Spline Trajectory Planning for Road-Like Path with Piecewise Linear Boundaries Allowing Double Corner Points

Hiroyuki KANO * and Hiroyuki FUJIOKA **

Abstract: We consider a problem of trajectory planning for road-like path on the two-dimensional plane. As the basic tool for constructing trajectories, we employ smoothing splines using normalized uniform B-splines as the basis functions. The pass is assumed to possess piecewise linear boundaries, specified by a series of pairs of right and left corner points. We solve two-fold corner points in order to allow flexible description of the path. On constructing smoothing splines, we impose the boundary constraints as a collection of inequality pairs by right and left boundary lines, yielding a set of linear inequality constraints on the so-called control point vector. Unlike standard smoothing spline settings, a piecewise linear centerline of the given path is provided as the data for the trajectory to follow, where the given entire time interval is divided into subintervals according to the centripetal distribution rule. Other constraints, typically as the initial and final conditions, can be imposed on the trajectory easily, and we see that the problem reduces to strictly convex QP (quadratic programming) problem. Efficient QP solvers are available for numerical solution, and the effectiveness of the proposed method is confirmed by three examples: two with piecewise linear boundaries including an example of obstacle avoidance problem, and the third with piecewise linear approximation of circular boundaries.

Key Words: trajectory planning, road-like path, B-splines, smoothing splines, convex quadratic programming.

1. Introduction

In general, the problem of trajectory planning consists of constructing a function of time that satisfies initial and final conditions with some other specifications such as via points and obstacle avoidance. Splines are very useful for such a purpose since they are piecewise functions and hence easy to construct and modify. In fact they have been used frequently in the field of robotics as for trajectory planning of robotic arms and mobile robots [1]–[3].

Via points can be treated by interpolating splines which pass the points exactly or by approximating or smoothing splines where the points are passed only approximately [4],[5]. The latter is more desirable since allowing small errors in general yields more flexible design of curves. On the other hand, the problem of obstacle avoidance is often treated in trajectory planning by introducing a cost function or by imposing inequality constraints on the trajectory.

A cost function is constructed for example based on the distance to obstacles together with, e.g., the trajectory length, which is expressed as a nonlinear function of some parameters representing the trajectories. The resulting nonlinear optimization problem has to be solved numerically to yield desired trajectory, frequently by expressing the trajectory by cubic splines [6]–[8]. In [8], a particle swarm method is employed to solve optimization problems.

In [9], obstacles are treated by linear inequality constraints, and a nonlinear programming (NLP) problem is solved for quartic B-spline curves with minimum curvature. But only B-splines of degrees two to four are allowed. In [10], by representing trajectories by a sum of harmonics (sine and cosine functions), a trajectory planning problem is formulated as an NLP problem, where obstacles are treated by inequality constraints by assuming their parametric representation as polygons and ellipses. In [11], minimum-time trajectory planning is considered by employing cubic B-splines, first by finding obstacle avoidance path and then optimizing in traveling time, where NLP problems have to be solved.

There is another class of studies on spline trajectory planning by the so-called control theoretic splines [2],[12], where optimal control problems are solved using linear control systems as spline generator [4],[5],[13]. This approach has an advantage that, by a proper choice of the system, splines can be constructed from wider class of functions including polynomials, exponentials, trigonometric functions and their combinations. However, imposing inequality constraints over an interval of time, as treated here, is not easy and has not been studied.

On the other hand, the authors have developed a framework for constructing smoothing splines with various types of constraints (e.g., [14]–[17]). The splines are constructed using B-splines as the basis functions and developed for splines of arbitrary degrees [18],[19]. The types of the constraints include those at isolated time instants, those over an interval of time, those on function values as well as time derivatives of arbitrary degrees, those on integral values, and so forth. Such constraints can be of equality or inequality, and any combination of constraints, as long as they are consistent, can be incorporated systematically into the smoothing splines settings and formulated as convex quadratic programming (QP) problems.

In this paper, we consider a problem of trajectory planning for road-like path on a two-dimensional plane [20],[21]. Here our B-spline based approach on constrained splines is very ef-
fective. The boundaries of the path are assumed to be piecewise linear and specified by a series of pairs of right and left corner points. Our basic problem is then to construct a smooth trajectory within the path, which can be treated by inequality constraints, while satisfying initial and final conditions. Although such a problem of planning trajectories for path with boundaries naturally arises, the treatment as in this study seems novel to the authors’ knowledge. The paths with piecewise linear boundaries are frequently encountered, but they can also be used as a first approximation for paths with nonlinear boundary curves. The importance of this problem can further be justified by the fact that, in its present form, is considerably general, the formulation is concise, has further extendibility, and is sound theoretically. An obstacle avoidance problem can also be treated by modeling the obstacles by polygons. The spline can be of arbitrary degree in our B-spline approach unlike in existing literatures, and the concise description allows us to extend the theory, e.g., to higher dimensions.

The basic issue of the present problem is to keep the trajectory within the path, which we realize by treating the trajectory piecewise in accordance with each piece of right and left boundary pairs. The description of the problem is easy since the boundaries can be defined by simply providing a series of pairs of right and left corners. The corner points considered in [20],[21] are extended to include double points so that more flexible treatments of sharp corners in particular are allowed in the path. We show that the problem is formulated as a strictly convexQP problem, and can be solved efficiently by existing QP solver. It is noted that the method in [20] for the simple corner points has been extended so as to include a velocity and acceleration constraint in [21], yielding a QCQP (quadratically constrained QP) problem.

The usefulness of the proposed method is confirmed by three examples: first, for relatively complex path with piecewise linear boundaries, second, for application to obstacle avoidance trajectory planning problem, and third, for application to path with non piecewise linear boundaries by approximation.

This paper is organized as follows. In Section 2, we present problem statement and, in Section 3, we describe 2-dimensional vector smoothing splines based on B-splines. Then in Section 4, the trajectory planning problem is formulated and solved. Three numerical examples are considered in Section 5. Concluding remarks are given in Section 6.

We use the following symbols throughout the paper: The symbol ⊗ denotes the Kronecker product, and ‘vec’ the vector-function, i.e., for a matrix $A = [a_1\, a_2 \ldots\, a_n] \in \mathbb{R}^{m \times n}$ with $a_i \in \mathbb{R}^m$, vec $A = [a_1^T\, a_2^T \ldots\, a_n^T] \in \mathbb{R}^{mn}$ (see, e.g., [22]). The inequalities $Q > 0$ and $Q \geq 0$ denote that a real symmetric matrix $Q$ is positive- and nonnegative-definite respectively. Vector norms are $||u|| = \sqrt{u^Tu}$ and $||u||_Q = \sqrt{u^TQu}$ for $u \in \mathbb{R}^n$ and $Q \succeq 0$ with $Q \in \mathbb{R}^{n	imes n}$. Finally tr $A$ denotes the trace of a square matrix $A$.

2. Problem Statement

First we introduce some notations and assumptions, and then state our problem. For planning trajectory $p(t) \in \mathbb{R}^2$ in the xy-plane, let the time interval be given as $[t_0, t_m]$ together with the initial and final conditions on $p(t_0)$ and $p(t_m)$.

We consider road-like path on the xy-plane, which is assumed to possess piecewise linear boundaries. Let such a path be specified by pairs of right and left corner points $(R_i, L_i)$, $i = 0, 1, \ldots, n$ on the xy-plane (see Fig. 1). Thus the boundaries of the path are the polygonal lines connecting the $n + 1$ points $R_0, R_1, \ldots, R_n$ on one side and the $n + 1$ points $L_0, L_1, \ldots, L_n$ on the other. Here the corner point needs not be distinct but its multiplicity is limited up to two as, e.g., $R_i = R_{i+1}$ with $R_i \neq R_{i-1}$ and $R_{i+1} \neq R_{i+2}$.

Without loss of generality, we assume that the initial and final positions, $p(t_0)$ and $p(t_m)$, are located between the points $R_0, L_0$ and $R_n, L_n$ respectively, i.e., $p(t_0) = \gamma_0 R_0 + (1 - \gamma_0) L_0$ and $p(t_m) = \gamma_n R_n + (1 - \gamma_n) L_n$ for some $\gamma_0, \gamma_n (0 \leq \gamma_0, \gamma_n \leq 1)$.

Now we are in the position for stating our problem. Our problem is to plan a smooth trajectory $p(t) \in \mathbb{R}^2$ in the xy-plane,

$$p(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}, \quad t \in [t_0, t_m]$$ (1)

such that the given initial and final conditions as well as the path constraint $p(t) \in \mathcal{P} \forall t \in [t_0, t_m]$ are satisfied, where $\mathcal{P} \subset \mathbb{R}^2$ is the polygon formed from the points $R_0, \ldots, R_n, L_0, \ldots, L_n$. (Thus the trajectory remains within the path during the entire time interval.) Moreover, the trajectory $p(t)$ is constructed by using polynomial spline of degree $k$ given arbitrarily.

We construct spline trajectories $p(t) \in \mathbb{R}^2$ employing vector smoothing splines in Section 3, and the path constraint is then considered in Section 4.

3. Vector Smoothing Splines

The trajectory $p(t)$ for $t \in [t_0, t_m]$ is constructed as polynomial splines using uniform binomial B-spline $B_k(t)$ of degree $k(\geq 1)$ as the basis functions, namely,

$$p(t) = \sum_{i=k}^{m-1} \tau_i B_k(\alpha(t-t_i)).$$ (2)

Here $\tau_i \in \mathbb{R}^2$ are weighting coefficients called control points, $m$ is an integer specifying the number of control points, $t_i$ are equally spaced knot points, and $\alpha (> 0)$ is a constant for scaling the knot point interval by

$$t_{i+1} - t_i = \frac{1}{\alpha}.$$ (3)

Note that, for fixed time interval $[t_0, t_m]$, we can design more complex curves by setting larger $m$ or equivalently smaller knot point interval.

In (2), the B-spline $B_k(t)$ is defined by
Here version \[22\] of JG for numerical solutions. First, let \( \hat{t} \) be derived recursively by de Boor’s algorithm \[18\] for any integer \( l \) of the path as will be described in the next section. We take the spline \((s_i, 0)\) are given set of discrete-time data \[23\].

When the design parameters such as \( k, m, \alpha \) are fixed, our task is to determine the control points \( \tau_i \), or the control point matrix \( \tau \in \mathbb{R}^{2m} (M = m + k) \),

\[
\tau = [\tau_{-4}, \tau_{-4+1}, \ldots, \tau_{m-1}].
\]

Such a trajectory \( p(t) \in \mathbb{R}^2 \) is constructed as vector smoothing splines obtained by minimizing the following cost function:

\[
J(\tau) = \lambda \int_{t_0}^{t_m} \| p(t) \|^2 \, dt + \int_{t_0}^{t_m} \| p(t) - f(t) \|^2 \, dt.
\]  (7)

Here \( \lambda (> 0) \) is a smoothing parameter, and \( \Lambda \in \mathbb{R}^{2 \times 2} \) is a weighting matrix satisfying \( \Lambda > 0 \). The function \( f(t) \in \mathbb{R}^2, t \in [t_0, t_m] \), is assumed to be given, and by minimizing \( J(\tau) \) in (7), we are constructing a smooth spline \( p(t) \) so as to approximate the function \( f(t) \). Typically \( f(t) \) is set up from the ‘centerline’ of the path as will be described in the next section. We take the integer \( l \) as \( l = 2 \) for cubic spline \( (k = 3) \) and \( l = 3 \) for quintic spline \( (k = 5) \).

Note that, in the typical smoothing spline problems, the term like \( \sum_{i=1}^{N} \| p(s_i) - f(s_i) \|^2 \) is used as the second term in (7), where \( (s_i, f_i) \) are given set of discrete-time data \[23\].

In the sequel, we briefly review the solution for this optimization problem in the form that can be readily incorporated for numerical solutions. First, let \( \hat{\tau} \in \mathbb{R}^{2m} \) be the vectorized version \[22\] of \( \tau \), i.e.,

\[
\hat{\tau} = \text{vec} \, \tau.
\]  (8)

Then, \( J(\tau) \) in (7) is expressed as a quadratic function \( J(\hat{\tau}) \) in \( \hat{\tau} \) (see, e.g., \[24\]),

\[
J(\hat{\tau}) = \hat{\tau}^T G \hat{\tau} - 2 \hat{\tau}^T g + g_c,
\]  (9)

where \( G \in \mathbb{R}^{2m \times 2m}, g \in \mathbb{R}^{2m} \), and \( g_c \in \mathbb{R} \) are given by

\[
G = \underbrace{Q \otimes \Lambda + Q_0 \otimes I_2}_{G},
\]

\[
g = \int_{t_0}^{t_m} b(t) \otimes f(t) \, dt,
\]

\[
g_c = \int_{t_0}^{t_m} \| f(t) \|^2 \, dt.
\]  (12)

Here \( Q, Q_0 \in \mathbb{R}^{M \times M} \) are Gram matrices defined by

\[
Q = \int_{t_0}^{t_m} \frac{d^2 b(t)}{dt^2} \frac{d^2 b(t)}{dt^2}^T \, dt,
\]

\[
Q_0 = \int_{t_0}^{t_m} b(t) b(t)^T \, dt,
\]  (13)

and \( b(t) \in \mathbb{R}^m \) is a vector of shifted B-splines defined by

\[
b(t) = \begin{bmatrix} B_1(\alpha(t - t_0)) & B_2(\alpha(t - t_{k+1})) & \cdots & B_{k}(\alpha(t - t_{m-1})) \end{bmatrix}^T.
\]  (15)

In actual computations, we construct the matrix \( Q \) in (13) by

\[
Q = \alpha^{2m-1} R, \quad R = \int_{t_0}^{t_m} \hat{b}^{(i)}(t) \hat{b}^{(i)}(t)^T \, dt,
\]  (16)

where \( \hat{b}(t) \) is defined by

\[
\hat{b}(t) = \begin{bmatrix} B_1(t - (k)) & B_2(t - (k + 1)) & \cdots & B_{l}(t - (m - 1)) \end{bmatrix}^T.
\]  (17)

Note that \( \hat{b}(t) \) does not involve \( \alpha \) and \( t \), unlike \( b(t) \), and \( R \) is a banded matrix with its elements easily set up using \( B_k(t) \) \[25\]. The matrix \( Q_0 \) in (14) can be computed similarly by letting \( l = 0 \) in (16).

**Remark 1** Since \( Q_0 > 0 \) \[24\] and hence \( G > 0 \) in (10), we see that the function \( J(\hat{\tau}) \) in (9) is strictly convex in \( \hat{\tau} \).

### 4. Trajectory Planning

The entire path can be decomposed into pieces of path constrained by a pair of right and left boundary line segments (see Fig. 1), and the trajectory \( p(t) \) can be planned for each piece of the path. The construction of \( p(t) \) in (2) is very suitable for this purpose since \( p(t) \) itself is a piecewise polynomial with the knot points \( t_i \).

We examine two cases as the corner types of the path, say (i) a simple case and (ii) a multiple case. In the case (i), any consecutive points are distinct for both right and left corners \( R_i \) and \( L_i \) as shown in Fig. 2, and we introduce right and left boundary lines \( l_i \) and \( l'_i \) as straight lines passing the two points \( R_i \) and \( R_{i+1} \), and \( L_i \) and \( L_{i+1} \), respectively. Then we plan the trajectory \( p(t) \) so that it lies between the lines \( l_i \) and \( l'_i \) for all \( t \) in a specified time interval \([s_i, s_{i+1}]\). This interval \([s_i, s_{i+1}]\) is taken as a knot point interval \([t_i, t_{i+1}]\).

In the case (ii), two points can coincide, e.g., \( R_i = R_{i+1} \), as shown Fig. 3. For convenience, however, we restrict the multiplicity to two so that \( R_i \neq R_{i-1} \) and \( R_{i+1} \neq R_{i+2} \). Moreover,
we assume that the number of right and left corner points is the same as given \( R_i \) and \( L_i \) for \( i = 0, 1, \ldots, n \) counting multiplicities in the case (ii).

In the sequel, we describe the condition for \( p(t) \) to lie between the two boundary lines \( l_i \) and \( l'_i \) for all \( i \) in \([s_i, s_{i+1}]\) in Section 4.1, a method of assigning \([s_i, s_{i+1}]\) as a knot point interval \([t_i, t_{i+1}]\) in Section 4.2, and the spline construction procedure will be given in Section 4.3, e.g., taking initial and final conditions into account.

### 4.1 Trajectory between Two Lines

For convenience of description, we first consider the case (i) where all the corner points are distinct as shown in Fig. 2. Then, it is natural to introduce the following assumptions (see Fig. 4).

(A1) The polygon \( P_i = R_iR_{i+1}L_{i+1}L_i \) is a convex quadrangle for all \( i \).

(A2) In \( P_i \), the vertices \( R_i, R_{i+1}, L_{i+1}, L_i \) are located counterclockwise.

By (A1), the line segment \( R_iR_{i+1} \) does not intersects with \( L_iL_{i+1} \). By (A2), the quadrangle \( P_i \) constitutes part of the path with piecewise linear boundaries, and the entire path is the union of the quadrangles.

We derive a condition such that \( p(t) \) remains in a region between the two lines \( l_i \) and \( l'_i \) for all \( i \) in an interval \([s_i, s_{i+1}]\). Note that we take this interval as a knot point interval so that we let \([s_i, s_{i+1}] = [t_i, t_i']\) for some integers \( t_i, t_i' \) (\( t_i < t_i' \)). Here and hereafter, by the term ‘region between the two lines \( l_i \) and \( l'_i \)’, we mean the region between the two lines \( l_i \) and \( l'_i \) including the quadrangle \( P_i \). Thus if \( l_i \) and \( l'_i \) intersect, then among the two wedge-shaped regions, the one including \( P_i \) is meant.

Since the lines \( l_i \) and \( l'_i \) respectively play the roles of right and left boundaries of the path as in Fig. 4, the condition for a point \( p = p(t) \) to lie between the two lines \( l_i \) and \( l'_i \) can be stated as

\[
\text{(C1) the point } p \text{ to lie to the left of } l_i \text{ when viewed from } R_i \text{ toward } R_{i+1} \text{ along } l_i, \text{ and}
\]

\[
\text{(C2) the point } p \text{ to lie to the right of } l'_i \text{ when viewed from } L_i \text{ toward } L_{i+1} \text{ along } l'_i.
\]

The conditions (C1) and (C2) can then be written as

\[
\begin{align*}
\overrightarrow{R_i p} \times \overrightarrow{R_{i+1} p} & \leq 0, 
\text{(18)} \\
\overrightarrow{L_i p} \times \overrightarrow{L_{i+1} p} & \geq 0, 
\text{(19)}
\end{align*}
\]

respectively. Here \( \times \) denotes cross product of 2-dimensional vectors, and it holds that, for \( u = [u_1, u_2]^T \) and \( v = [v_1, v_2]^T \)

\[
\begin{align*}
\overrightarrow{R_i p} \times \overrightarrow{R_{i+1} p} &= \begin{vmatrix}
R_{i+1} - R_i & p - R_i \\
R_{i+1} - R_i & R_{i+1} - p \\
\end{vmatrix} \\
&= R_i[1, v_2, -v_1]^T - R_{i+1}[1, v_2, -v_1]^T \\
&= \begin{vmatrix}
1 & v_2 & -v_1 \\
1 & v_2 & -v_1 \\
\end{vmatrix} \\
&= v_1 - v_2 \\
&= u_1 v_2 - u_2 v_1.
\end{align*}
\]

\[
\begin{align*}
\overrightarrow{L_i p} \times \overrightarrow{L_{i+1} p} &= \begin{vmatrix}
L_{i+1} - L_i & p - L_i \\
L_{i+1} - L_i & L_{i+1} - p \\
\end{vmatrix} \\
&= L_i[1, u_2, -u_1]^T - L_{i+1}[1, u_2, -u_1]^T \\
&= \begin{vmatrix}
1 & u_2 & -u_1 \\
1 & u_2 & -u_1 \\
\end{vmatrix} \\
&= u_1 - u_2 \\
&= u_1 v_2 - u_2 v_1.
\end{align*}
\]

Applying this relation to the vectors in (18) and (19), for example \( \overrightarrow{R_i p} = [x - x_i, y - y_i]^T \), the conditions (18) and (19) can be rewritten as linear inequalities in \( p = [x, y]^T \) as follows:

\[
\begin{align*}
Ap & \leq d, 
\text{(21)}
\end{align*}
\]

where

\[
A = \begin{bmatrix}
y_i + y_{i+1} & x_i - x_{i+1} \\
y'_i - y'_{i+1} & -x'_i + x'_{i+1}
\end{bmatrix},
\]

\[
d = \begin{bmatrix}
x_i(y_{i+1} - y'_i) - y_i(x'_{i+1} - x'_i) \\
x'_i(y_{i+1} - y'_i) - y'_{i+1}(x'_{i+1} - x'_i)
\end{bmatrix}.
\]

Now we consider the case (ii) of double corner points, where the quadrangle \( P_i \) degenerates to a triangle. For right double points with \( R_i = R_{i+1} \), we take the right border line \( l_{i}' \) as the one parallel to the left border line \( l'_{i} \) among those passing the point \( R_i \), i.e., in the direction of vector \( L_{i}L_{i+1} \) (see Fig. 3). Obviously \( L_i \neq L_{i+1} \) is assumed. Hence for the point \( p = p(t) \) to lie between the two lines \( l_i \) and \( l'_i \), we simply replace the condition (18) by

\[
\begin{align*}
\overrightarrow{R_i p} \times \overrightarrow{R_{i+1} p} & \leq 0, 
\overrightarrow{L_i p} \times \overrightarrow{L_{i+1} p} & \geq 0,
\end{align*}
\]

so that the first row of \( A \) and \( d \) in (21) should be modified according to

\[
\begin{align*}
\begin{bmatrix}
y_i + y_{i+1} & x_i - x_{i+1} \\
y'_i - y'_{i+1} & -x'_i + x'_{i+1}
\end{bmatrix} \\
x_i(y_{i+1} - y'_i) - y_i(x'_{i+1} - x'_i)
\end{bmatrix} \\
\begin{bmatrix}
x'_i(y_{i+1} - y'_i) - y'_{i+1}(x'_{i+1} - x'_i)
\end{bmatrix}.
\]

The case of left double corner point \( L_i = L_{i+1} \) can be treated similarly. Namely we define the left border line \( l'_i \) as parallel to the right border line \( l_i \), and then the condition (19) is replaced by

\[
\begin{align*}
\overrightarrow{L_i p} \times \overrightarrow{L_{i+1} p} & \geq 0, 
\end{align*}
\]

Thus the second row of \( A \) and \( d \) in (21) becomes

\[
\begin{align*}
\begin{bmatrix}
y_i - y_{i+1} & (-x_i + x_{i+1}) \\
y'_i - y'_{i+1} & -x'_i + x'_{i+1}
\end{bmatrix} \\
x_i(y_{i+1} - y'_i) + y_i(x'_{i+1} - x'_i)
\end{bmatrix} \\
\begin{bmatrix}
x'_i(y_{i+1} - y'_i) + y'_{i+1}(x'_{i+1} - x'_i)
\end{bmatrix}.
\]

Now that we derived the condition for a point \( p \) to lie between the lines \( l_i \) and \( l'_i \), next we consider the condition such that \( p(t) \) given by (2) stays between the two lines \( l_i \) and \( l'_i \) for all \( t \) in the knot point interval \([t_i, t_{i+1}]\). By (21), it suffices to derive the condition for \( Ap(t) \leq d \) for all \( t \in [t_i, t_{i+1}] \).

**Lemma 1** The trajectory \( p(t) \) lies between the two lines \( l_i \) and \( l'_i \) for all \( t \) in a unit knot point interval \([t_i, t_{i+1}]\) if the control point \( \tau \) satisfies
\[ A \tau_i \leq d, \quad j - k \leq i \leq j, \quad (28) \]
and, accordingly, it lies between the two lines for all \( t \) in a knot point interval \([t_i, t_{i+1}]\) \((k < \mu)\) if \( \tau_i \) satisfies
\[ A \tau_i \leq d, \quad k - k \leq i \leq \mu - 1. \quad (29) \]

(Proof) First note that, by the definition of \( B_2(t) \) in (4), \( p(t) \) is written for unit knot point interval \([t_i, t_{i+1}]\) as
\[ p(t) = \sum_{j=0}^{k} \tau_{j-k+1} N_{j,k}(a(t-t_j)), \quad t \in [t_i, t_{i+1}], \quad (30) \]
and it depends on only the \( k + 1 \) weights \( \tau_{j-k}, \tau_{j-k+1}, \ldots, \tau_j \). Introducing a new variable \( u = \alpha(t-t_j) \), we see that \([t_i, t_{i+1}]\) in \( t \) is normalized to \([0, 1]\) in \( u \) and \( p(t) \) is written as \( p(u) \),
\[ \hat{p}(u) = \sum_{j=0}^{k} \tau_{j-k+1} N_{j,k}(u), \quad u \in [0, 1]. \quad (31) \]
Thus if (28), or \( A \tau_i \leq d \) for \( i = 0, 1, \ldots, k \) holds, then using (5), we get
\[ A \hat{p}(u) = \sum_{j=0}^{k} A \tau_{j-k+1} N_{j,k}(u) \leq d \sum_{j=0}^{k} N_{j,k}(u) = d, \quad (32) \]
and hence \( A \hat{p}(t) \leq d \) for all \( t \in [t_i, t_{i+1}] \).

This result can be used for the intervals \([t_i, t_{i+1}], [t_{i+1}, t_{i+2}], \ldots, [t_{i-1}, t_i] \), and we obtain the condition (29) for the interval \([t_i, t_{i+1}] \). Namely if (29) is satisfied, then it holds that
\[ A \hat{p}(t) \leq d \forall t \in [t_i, t_{i+1}], \quad (33) \]
(QED)

Now the remaining task is to express the inequality condition (29) in terms of the control point vector \( \hat{r} \) defined in (8). Let \( I_1 = [1 \ldots 1] \in \mathbb{R}^i \).

**Lemma 2** The trajectory \( p(t) \) lies for all \( t \in [s_i, s_{i+1}] = [t_i, t_{i+1}] \) between the two lines \( l_i \) and \( l'_{i+1} \) if the control point vector \( \hat{r} \) satisfies the following inequality:
\[ F_i \hat{r} \leq h_i. \quad (34) \]
Here \( F_i \in \mathbb{R}^{2(i-\nu) \times 2M} \) and \( h_i \in \mathbb{R}^{2(i-\nu)} \) are given by
\[ F_i = E_{i,\nu} \otimes A, \quad h_i = l_{i-\nu} \otimes d, \quad (35) \]
and \( E_{i,\nu} \in \mathbb{R}^{M(i-\nu)} \) is defined by
\[ E_{i,\nu} = \left[ \begin{array}{cc} 0_{i-\nu, i-\nu} & I_{i-\nu, 0} \end{array} \right] \in \mathbb{R}^{M(i-\nu) \times M(i-\nu)}. \quad (36) \]
(Proof) First rewrite (29) as
\[ A \hat{r} \leq 1_{i-\nu} \otimes d \quad (37) \]
with \( T_{i,\nu} = \left[ \begin{array}{c} T_{i-\nu} \quad \ldots \quad T_{i-1} \end{array} \right] \). Then noting that \( T_{i,\nu} \) is a submatrix of \( \tau \) consisting of its columns from \( k + 1 \) through \( \mu + k \), it can be expressed as
\[ T_{i,\nu} = \tau E_{i,\nu}. \quad (38) \]
Thus (37) is written in \( \tau \) as
\[ A \tau E_{i,\nu} \leq 1_{i-\nu} \otimes d. \quad (39) \]
Using a formula \( \text{vec}(AXB) = (B^T \otimes A) \text{vec}X \) for matrices \( A, B, X \) of compatible dimensions (see, e.g., [22]), and noting vec \( \tau = \hat{r} \), (39) yields
\[ (E_{i,\nu} \otimes A) \hat{r} \leq 1_{i-\nu} \otimes d, \quad (40) \]
which is the desired expression. (QED)

**4.2 Centerline and Intermediate Time Instants**

We introduce ‘centerline’ of the path for constructing smoothing splines \( p(t) \) as approximation of a function \( f(t) \) in (7) and for specifying the intermediate time instants \( s_i, i = 0, 1, \ldots, n \).

Let points \( C_i, \quad i = 0, 1, \ldots, n \) be defined by
\[ C_i = \gamma_i R_i + (1 - \gamma_i) L_i \quad (41) \]
for some \( \gamma_i \left( 0 \leq \gamma_i \leq 1 \right) \), usually set as \( \gamma_i = 1/2 \) (see Fig. 5).

Then we call the piecewise linear line \( C_1 C_2 \ldots C_n \) as centerline, and \( f(t) \) is taken as a linear function in time \( t \) along the centerline. Specifically, \( f(t) \) is constructed in each time interval \([s_i, s_{i+1}] \) as
\[ f_i(t) = q_i t + r_i, \quad t \in [s_i, s_{i+1}], \quad (42) \]
where \( q_i, r_i \in \mathbb{R}^2 \) are determined so that
\[ f_i(s_i) = C_i, \quad f_i(s_{i+1}) = C_{i+1}. \quad (43) \]

**Remark 2** The centerline needs not exist actually. This concept is used to constitute the function \( f(t) \) for approximation by \( p(t) \) in (7). It can be defined from pairs of corner points \( (R_i, L_i) \) and can be adjusted by the parameter \( \gamma_i \) in (41). Roughly, setting smaller \( \gamma_i \), for example, implies route keeping more left in the path, while \( \gamma_i = 1/2 \) tends to keep the center.

Finally, we need to allocate the given entire time interval \([0, t_m] \) to \( n \) knot point intervals \([s_i, s_{i+1}] \) \((i = 0, 1, \ldots, n - 1) \), where each \( s_i \) is selected from the knot points \([0, t, \ldots, t_m] \) with \( s_0 = t_0 \) and \( s_n = t_m \). It is natural to take the length of center line \( C_i C_{i+1} \) into account, and here we employ the so-called centripetal distribution [1]. Specifically the interval \([0, t_m] \) is divided into \( n \) subintervals \([s_i, s_{i+1}] \) \((i = 0, 1, \ldots, n - 1) \) in proportion to the following value \( \xi_i \),
\[ \xi_i = \|C_{i+1} - C_i\|^\nu, \quad (44) \]
where \( \nu (0 < \nu < 1) \) is often taken as \( \nu = 1/2 \). Usually the resulting \( s_i \) does not coincide with any of the knot points and, noting that \( m \gg n \), we can replace \( s_i \) by the smallest knot point \( t_j \) that exceeds the original value of \( s_i \). It is noted that such a centripetal distribution method requires less accelerations than other methods, e.g., cord length distribution, distributed proportionally to the distance \( \|C_{i+1} - C_i\|\).

**4.3 Smoothing Spline Trajectory**

We are now in the position to formulate our trajectory planning problem in the form ready for numerical solutions. First, the constraints on the trajectory \( p(t) \) resulting from the piecewise linear boundaries are described as a collection of linear inequalities derived for each piece of the path. Namely by Lemma 2, we impose
\[ F \hat{\tau} \leq h_i, \quad i = 0, 1, \ldots, n - 1. \] (45)

Then, the other constraints can be included as desired. The initial and terminal conditions of \( p(t) \) are given, for example as
\[
\begin{align*}
p(t_0) &= p_0, \quad p^{(1)}(t_0) = 0, \quad p^{(2)}(t_0) = 0, \\
p(t_m) &= p_m, \quad p^{(1)}(t_m) = 0, \quad p^{(2)}(t_m) = 0
\end{align*}
\]
(46) and (47)

with \( p_0 \) and \( p_m \) lying on the line segments \( R_0L_0 \) and \( R_mL_m \) respectively. In terms of \( \hat{\tau} \), these conditions can be expressed as follows [24]:
\[
H(t)\hat{\tau} \leq h_0, \quad H(t_m)\hat{\tau} = \hat{h}_m,
\]
where \( H(t) \in \mathbb{R}^{6 \times 2M}, \hat{h}_0 \in \mathbb{R}^6 \), and \( \hat{h}_m \in \mathbb{R}^6 \) are
\[
H(t) = \begin{bmatrix}
b(t)^T & I_2 \\
b^{(1)}(t)^T & I_2 \\
b^{(2)}(t)^T & I_2 \\
\end{bmatrix}, \quad \hat{h}_0 = \begin{bmatrix} p_0 \\
0_2 \\
0_2 \\
\end{bmatrix}, \quad \hat{h}_m = \begin{bmatrix} p_m \\
0_2 \\
0_2 \\
\end{bmatrix}.
\]
(48)

We can set up the matrices \( H(t_0) \) and \( H(t_m) \), for example, by using Table 1 when \( k = 3 \). For \( H(t_0) \), noting that \( a(t_0 - t_i) = -\iota_i \) by (3), we get
\[
b^{(i)}(t_0) = a_i' \begin{bmatrix} B_i^{(0)}(3) & B_i^{(1)}(2) & B_i^{(2)}(1) \\
\vdots & \ddots & \vdots \\
B_i^{(0)}(m - 1) \\
\end{bmatrix} \hat{\tau}.
\]
(50)

Hence using (4) and Table 1, the matrix \( H(t_0) \) can be constructed by
\[
b(t_0) = \begin{bmatrix} 1/6 & 4/6 & 1/6 & 0^{T}_{M-3} \\
-1/2 & 0 & 1/2 & 0^{T}_{M-3} \\
1 & -2 & 1 & 0^{T}_{M-3} \\
\end{bmatrix} \hat{\tau},
\]
\[
b^{(1)}(t_0) = a_1' \begin{bmatrix} -1/2 & 0 & 1/2 & 0^{T}_{M-3} \\
1 & -2 & 1 & 0^{T}_{M-3} \\
\end{bmatrix} \hat{\tau},
\]
\[
b^{(2)}(t_0) = a_2' \begin{bmatrix} 1/6 & 4/6 & 1/6 & 0^{T}_{M-3} \\
-1/2 & 0 & 1/2 & 0^{T}_{M-3} \\
\end{bmatrix} \hat{\tau}.
\]
(51)

Our trajectory planning problem is then formulated as constrained smoothing splines with the cost function \( J(\hat{\tau}) \) in (9) and the constraints from the boundary inequality conditions in (45) and the initial and terminal conditions in (48). Namely the problem is to minimize the cost function,
\[
\min_{\hat{\tau} \in \mathbb{R}^n} J(\hat{\tau}) = \frac{1}{2} \hat{\tau}^T G \hat{\tau} - g^T \hat{\tau}
\]
(52)

subject to the constraints of the form
\[
A_{eq} \hat{\tau} = d_{eq}, \quad A_{in} \hat{\tau} \leq d_{in},
\]
subject to the constraints of the form
\[
A_{eq} \hat{\tau} = d_{eq}, \quad A_{in} \hat{\tau} \leq d_{in},
\]
where \( G \) and \( g \) are given in (10) and (11) respectively, \( A_{eq} \) and \( d_{eq} \) are formed as the collection of equalities (48), and \( A_{in} \) and \( d_{in} \) as collection of inequalities in (45).

Note that \( A_{eq} \hat{\tau} = d_{eq} \) by (48) consists of 12 equality constraints in \( 2M \) unknowns \( \hat{\tau} \), whereas \( A_{in} \hat{\tau} \leq d_{in} \) by (45) contains \( 2M + 2(m - 1)k \) linear inequality constraints.

This is a strictly convex QP problem and efficient numerical solver is available, e.g., the function ‘quadprog’ in MATLAB. If necessary, other methods may be introduced such as constraints on the magnitude of velocity or acceleration as long as all the constraints are consistent.

Finally, note that, by our construction and Lemma 2, the planned trajectory possesses the following property. This is the main theoretical result of this paper. Here recall from Section 4.1 that the assertion ‘the trajectory lies between the two lines \( l_i \) and \( l'_i \) should be understood as ‘the trajectory lies in the region between the two lines \( l_i \) and \( l'_i \) including the polygon \( P_i = R_iR_{i+1}L_{i+1}L_i \).’

\textbf{Theorem 1} The spline trajectory \( p(t) \) in (2) planned by solving the QP problem formulated as (52) and (53) with the constraints in (45) is guaranteed to lie, for each \( i = 0, 1, \ldots, n - 1 \), between the two lines \( l_i \) and \( l'_i \) for all time \( t \in [s_i, s_{i+1}] \). In particular, \( p(t) \) is in the corner quadrangle formed from the four lines \( l_i, l'_i, l_{i-1}, l'_{i-1} \) at time \( t = s_i \) for \( i = 1, 2, \ldots, n - 1 \).

\section{Numerical Examples}

We consider three examples of trajectory planning, where the first two are for paths with piecewise linear boundaries and the third one with circular boundaries. In the third case, the proposed method is applied by approximating circles with piecewise linear functions.

In all the cases, we use cubic splines, and we set \( k = 3 \) in (2). QP problems are solved numerically by employing MATLAB function ‘quadprog’.

\subsection{5.1 Path with Piecewise Linear Boundaries}

Table 2 shows the coordinates of the right and left corners \( R_i \) and \( L_i \) with \( n = 13 \). This path contains two right double points at \( R_1 = R_4 = (14, 13)^T \) and \( R_7 = R_{10} = (14, 5)^T \) and one left double point at \( L_3 = L_6 = (7, 9)^T \).

The parameters used for planning the trajectory \( p(t) \) are as follows: The entire time interval is set as \([t_0, t_m] = [0, 10] \); the initial and final conditions are \( p_0 = (R_0 + L_0)/2 \) and \( p_m = (R_6 + L_6)/2 \) in (46) and (47) respectively; centerline parameters are \( \alpha = 1/2 \) and \( \gamma \) in (41); the number of knot points is \( m = 80 \) (hence the knot point interval is \( t_{i+1} - t_i = 10/80 = 0.125 \)); and finally, the smoothing parameter is \( \lambda = 0.01 \) in (7).

In Fig. 6, the constructed trajectory \( p(t) = (x(t), y(t)) \) is shown in thick solid line on the \( xy \) plane. The boundaries of the given path are shown by thin solid lines, the start and goal positions are denoted by \( \times \) and \( \circ \) respectively, and the center line is denoted by dotted line. Also plotted in the dashed line is the trajectory \( p_2(t) = (x_2(t), y_2(t)) \) constructed similarly as \( p(t) \) but without the boundary constraints (45). We see that \( p_2(t) \) is exceeding the boundaries at five corners, while the trajectory \( p(t) \) satisfies the boundary constraints in addition to initial and final conditions, and the effect of introducing the inequality constraints is apparent.

The corresponding trajectories \( x(t) \) and \( y(t) \) are plotted in Fig. 7(a) in thick solid and dotted lines respectively, and the trajectories \( x_2(t) \) and \( y_2(t) \) in thin solid and dotted lines.

\subsection{5.2 Path in Obstacle Avoidance Problem}

We apply our method to trajectory planning in obstacle avoidance problems of mobile robots. As an example, we consider an environment as shown in Fig. 8, where dark rectangles denote ‘obstacles’, and \( \times \) and \( \circ \) denote the start and goal respectively. We introduce the piecewise linear path given by the right and left corner points as defined in Table 3. In Fig. 8, these points are shown by \( \triangle \) and \( \bigtriangledown \) with the boundary lines plotted in dotted lines. Notice that this path contains one double corner at \( L_2 = L_3 = (6, 10)^T \). Although other choices of the path are possible, this will be a reasonable choice once we decide to take the upper route of the table-like obstacle in the figure.
Table 2 Right and left corners $R_i$ and $L_i$ for $i = 0, 1, \ldots, n$ with $n = 13$.

| $i$ | $R_i$ | $L_i$ |
|-----|-------|-------|
| 0   | 0     | 0     |
| 1   | 4     | 2     |
| 2   | 14    | 2     |
| 3   | 14    | 9     |
| 4   | 5     | 9     |
| 5   | 5     | 5     |
| 6   | 5     | 0     |
| 7   | 2     | 0     |
| 8   | 13    | 13    |
| 9   | 10    | 8     |
| 10  | 8     | 5     |
| 11  | 5     | 5     |
| 12  | 3     | 3     |
| 13  | 2     | 2     |

Fig. 6 Spline trajectories $(x(t), y(t))$ and $(x_2(t), y_2(t))$ in the $xy$ plane constructed with and without the boundary constraints in (45), respectively.

Fig. 7 Position (a), velocity (b) and acceleration (c) profiles of spline trajectories $(x(t), y(t))$ and $(x_2(t), y_2(t))$ constructed with and without the boundary constraints in (45), respectively.

The parameters used for planning the trajectory are as follows: The time interval is set as $[t_0, t_m] = [0, 10]$; the initial and final conditions are $p_0 = (R_0 + L_0)/2$ and $p_m = (R_n + L_n)/2$ in (46) and (47) respectively; the number of knot points is $m = 50$; and the smoothing parameter is $\lambda = 0.1$ in (7). We examine two cases of the centerline parameters $\gamma_i$ in (41) as shown in Table 3. Case (i) sets the line in the middle of the path, whereas Case (ii) sets the line closer to the inner corner of the path at corners $i = 1$ and $i = 4$. Figure 8 shows the constructed trajectories in the $xy$ plane in thick solid line for Case (i) $p(t) = (x(t), y(t))$, and thick dotted line for Case (ii) $p_2(t) = (x_2(t), y_2(t))$. We see that, in both cases, reasonably good trajectories have been obtained, but the second case seems more desirable. Thus the parameters $\gamma_i$ for the center line could be effectively used to adjust the trajectory. Note that the path is defined by only six pairs of right and left corners.

Table 3 Right and left corners $R_i$ and $L_i$ and the centerline parameter $\gamma_i$ for $i = 0, 1, \ldots, n$ with $n = 5$.

| $i$ | $R_i$ | $L_i$ | $\gamma_i$ : Case (i) | $\gamma_i$ : Case (ii) |
|-----|-------|-------|-----------------------|-----------------------|
| 0   | 3     | 0     | 1/2                   | 1/2                   |
| 1   | 3     | 0     | 1/2                   | 1/2                   |
| 2   | 4     | 1     | 1/2                   | 1/2                   |
| 3   | 7     | 6     | 1/2                   | 1/2                   |
| 4   | 13    | 16    | 1/2                   | 1/2                   |
| 5   | 13    | 16    | 1/2                   | 1/2                   |

Fig. 8 Spline trajectories $(x(t), y(t))$ and $(x_2(t), y_2(t))$ in the $xy$ plane constructed for two Cases (i) and (ii) of the centerline (see Table 3).

5.3 Circular Path by Approximation

As the third example, we examine how the proposed method works for a path with nonlinear boundaries. An obvious approach is by approximation of the boundary by piecewise linear functions. For convenience, here we consider circular path with both right and left boundaries described by circles. Let the radius of two circles be $r_1 = 1$ and $r_2 = 0.8$ with their centers at the origin of an $xy$ plane. Considering a trajectory for counterclockwise motion, these circles are approximately...
mated by piecewise linear functions, or conveniently by regular \( n \)-sided polygons. Specifically, the coordinates of right corners \( R_i \) for piecewise linear boundary are computed as vertices of inscribed polygon of the outer circle, and those for left corners \( L_i \) as vertices of circumscribed polygon of the inner circle, or equivalently of the inscribed polygon of a larger circle with the radius \( r_i^2 = r_2 / \cos(\pi/n) \). Then these coordinates are given by

\[
R_i = \begin{bmatrix} r_1 \cos \theta_i \\ r_1 \sin \theta_i \end{bmatrix}, \quad L_i = \begin{bmatrix} r_i^2 \cos \theta_i \\ r_i^2 \sin \theta_i \end{bmatrix}
\]

for \( i = 0, 1, \ldots, n \), where \( \theta_i = 2\pi i/n \). We use \( n = 8 \) and hence \( r_2^2 = 0.8659 \).

For this path as approximation of circular path, a smoothing spline \( p(t) \) is constructed for the time interval \([t_0, t_m]\) = \([0, 10]\), and initial and final conditions are as given in (46) and (47) with \( p_0 = (R_0 + L_0)/2 \) and \( p_m = (R_n + L_n)/2 \). The centerline is given by \( \gamma_i = 1/2 \forall i \) in (41). Moreover we set \( m = 80 \) and the smoothing parameter as \( \lambda = 1 \).

In Fig. 9, the planned trajectory \( p(t) = (x(t), y(t)) \) is shown on the \( xy \)-plane in thick solid line together with the original circular boundaries (thin solid lines) and their approximated piecewise linear boundaries in dash-dot lines with \( \Delta \) and \( \triangledown \) marks for the right and left corners respectively. The trajectory proceeds counter clockwise from the start point on the \( x \)-axis, forming circular curve as we expect. Without the inequality boundary constraint (see dashed curve, denoted by \( p_2(t) = (x_2(t), y_2(t)) \)), the trajectory deviate greatly from the path. Thus the effectiveness is confirmed of introducing the approximate boundary constraints.

Moreover, we confirmed that such a path with circular boundaries can be made narrower, for instance, we could plan a smooth trajectory within the path for the case \( r_1 = 1 \) and \( r_2 = 0.9 \) or further \( r_2 = 0.95 \) by setting, e.g., \( n = 16 \). Thus the proposed method may be used effectively for paths with not necessarily piecewise linear boundaries.

6. Concluding Remarks

We developed a method of trajectory planning for path with piecewise linear right and left boundaries. Such a path is fundamental for representing paths on a plane and can also be used as a first approximation for paths with nonlinear boundary curves. The representation of the boundaries are simple and we only need to provide a series of pairs of right and left corners \( (R_i, L_i) \). Double corner points can be included, allowing more flexible arrangements of corner points for such a path with sharp corners.

We construct the trajectory as vector smoothing splines employing normalized uniform B-splines as the basis functions. Due to the introduction of centerline of the path, the cost function for smoothing splines becomes strictly convex quadratic function of control point vector \( \hat{r} \). It is shown that the boundary constraints are represented as a set of linear inequalities on the vector \( \hat{r} \). This is due to the fact that the boundaries are treated piecewise as a pair of linear inequalities and the trajectory is itself constructed as piecewise polynomial.

The problem is thus formulated as strictly convex QP problem and very efficient numerical solvers are available as MATLAB function quadprog. We confirmed the effectiveness of the proposed method by the three numerical examples. In particular, the second example showed usefulness of the method for treating environment with obstacles by proper settings of the path. The third one with circular boundaries showed that other types of boundaries can also be treated by introducing proper approximations. The computational times required for solving the QP problem in Sections 5.1, 5.2, and 5.3 were respectively 0.174792 s, 0.167705 s for Case (i), and 0.188435 s by the processor Intel Core i7-4810MQ (2.80 GHz).

In conclusion, this paper provides a new method for trajectory planning which is sound in theory as shown in Theorem 1 and is applicable to a wide range of problems as demonstrated in the examples in Section 5. The method is computationally feasible since the problem reduces to strictly convex QP problem, whereas solutions to an NLP problem are required in existing methods.

Important future issues include extensions of the present method to the cases of higher order boundary curves and to the planning in 3-dimensional space for, e.g., aircrafts and aerial robots [4],[26] with airspace constraints. In either case, the central issues will be the concise description of boundary curves or planes for the path and smooth trajectory planning guaranteed to stay within the path. Such generalizations are important since they provide new methods for treating broader class of applications while maintaining theoretical support. Obviously one of the final goals is an application of the proposed method to trajectory planning and control of mobile robots.

References

[1] L. Biagiotti and C. Melchiorri: Trajectory Planning for Autonomous Machines and Robots, Springer, 2008.
[2] M. Egerstedt and C.F. Martin: Control Theoretic Splines: Optimal Control, Statistics and Path Planning, Princeton University Press, 2010.
[3] W. Khalil and E. Dombre: Modeling, Identification and Control of Robots, Hermes Penton, 2002.
[4] P. Crouch and J. Jackson: Dynamic interpolation and application to flight control, J. Guidance, Control and Dynamics, Vol. 14, No. 4, pp. 814–822, 1991.
[5] M. Egerstedt and C.F. Martin: Optimal trajectory planning and smoothing splines, Automatica, Vol. 37, No. 7, pp. 1057–1064, 2001.
[6] J.Z. Kolter and A.Y. Ng: Task-space trajectories via cubic spline optimization, Proc. 2009 Int. Conf. Robotics and Automation, pp. 1675–1682, 2009.
[7] A. Piazzi and A. Visioli: Global minimum-jerk trajectory plan-


Hiroyuki Kano (Member)

He received the B.E. and M.E. degrees in Mechanical Engineering from Kyoto Institute of Technology, Japan, in 1970 and 1972, respectively, and the Ph.D. in Electrical Engineering - Systems Science from Polytechnic Institute of New York (now Polytechnic Institute of New York University), USA, in 1976. From 1976 to 1992, he was with the International Institute for Advanced Study of Social Information Sciences, Fujitsu Limited. In 1992, he joined as a Professor the Department of Information Sciences, Tokyo Denki University, and is currently with the Division of Science. His research interest includes control and estimation theory, robotics, and signal processing. He is a member of the IEEE, ISCIIE, RSJ and JSIAM.

Hiroyuki Fujioka

He received the B.E., M.E., and Dr.E. degrees from Ritsumeikan University, Kyoto, Japan, in 1997, 1999, and 2004, respectively. From 2002 to 2008, he was with the Department of Information Sciences, Tokyo Denki University, Saitama, Japan, where he was an Instructor until 2005 and afterward was promoted to an Assistant Professor. In 2008, he joined as an Assistant Professor the Department of System Management, Fukuoka Institute of Technology, Fukuoka, Japan, and is currently a Professor. His research interests include theory and applications of splines, and task planning and control of robots. He is a member of IEEE, ISCIIE, RSJ and JSIAM. He is a recipient of the 2004 Sunahara Prize and Young Investigators Award from the ISCIIE.

S. Saska, M. Macas, L. Preucil, and L. Lhotska: Robot path planning using particle swarm optimization of Ferguson splines, Proc. IEEE Conf. Emerging Technologies and Factory Automation (ETFA ’06), pp. 833–839, 2006.

T. Berglund, A. Brodnik, H. Jonsson, M. Staffansson, and I. Soderkvist: Planning smooth and obstacle-avoiding B-spline paths for autonomous mining vehicles, IEEE Trans. Automat. Sci. Eng., Vol. 7, No. 1, pp. 167–172, 2010.

P. Gallina and A. Gasparetto: A technique to analytically formulate and to solve the 2-dimensional constrained trajectory planning problem for a mobile robot, J. Intelligent and Robotic Systems, Vol. 27, No. 3, pp. 237–262, 2000.

M. Yamamoto, M. Iwamura, and A. Mohri: Near-time-optimal trajectory planning for mobile robots with two independently driven wheels considering dynamical constraints and obstacles, J. Robotics Society of Japan, Vol. 16, No. 8, pp. 1123–1130, 1998 (in Japanese).

S. Sun, M. Egerst edt, and C.F. Martin: Control theoretic smoothing splines, IEEE Trans. Automat. Control, Vol. 45, No. 12, pp. 2271–2279, 2000.

C.F. Martin, S. Sun, and M. Egerstedt: Optimal control, statistics and path planning, Math. Comput. Modelling, Vol. 33, No. 1-3, pp. 237–253, 2001.

H. Fujioka and H. Kano: Optimal vector smoothing splines with coupled constraints, Trans. Institute of Systems, Control and Information Engineers, Vol. 25, No. 11, pp. 299–307, 2012.

H. Kano, H. Fujioka, and C.F. Martin: Optimal smoothing and interpolating splines with constraints, Applied Mathematics and Computation, Vol. 218, No. 5, pp. 1831–1844, 2011.

H. Kano, H. Fujioka, and C. Martin: Optimal smoothing spline with constraints on its derivatives, SICE Journal of Control, Measurement, and System Integration, Vol. 7, No. 2, pp. 104–111, 2014.

H. Fujioka, H. Kano, and C. Martin: Constrained smoothing and interpolating spline surfaces using normalized uniform B-splines, Communications in Information and Systems, Vol. 14, No. 1, pp. 23–56, 2014.

Hiroyuki Fujioka

He received the B.E., M.E., and Dr.E. degrees from Ritsumeikan University, Kyoto, Japan, in 1997, 1999, and 2004, respectively. From 2002 to 2008, he was with the Department of Information Sciences, Tokyo Denki University, Saitama, Japan, where he was an Instructor until 2005 and afterward was promoted to an Assistant Professor. In 2008, he joined as an Assistant Professor the Department of System Management, Fukuoka Institute of Technology, Fukuoka, Japan, and is currently a Professor. His research interests include theory and applications of splines, and task planning and control of robots. He is a member of IEEE, ISCIIE, RSJ and JSIAM. He is a recipient of the 2004 Sunahara Prize and Young Investigators Award from the ISCIIE.

Hiroyuki Kano 

He received the B.E. and M.E. degrees in Mechanical Engineering from Kyoto Institute of Technology, Japan, in 1970 and 1972, respectively, and the Ph.D. in Electrical Engineering - Systems Science from Polytechnic Institute of New York (now Polytechnic Institute of New York University), USA, in 1976. From 1976 to 1992, he was with the International Institute for Advanced Study of Social Information Sciences, Fujitsu Limited. In 1992, he joined as a Professor the Department of Information Sciences, Tokyo Denki University, and is currently with the Division of Science. His research interest includes control and estimation theory, robotics, and signal processing. He is a member of the IEEE, ISCIIE, RSJ and JSIAM.

S. Saska, M. Macas, L. Preucil, and L. Lhotska: Robot path planning using particle swarm optimization of Ferguson splines, Proc. IEEE Conf. Emerging Technologies and Factory Automation (ETFA ’06), pp. 833–839, 2006.

T. Berglund, A. Brodnik, H. Jonsson, M. Staffansson, and I. Soderkvist: Planning smooth and obstacle-avoiding B-spline paths for autonomous mining vehicles, IEEE Trans. Automat. Sci. Eng., Vol. 7, No. 1, pp. 167–172, 2010.

P. Gallina and A. Gasparetto: A technique to analytically formulate and to solve the 2-dimensional constrained trajectory planning problem for a mobile robot, J. Intelligent and Robotic Systems, Vol. 27, No. 3, pp. 237–262, 2000.

M. Yamamoto, M. Iwamura, and A. Mohri: Near-time-optimal trajectory planning for mobile robots with two independently driven wheels considering dynamical constraints and obstacles, J. Robotics Society of Japan, Vol. 16, No. 8, pp. 1123–1130, 1998 (in Japanese).

S. Sun, M. Egerstedt, and C.F. Martin: Control theoretic smoothing splines, IEEE Trans. Automat. Control, Vol. 45, No. 12, pp. 2271–2279, 2000.

C.F. Martin, S. Sun, and M. Egerstedt: Optimal control, statistics and path planning, Math. Comput. Modelling, Vol. 33, No. 1-3, pp. 237–253, 2001.

H. Fujioka and H. Kano: Optimal vector smoothing splines with coupled constraints, Trans. Institute of Systems, Control and Information Engineers, Vol. 25, No. 11, pp. 299–307, 2012.

H. Kano, H. Fujioka, and C.F. Martin: Optimal smoothing and interpolating splines with constraints, Applied Mathematics and Computation, Vol. 218, No. 5, pp. 1831–1844, 2011.

H. Kano, H. Fujioka, and C. Martin: Optimal smoothing spline with constraints on its derivatives, SICE Journal of Control, Measurement, and System Integration, Vol. 7, No. 2, pp. 104–111, 2014.

H. Fujioka, H. Kano, and C. Martin: Constrained smoothing and interpolating spline surfaces using normalized uniform B-splines, Communications in Information and Systems, Vol. 14, No. 1, pp. 23–56, 2014.

C. de Boor: A Practical Guide to Splines, Revised Edition, Springer-Verlag, 2001.

H. Kano, M. Egerstedt, H. Nakata, and C. Martin: B-splines and control theory, Appl. Math. Comput., Vol. 145, No. 2-3, pp. 263–288, 2003.

H. Kano and H. Fujioka: Spline trajectory planning for path with piecewise linear boundaries, Proc. 9th Eurosim Congress on Modelling and Simulation (EUROSIM2016), pp. 402–407, 2016.

H. Kano and H. Fujioka: Velocity and acceleration constrained trajectory planning by smoothing splines, Proc. 26th IEEE Int. Symp. Industrial Electronics, pp. 1167–1172, 2017.

P. Lancaster and M. Tismenetsky: The Theory of Matrices, 2nd Edition, Academic Press, 1985.

G. Wahba: Spline Models for Observational Data, CBMS-NSF Regional Conference Series in Applied Mathematics 59, Society for Industrial and Applied Mathematics (SIAM), 1990.

H. Kano, H. Nakata, and C.F. Martin: Optimal curve fitting and smoothing using normalized uniform B-splines: A tool for studying complex systems, Applied Mathematics and Computation, Vol. 169, No. 1, pp. 96–128, 2005.

H. Fujioka, H. Kano, M. Egerstedt, and C.F. Martin: Smoothing spline curves and surfaces for sampled data, Int. J. Innovative Computing, Information and Control, Vol. 1, No. 3, pp. 429–449, 2005.

S.R. Chang and U.Y. Huh: Curvature-continuous 3D path-planning using QPMI method, Int. J. Advanced Robotic Sys-