60, 150)^T, (50, 50)^T, (0, 140)^T\}$. The value $u = 0.8$ was used for the initial curves, which are shown in grey. The polyline was partitioned into three C-shaped sections. The u-values that give an obstacle-avoiding curve were then calculated to be $u = 0.531$, $u = 0.372$, and $u = 0.661$ for the three C-shaped sections. The obstacle-avoiding curves are shown in black. It is worth noting that the curve sometimes passes far from the obstacles. This is partly because the calculations were designed to make the convex hull of the control points of each cubic Bézier exclude the obstacles rather than work with the curve itself. If the obstacle-avoiding curve is too far from the obstacles, one could achieve a better curve with very little effort by choosing a guiding polyline with more control points that are closer to a desired path.

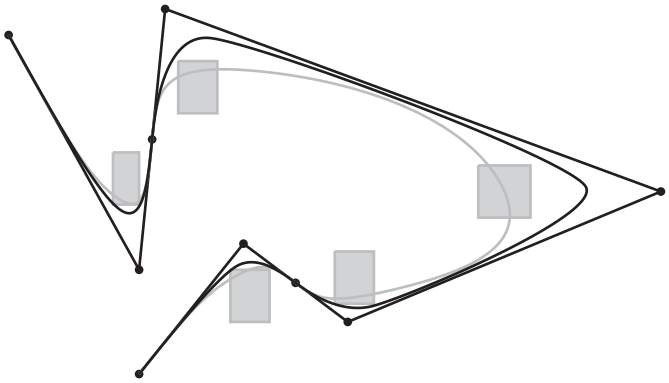

Fig. 9. Example 1 of an obstacle-avoiding curve.

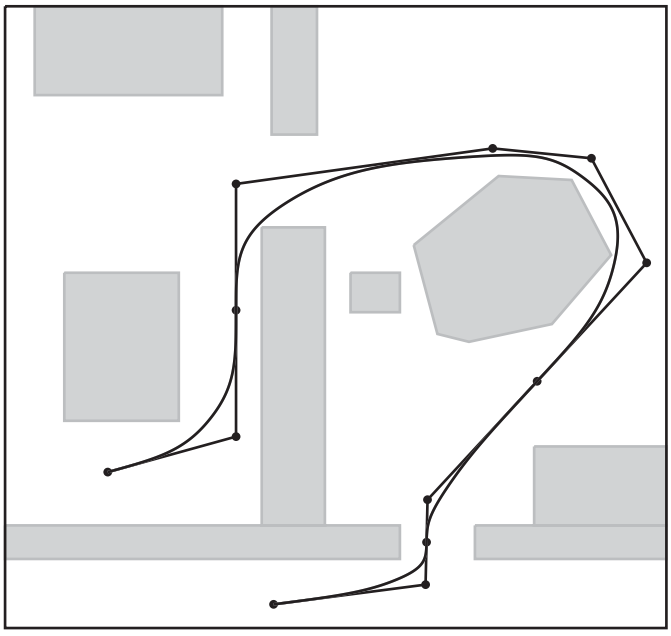

Fig. 10. Example 2 of an obstacle-avoiding curve.

Example 2, shown in Fig. 10, is taken from [16, Fig. 6]. It shows a robot's polyline path that threads around obstacles. The smoothed version of this path obtained with $u = 0.8$ is shown. This value of u gives an obstacle-avoiding path without modification. There are four C-shaped sections in this example.

5. Conclusions

A method for deriving a G^2 obstacle-avoiding cubic spline curve from an obstacle-avoiding polyline path is given. The chief advantages of this method are the simplicity of the curve and simplicity of the algorithm. The curve is found through the solution of (at worst)

quadratic equations. The use of a convex-hull-finding algorithm can improve the efficiency of the curve-finding algorithm described in this paper.

Concerning the complexity of the algorithm, most results are given by explicit formula. The most time-consuming part of the algorithm is the finding of convex hulls of a set of points. If there are n points, the best algorithm is known to be $O(n \log n)$. For the formulas we provide, a polyline with n segments would require $O(n)$ operations and the most difficult type of equation you have to solve is the quadratic.

Acknowledgements

The authors acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada for this research. The authors gratefully appreciate the comments of two anonymous referees. They have improved the presentation of this paper, especially in Section 2.5.

References

[1] Dobkin DP, Gansner ER, Koutsofios E, North SC. Implementing a general-purpose edge router. In: Di Battista G, editor. Graph Drawing 97. Springer-Verlag Lecture Notes in Computer Science, vol. 1353. Rome, Italy: Springer; 1998.

[2] Farin G. Curves and surfaces for computer-aided geometric design: a practical guide, 4th ed. San Diego: Academic Press; 1997.

[3] Preparata FP, Shamos MI. Computational geometry: an introduction, third corrected printing. New York: Springer; 1990.

[4] Barber CB, Dobkin DP, Huhdanpan H. The quickhull algorithm for convex hulls. ACM Transactions on Mathematical Software 1996;22:469–83.

[5] Meek DS, Ong BH, Walton DJ. Constrained interpolation with rational cubics. Computer Aided Geometric Design 2003;20:253–75.

[6] Scheuer A, Fraichard Th. Planning continuous-curvature paths for car-like robots. In: IEEE/RSJ international conference on intelligent robots and systems, vol. 3, Osaka, Japan, 4–8 November 1996. p. 1304–11.

[7] Meek DS, Walton DJ. The use of Cornu spirals in drawing planar curves of controlled curvature. Journal of Computational and Applied Mathematics 1989;25:69–78.

[8] Fleury S, Souàres P, Laumond J-P, Chatila R. Primitives for smoothing mobile robot trajectories. In: IEEE international conference on robotics and automation, vol. 1, Atlanta, Georgia, 2–6 May 1993. p. 832–9.

[9] Piegl L, Tiller W. The NURBS book. 2nd ed. Berlin: Springer; 1997.

[10] Press WH, Teukolsky SA, Vetterling WT, Flannery BP. Numerical recipes in C++. 2nd ed. Cambridge, England: Cambridge University Press; 2002.

[11] Lutterkort D, Peters J. Smooth paths in a polygonal channel. In: Proceedings of the 15th annual symposium on computational geometry, 1999. p. 316–21.

[12] Myles A, Peters J. Threading splines through 3D channels. Computer-Aided Design 2005;37:139–48.

[13] Walton DJ, Meek DS. Curvature extrema of planar parametric cubic curves. Journal of Computational and Applied Mathematics 2001;134:69–83.

[14] Walton DJ, Meek DS. Curvature bounds for planar B-spline curve segments. Computer-Aided Design 1988;20:146–50.

[15] Schäffer AA, Van Wyk CJ. Convex hulls of piecewise-smooth Jordan curves. Journal of Algorithms 1987;8:66–94.

[16] Bourhis G, Horn O, Habert O, Pruski A. An autonomous vehicle for people with motor disabilities. IEEE Robotics & Automation Magazine 2001;8:20–8.