Continuous Curvature Path Generation Based on Bézier Curves for Autonomous Vehicles

Ji-wung Choi *, Renwick E. Curry †, Gabriel Hugh Elkaim ‡

Abstract—**In this paper we present two path planning algorithms based on Bézier curves for autonomous vehicles with waypoints and corridor constraints. Bézier curves have useful properties for the path generation problem. This paper describes how the algorithms apply these properties to generate the reference trajectory for vehicles to satisfy the path constraints. Both algorithms join a set of low-degree Bézier curves segments smoothly to generate the path. Additionally, we discuss the constrained optimization problem that optimizes the resulting path for a user-defined cost function. The simulation demonstrates the improvement of trajectory generation in terms of smoother steering control and smaller cross track error compared to previous work.**

Keywords: Bézier, Path Planning, Optimization, Autonomous Vehicle, Feedback Control

1 Introduction

Bézier Curves were invented in 1962 by the French engineer Pierre Bézier for designing automobile bodies. Today Bézier Curves are widely used in computer graphics and animation [6]. The Bézier curves have useful properties for the path generation problem as described in Section 2 of this paper. Hence many path planning techniques for autonomous vehicles have been discussed based on Bézier Curves in the literature. Cornell University Team for 2005 DARPA Grand Challenge [8] used a path planner based on Bézier curves of degree 3 in a sensing/action feedback loop to generate smooth paths that are consistent with vehicle dynamics. Skrjanc [7] proposed a new cooperative collision avoidance method for multiple robots with constraints and known start and goal velocities based on Bézier curves of degree 4. In this method, four control points out of five are placed such that desired positions and velocities of the start and the goal point are satisfied. The fifth point is obtained by minimizing penalty functions. Lizarraga [5] used Bézier curves for generating spatially deconflicted paths for multiple UAVs.

Choi has presented two path planning algorithms based on Bézier curves for autonomous vehicles with waypoints and corridor constraints [2]. Both algorithms join cubic Bézier curve segments smoothly to generate the reference trajectory for vehicles to satisfy the path constraints. Also, both algorithms are constrained in that the path must cross over a bisector line of corner area such that the tangent at the crossing point is normal to the bisector. Additionally, that paper discuss the constrained optimization problem that optimizes the resulting path for user-defined cost function. Since the Bézier curve is uniquely defined by its control points, the optimization problem is parameterized by the location of control points. Even though the simulation provided in that paper has shown the generation of smooth routes, discontinuities of the yaw angular rate have appeared at junction nodes between curve segments. This is because the curve segments are constrained to connect each other by only C^1 continuity, so the curvature of the path is discontinuous at the nodes. (Section 2 describes this more detail.)

To resolve this problem, new path planning algorithms were proposed [3]. The algorithms impose constraints such that curve segments are C^2 continuous in order to have curvature continuous for every point on the path. In addition, they give the reference path more freedom by eliminating redundant constraints used in [2], such as the tangent being normal to the bisector, the initial/final heading, and symmetry of curve segments on corner area. The degree of each Bézier curve segments are determined by the minimum number of control points to satisfy imposed constraints while cubic Bézier curves are used for every segments in [2]. The optimized resulting path is obtained by computing the constrained optimization problem for the same cost function as the one in [2]. The numerical simulation results provided in the paper demonstrate the improvement of trajectory generation in terms of smoother steering control and smaller cross track error.

The paper is organized as follows: Section 2 begins by describing the definition of the Bézier curve and its useful properties for path planning. Section 3 discusses the control problem for autonomous vehicles, the vehicle dynam-

*University of California, Santa Cruz Computer Engineering, Autonomous Systems Lab, 1156 High Street, Santa Cruz CA USA 95064 Tel/Fax: 831-428-2146/831-459-4829 Email: jwchoi@soe.ucsc.edu

†University of California, Santa Cruz Computer Engineering, Autonomous Systems Lab, 1156 High Street, Santa Cruz CA USA 95064 Email: rcurry@ucsc.edu

‡University of California, Santa Cruz Computer Engineering, Autonomous Systems Lab, 1156 High Street, Santa Cruz CA USA 95064 Email: elkaim@soe.ucsc.edu

ics, and vehicle control algorithms. Section 4 proposes two path planning methods based on Bézier curves, and discusses the constrained optimization problem of these methods. In Section 5, simulation results of control problem for autonomous vehicles are given. Finally, Section 6 provides conclusions.

2 Bézier Curve

A Bézier Curve of degree n can be represented as

$$\mathbf{P}(t) = \sum_{i=0}^{n} B_i^n(t)\mathbf{P}_i, \quad t \in [0,1]$$

where $\mathbf{P}_i$ are control points such that $\mathbf{P}(0) = \mathbf{P}_0$ and $\mathbf{P}(1) = \mathbf{P}_n$, $B_i^n(t)$ is a Bernstein polynomial given by

$$B_i^n(t) = \binom{n}{i}(1-t)^{n-i}t^i, \quad i \in \{0,1,\ldots,n\}$$

Bézier Curves have useful properties for path planning:

- They always passes through $\mathbf{P}_0$ and $\mathbf{P}_n$.
- They are always tangent to the lines connecting $\mathbf{P}_0 \to \mathbf{P}_1$ and $\mathbf{P}_n \to \mathbf{P}_{n-1}$ at $\mathbf{P}_0$ and $\mathbf{P}_n$ respectively.
- They always lie within the convex hull consisting of their control points.

2.1 The de Casteljau Algorithm

The de Casteljau Algorithm is named after the French mathematician Paul de Casteljau, who developed the algorithm in 1959. The de Casteljau algorithm describes a recursive process to subdivide a Bézier curve $\mathbf{P}(t)$ into two segments. The subdivided segments are also Bézier curves. Let $\{\mathbf{P}_0^0, \mathbf{P}_1^0, \ldots, \mathbf{P}_n^0\}$ denote the control points of $\mathbf{P}(t)$. The control points of the segments can be computed by

$$\begin{aligned}\mathbf{P}_i^j =& (1-\tau)\mathbf{P}_i^{j-1} + \tau\mathbf{P}_{i+1}^{j-1}, \\ & j \in \{1,\ldots,n\}, \quad i \in \{0,\ldots,n-j\}\end{aligned} \tag{1}$$

where $\tau \in (0,1)$. Then, $\{\mathbf{P}_0^0, \mathbf{P}_0^1, \ldots, \mathbf{P}_0^n\}$ are the control points of one segment and $\{\mathbf{P}_0^n, \mathbf{P}_1^{n-1}, \ldots, \mathbf{P}_n^0\}$ are the another. Figure 1 shows an example of subdividing a cubic Bézier curve by applying the de Casteljau algorithm with $\tau = .4$.

This algorithm leads to the following property [2]:

Remark 1. A Bézier curve $\mathbf{P}(t)$ constructed by control points $\{\mathbf{P}_0^0, \mathbf{P}_1^0, \ldots, \mathbf{P}_n^0\}$ always passes through the point $\mathbf{P}_0^n$ computed by applying the de Casteljau algorithm and using (1). Also, it is always tangent to $\overline{\mathbf{P}_0^{n-1}\mathbf{P}_1^{n-1}}$ at $\mathbf{P}_0^n$.

The path planning method introduced in the Section 4.2 is motivated by this property.

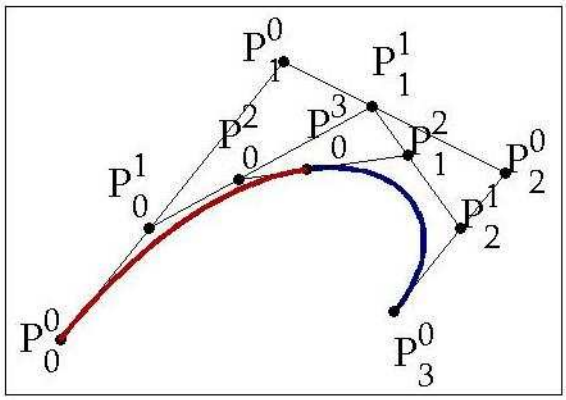


Figure 1: Subdividing a cubic Bézier curve with $\tau = .4$ by the de Casteljau Algorithm.

2.2 Derivatives, Continuity and Curvature

The derivatives of a Bézier curve, referred to as the hodograph, can be determined by its control points [6]. For a Bézier curve $\mathbf{P}(t) = \sum_{i=0}^{n} B_i^n(t)\mathbf{P}_i$, the first derivative can be represented as:

$$\dot{\mathbf{P}}(t) = \sum_{i=0}^{n-1} B_i^{n-1}(t)\mathbf{D}_i \tag{2}$$

where $\mathbf{D}_i$, control points of $\dot{\mathbf{P}}(t)$ are

$$\mathbf{D}_i = n(\mathbf{P}_{i+1} - \mathbf{P}_i)$$

Geometrically, (2) provides us with a tangent vector.

The first derivative of the Bézier curve $\mathbf{P}(t)$ at endpoints can be rewritten as the following equations by applying (2).

$$\begin{aligned}\dot{\mathbf{P}}(0) &= n(\mathbf{P}_1 - \mathbf{P}_0) \\ \dot{\mathbf{P}}(1) &= n(\mathbf{P}_n - \mathbf{P}_{n-1})\end{aligned} \tag{3}$$

The higher order derivative of a Bézier curve can be obtained by using the relationship of (2), iteratively. Thus the resulting second order derivatives of $\mathbf{P}(t)$ at endpoints are

$$\begin{aligned}\ddot{\mathbf{P}}(0) &= n(n-1)(\mathbf{P}_2 - 2\mathbf{P}_1 + \mathbf{P}_0) \\ \ddot{\mathbf{P}}(1) &= n(n-1)(\mathbf{P}_n - 2\mathbf{P}_{n-1} + \mathbf{P}_{n-2})\end{aligned} \tag{4}$$

Two Bézier curves $\mathbf{P}(t)$ and $\mathbf{Q}(t)$ are said to be C^k at t_0 continuous [6] if

$$\begin{aligned}\mathbf{P}(t_0) &= \mathbf{Q}(t_0), \\ \dot{\mathbf{P}}(t_0) &= \dot{\mathbf{Q}}(t_0), \\ &\vdots \\ \mathbf{P}^{(k)}(t_0) &= \mathbf{Q}^{(k)}(t_0)\end{aligned} \tag{5}$$

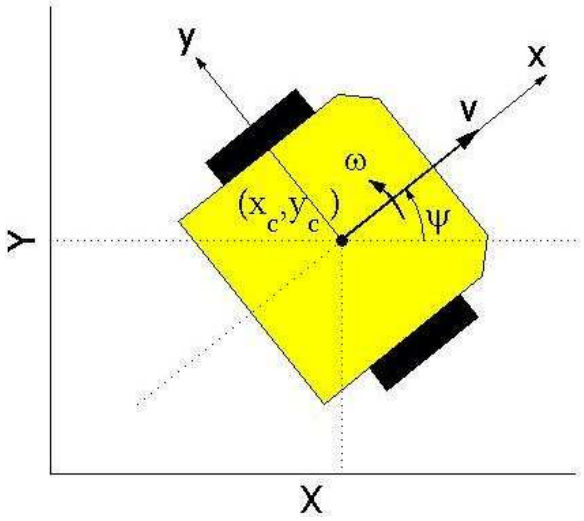


Figure 2: Schematic drawing of dynamic model of vehicle motion.

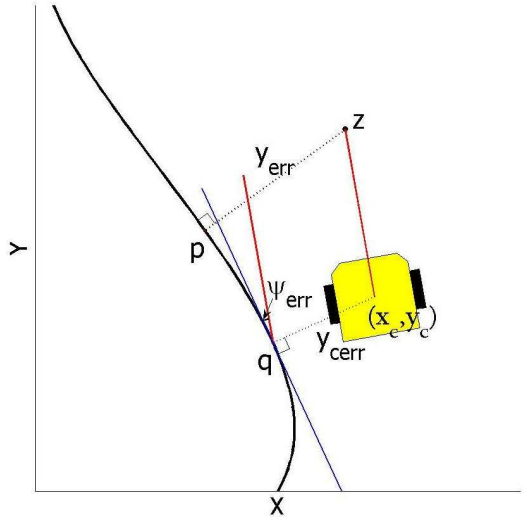


Figure 3: The position error is measured from a point **z**, projected in front of the vehicle, and unto the desired curve to point **p**.

The curvature of a Bézier curve $\mathbf{P}(t) = \big(x(t), y(t)\big)$ at t is given by [6]

$$\kappa(t) = \frac{|\dot{x}(t)\ddot{y}(t) - \dot{y}(t)\ddot{x}(t)|}{(\dot{x}^2(t) + \dot{y}^2(t))^{\frac{3}{2}}} \tag{6}$$

We can come up with the following property:

Lemma 1. For the path constructed by two Bézier curve segments $\mathbf{P}(t)|_{t\in[t_0,t_1]}$ and $\mathbf{Q}(t)|_{t\in[t_1,t_2]}$, if $\mathbf{P}(t)$ and $\mathbf{Q}(t)$ are at least C^2 continuous at t_1 then the path has continuous curvature for every point on it.

Proof. The curvature is expressed in terms of the first and the second derivative of a curve in (6). Since the Bézier curves are defined as polynomial functions of t, their k-th derivative for all $k = 1, 2, \ldots$ are continuous. Hence, they always have continuous curvature for all t. For two different Bézier curves $\mathbf{P}(t)$ and $\mathbf{Q}(t)$, it is sufficient that $\kappa(t_1)$, the curvature at the junction node is continuous if $\dot{\mathbf{P}}(t) = \dot{\mathbf{Q}}(t)$ and $\ddot{\mathbf{P}}(t) = \ddot{\mathbf{Q}}(t)$ are continuous at t_1. Therefore, if $\mathbf{P}(t)$ and $\mathbf{Q}(t)$ are at least C^2 continuous at t_1 then the path have the curvature continuous for every point on it. □

3 Problem Statement

Consider the control problem of a ground vehicle with a mission defined by waypoints and corridor constraints in a two-dimensional free-space. Our goal is to develop and implement an algorithm for navigation that satisfies these constraints. Let us denote each waypoint $\mathbf{W}_i \in \mathbb{R}^2$ for $i \in \{1, 2, \ldots, N\}$, where $N \in \mathbb{R}$ is the total number of waypoints. Corridor width is denoted as w_j for $j \in \{1, \ldots, N-1\}$, j-th widths of each segment between two waypoints, $\mathbf{W}_j$ and $\mathbf{W}_{j+1}$.

3.1 Dynamic Model of Vehicle Motion

This section describes a dynamic model for motion of a vehicle that is used in the simulation in Section 5. Figure 2 shows the schematic drawing of dynamic model of the vehicle that is used in the simulation.

For the dynamics of the vehicle, the state and the control vector are denoted $\mathbf{q}(t) = (x_c(t), y_c(t), \psi(t))^T$ and $\mathbf{u}(t) = (v(t), \omega(t))^T$ respectively, where (x_c, y_c) represents the position of the center of gravity of the vehicle. The vehicle yaw angle ψ is defined to the angle from the X axis. v is the longitudinal velocity of the vehicle at the center of gravity. $\omega = \dot{\psi}$ is the yaw angular rate. It follows that

$$\dot{\mathbf{q}}(t) = \begin{pmatrix} \cos\psi(t) & 0 \\ \sin\psi(t) & 0 \\ 0 & 1 \end{pmatrix} \mathbf{u}(t)$$

3.2 Controls

The vehicle uses the path following technique with feedback corrections as illustrated in Figure 3 [4]. A position and orientation error is computed every 50 ms. The cross track error y_{cerr} is defined by the shortest distance between the reference trajectory and the position of the center of gravity of the vehicle (x_c, y_c). A point **z** is computed with the current longitudinal velocity and heading of the vehicle from the current position. **z** is projected onto the reference trajectory at point **p** such that $\overline{\mathbf{zp}}$ is normal to the tangent at **p**. The cross track error y_{err} is defined by the distance between **z** and **p**.

The steering control ω uses PID controller with respect

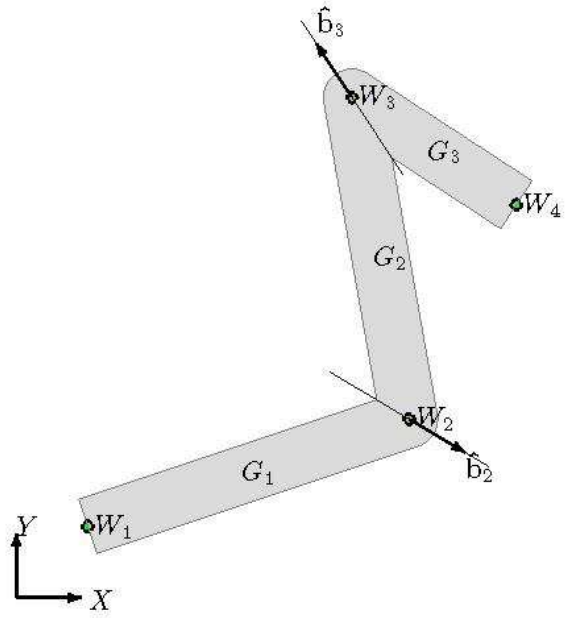


Figure 4: An example of the course with four waypoints. Gray area is the permitted area for vehicles under a corridor constraint.

to cross track error y_{err}.

$$\delta\omega = k_p y_{err} + k_d \frac{dy_{err}}{dt} + k_i \int y_{err} dt$$

4 Path Planning Algorithm

In this section, two path planning methods based on Bézier curves are proposed. To describe the algorithms, let us denote $\hat{\mathbf{b}}_j$ as the unit vector codirectional with the outward bisector of $\angle \mathbf{W}_{j-1}\mathbf{W}_j\mathbf{W}_{j+1}$ for $j \in \{2, \ldots, N-1\}$ as illustrated in Figure 4. The planned path must cross over the bisectors under the waypoint and the corridor constraints. The location of the crossing point is represented as $\mathbf{W}_j + d_j \cdot \hat{\mathbf{b}}_j$, where $d_j \in \mathbb{R}$ is an scalar value. The course is divided into segments G_i by bisectors. G_i indicates the permitted area for vehicles under corridor constraint w_i, from $\mathbf{W}_i$ to $\mathbf{W}_{i+1}$.

Bézier curves constructed by large numbers of control points are numerically unstable. For this reason, it is desirable to join low-degree Bézier curves together in a smooth way for path planning [7]. Thus both methods use a set of low-degree Bézier curves such that the neighboring curves are C^2 continuous at their end nodes. This will lead to continuous curvature on the resulting path by Lemma 1.

The Bézier curves used for the path plannings are denoted as ${}^i\mathbf{P}(t) = \sum_{k=0}^{n_i} B_k^{n_i}(t) \cdot {}^i\mathbf{P}_k$ for $i \in \{1, \ldots, M\}$, $t \in [0, 1]$ where M is the total number of the Bézier curves and n_i is the degree of ${}^iP$. The planned path denoted as P is a concatenation of all ${}^iP$.

4.1 Path Planning Placing Bézier Curves within Segments

In this path planning method, the Bézier curve ${}^i\mathbf{P}$ for $i \in \{1, \ldots, N-1\}$ are used within each segment G_i. The planned path $\mathbf{P}$ are designed such that it begins from $\mathbf{W}_1$ and ends to $\mathbf{W}_N$. Furthermore, the corridor and the C^2 continuity constraints are satisfied.

The control points of ${}^i\mathbf{P}$, ${}^i\mathbf{P}_k$ for $k = \{0, \ldots, n_i\}$ are determined to maintain these conditions.

- The beginning and the end point are $\mathbf{W}_1$ and $\mathbf{W}_N$.

$$ {}^1\mathbf{P}_0 = \mathbf{W}_1, \quad {}^{N-1}\mathbf{P}_{n_{N-1}} = \mathbf{W}_N \tag{7}$$

- The adjacent curves, ${}^{j-1}\mathbf{P}$ and ${}^j\mathbf{P}$ are C^2 continuous at the crossing point, $\mathbf{W}_j + d_j \cdot \hat{\mathbf{b}}_j$ for $j \in \{2, \ldots, N-1\}$.

$$\begin{aligned} &{}^{j-1}\mathbf{P}_{n_{j-1}} = {}^j\mathbf{P}_0 = \mathbf{W}_j + d_j \cdot \hat{\mathbf{b}}_j \\ &n_{j-1}({}^{j-1}\mathbf{P}_{n_{j-1}} - {}^{j-1}\mathbf{P}_{n_{j-1}-1}) = n_j({}^j\mathbf{P}_1 - {}^j\mathbf{P}_0) \\ &n_{j-1}(n_{j-1}-1)({}^{j-1}\mathbf{P}_{n_{j-1}} - 2 \cdot {}^{j-1}\mathbf{P}_{n_{j-1}-1} + \\ &{}^{j-1}\mathbf{P}_{n_{j-1}-2}) = n_j(n_j-1)({}^j\mathbf{P}_2 - 2 \cdot {}^j\mathbf{P}_1 + {}^j\mathbf{P}_0) \end{aligned} \tag{8}$$

- The crossing points are bounded within the corridor.

$$|d_j| < \frac{1}{2}\min(w_{j-1}, w_j) \tag{9}$$

- ${}^i\mathbf{P}_k$ always lie within the area of G_i.

$$ {}^i\mathbf{P}_1 \in G_i, \ldots, {}^i\mathbf{P}_{n_i-1} \in G_i \tag{10}$$

Equation (8) is obtained by applying (3), (4) and (5). Equation (10) makes the resulting Bézier curve satisfy the corridor constraint by the convex hull property.

At each crossing point, three control points of each adjacent Bézier curve are dedicated to the C^2 continuity constraint by (2), (6), and Lemma 1. So the minimum number of control points to satisfy the constraints independent on the others are four for ${}^1\mathbf{P}$, ${}^{N-1}\mathbf{P}$ and six for the others. n_i is determined by this:

$$\begin{cases} n_i = 3, & i \in \{1, N-1\} \\ n_i = 5, & i \in \{2, \ldots, N-2\} \end{cases} \tag{11}$$

Note that ${}^1\mathbf{P}_0$ and ${}^{N-1}\mathbf{P}_{n_{N-1}}$ are fixed in (7). ${}^{j-1}\mathbf{P}_{n_{j-1}}$ and ${}^j\mathbf{P}_0$ rely on d_j in (8). Also, ${}^{j-1}\mathbf{P}_{n_{j-1}-1}$ and ${}^{j-1}\mathbf{P}_{n_{j-1}-2}$ rely on ${}^j\mathbf{P}_1$ and ${}^{j-1}\mathbf{P}_2$.

So the free variables are, $\forall j \in \{2, \ldots, N-1\}$, $\mathbf{P}_1 = \{{}^j\mathbf{P}_1\}$, $\mathbf{P}_2 = \{{}^j\mathbf{P}_2\}$ and $\mathbf{d} = \{d_j\}$. The number of the variables or the degree of freedom is $5(N-2)$. The variables are computed by minimizing the constrained optimization problem:

$$\min_{\mathbf{P}_1,\mathbf{P}_2,\mathbf{d}} J = \sum_{i=1}^{N-1} J_i \tag{12}$$

subject to (9) and (10), where J_i is the cost function of ${}^i\mathbf{P}(t)$ which is defined in Section 5. As the result, the planned trajectory goes from $\mathbf{W}_1$ to $\mathbf{W}_N$ through inside of corridor with C^2 continuity at the crossing point on the bisectors. That is, the trajectory has curvature continuous at every point on it by Lemma 1.

4.2 Path Planning Placing Bézier Curves on Corners

In the Section 4.1, a Bézier curve is used within each segment. Another path planning method adds quadratic Bézier curves on the corner area around $\mathbf{W}_j$, $j \in \{2, \ldots, N-1\}$. The quadratic Bézier curves are denoted as ${}^j\mathbf{Q}(t) = \sum_{k=0}^{2} B_k^2(t) \cdot {}^j\mathbf{Q}_k^0$ intersects the j-th bisector. The first and the last control point, ${}^j\mathbf{Q}_0^0$ and ${}^j\mathbf{Q}_2^0$ are constrained to lie within G_{j-1} and G_j, respectively.

Let t_c denote the Bézier curve parameter corresponding to the crossing point of ${}^j\mathbf{Q}(t)$ on the bisector, such that

$${}^j\mathbf{Q}(t_c) = \mathbf{W}_j + d_j \cdot \hat{\mathbf{b}}_j. \tag{13}$$

Let θ_j denote the angle of the tangent vector at the crossing point from X-axis, such that

$${}^j\dot{\mathbf{Q}}(t_c) = \left(\, |{}^j\dot{\mathbf{Q}}(t_c)| \cos\theta_j, \; |{}^j\dot{\mathbf{Q}}(t_c)| \sin\theta_j \, \right). \tag{14}$$

The notations are illustrated in Figure 5. Due to the constraint of ${}^j\mathbf{Q}_0^0$ and ${}^j\mathbf{Q}_2^0$ within G_{j-1} and G_j, the feasible scope of θ_j is limited to the same directions as $\mathbf{W}_{j+1}$ is with respect to $\hat{\mathbf{b}}_j$. In other words, if $\mathbf{W}_{j+1}$ is to the right of $\hat{\mathbf{b}}_j$, then θ_j must point to the right of $\hat{\mathbf{b}}_j$, and vice versa.

Given ${}^j\mathbf{Q}_0^0$, ${}^j\mathbf{Q}_2^0$, d_j, and θ_j, the other control point ${}^j\mathbf{Q}_1^0$ is computed such that the crossing point is located at $\mathbf{W}_j + d_j \cdot \hat{\mathbf{b}}_j$ and the angle of the tangent vector at the crossing point is θ_j.

Since each control point is two-dimensional, the degrees of freedom of ${}^j\mathbf{Q}(t)$ is six. Since d_j and θ_j are scaler, representing ${}^j\mathbf{Q}(t)$ in terms of ${}^j\mathbf{Q}_0^0$, ${}^j\mathbf{Q}_2^0$, d_j, and θ_j does not affect the degrees of freedom. However, it comes up with an advantage for corridor constraint. If we compute ${}^j\mathbf{Q}_1^0$ such as above, then the points computed by applying the de Casteljau algorithm such that two subdivided curves are separated by the j-th bisector are represented as ${}^j\mathbf{Q}_0^0$ and ${}^j\mathbf{Q}_2^0$ as described in the following. Recall that the two curves are constructed by $\{{}^j\mathbf{Q}_0^0, {}^j\mathbf{Q}_0^1, {}^j\mathbf{Q}_0^2\}$ and $\{{}^j\mathbf{Q}_0^2, {}^j\mathbf{Q}_1^1, {}^j\mathbf{Q}_2^0\}$. We can test if the convex hull of $\{{}^j\mathbf{Q}_0^0, {}^j\mathbf{Q}_0^1, {}^j\mathbf{Q}_0^2\}$ lies within G_{j-1} and if that of $\{{}^j\mathbf{Q}_0^2, {}^j\mathbf{Q}_1^1, {}^j\mathbf{Q}_2^0\}$ lies within G_j in (27), instead of testing that of $\{{}^j\mathbf{Q}_0^0, {}^j\mathbf{Q}_1^0, {}^j\mathbf{Q}_2^0\}$. (Note that ${}^j\mathbf{Q}_1^0$ is not constrained to lie within corridor as shown in Figure 5.) As the result, the convex hull property is tested for more tight condition against the corridor constraint without increasing the degrees of freedom.

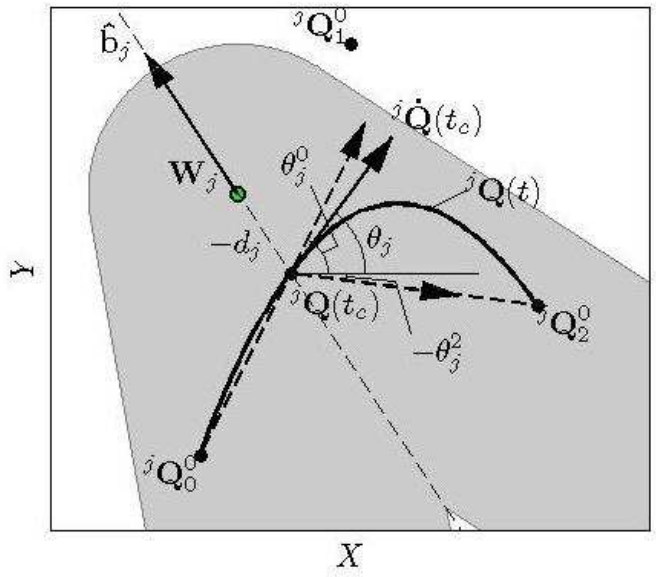


Figure 5: The geometry of the ${}^j\mathbf{Q}(t)$ on corner area around $\mathbf{W}_j$. t_c is corresponding to the crossing point of ${}^j\mathbf{Q}(t)$ on the bisector. θ_j is the angle of the tangent vector at the crossing point.

It is important to note that ${}^j\mathbf{Q}_0^0$ and ${}^j\mathbf{Q}_2^0$ can not be located on different sides with respect to the line passing through the crossing point and co-linear with the tangent vector. Recall that the sign of curvature of ${}^j\mathbf{Q}(t)$ is never changed for all t, since it is a quadratic Bézier curve. If ${}^j\mathbf{Q}_0^0$ and ${}^j\mathbf{Q}_2^0$ are at different sides with respect to the line, then the sign of curvature of $\mathbf{Q}(t)$ must change around t_c, which is infeasible. Let θ_j^0 denote the angle of the vector ${}^j\mathbf{Q}(t_c) - {}^j\mathbf{Q}_0^0$ as shown in Figure 5. If $\mod(\theta_j - \theta_j^0, 2\pi)$ is less than π or ${}^j\dot{\mathbf{Q}}(t_c)$ points to the left side of the ${}^j\mathbf{Q}(t_c) - {}^j\mathbf{Q}_0^0$, then $\mathbf{Q}_0^0$ is at left side with respect to the line. Otherwise, it is at the right. The calculation depends on which side $\mathbf{W}_{j+1}$ is at with respect to $\mathbf{W}_j - \mathbf{W}_{j-1}$. In the figure, $\mathbf{W}_{j+1}$ is at right side with respect to the vector. In the other case, the computation is opposite. The same computation can be applied to θ_j^2, the angle of ${}^j\mathbf{Q}_2^0 - {}^j\mathbf{Q}(t_c)$. The range of feasible θ is chosen as the range in which ${}^j\mathbf{Q}_0^0$ and ${}^j\mathbf{Q}_2^0$ locate on the same side with respect to the line. In the example shown in Figure 5, it ranges between θ_j^0 and $-\theta_j^2$. This property parallels with Lemma 2.

In order to compute ${}^j\mathbf{Q}_1^0$, the world coordinate frame T is transformed and rotated into the local frame ${}^jT$ where the origin is at the crossing point, ${}^j\mathbf{Q}(t_c)$ and X axis is codirectional with the tangent vector of the curve at the crossing point, ${}^j\dot{\mathbf{Q}}(t_c)$.

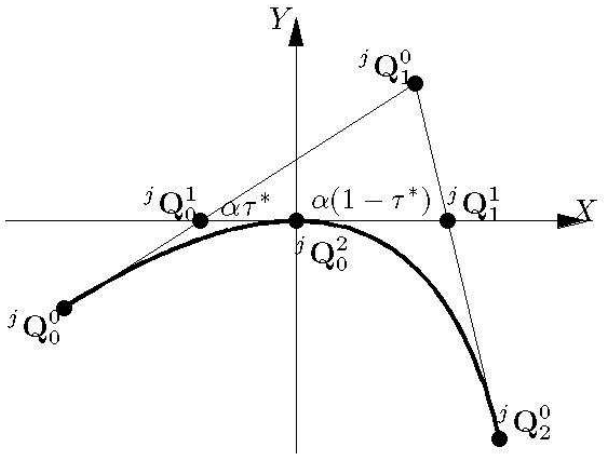


Figure 6: The geometry of the control points of ${}^j\mathbf{Q}(t)$ and the points computed by applying the de Casteljau algorithm with τ^* with respect to ${}^jT$.

Let us consider the subdivision ratio, $\tau^* \in (0,1)$ such that the location of ${}^j\mathbf{Q}_0^2$ computed by applying the de Casteljau algorithm with it is the crossing point. In other words, τ^* has ${}^j\mathbf{Q}_0^2$ be at the origin with respect to ${}^jT$ frame. Figure 6 illustrates the control points of ${}^j\mathbf{Q}(t)$ and the points computed by applying the de Casteljau algorithm with τ^* with respect to ${}^jT$ frame. Note that ${}^j\mathbf{Q}_0^2$ is at the origin by the definition of ${}^jT$ and τ^*. ${}^j\mathbf{Q}_0^1$ and ${}^j\mathbf{Q}_1^1$ are on the X axis by the definition of ${}^jT$ and Remark 1.

It is important to note the following lemma that describes the constraint of ${}^j\mathbf{Q}_0^0$ and ${}^j\mathbf{Q}_2^0$ for given d_j and θ_j. To describe this, let the coordinate of the control points be denoted as $\mathbf{Q}_i^0 = \big(x_i, y_i\big)$, $i \in \{0,1,2\}$, where all coordinates are with respect to ${}^jT$.

Lemma 2. Suppose $d_j \in \mathbb{R}$ and $\theta_j \in \mathbb{R}$ are given. For the quadratic Bézier curve ${}^j\mathbf{Q}(t)$ to intersect j-th bisector with the crossing point determined by the d_j and (13), and the tangent vector at the point determined by the θ_j and (14), it is necessary that $y_0 y_2 \geq 0$.

Proof. Let $\big(x(t), y(t)\big)$ denote the coordinate of ${}^j\mathbf{Q}(t)$ with respect to ${}^jT$. By the definition of ${}^jT$ and Remark 1, $\mathbf{Q}(t)$ passes through the origin with tangent slope of zero with respect to ${}^jT$. That is, $x(t_c) = 0$, $y(t_c) = 0$ and $\dot{y}(t_c) = 0$. Suppose that $y_0 = y(0) < 0$. Since $y(t)$ is a quadratic polynomial, $\dot{y}(t) > 0$ and $\ddot{y}(t) < 0$ for $t \in [0, t_c)$. Subsequently, $\dot{y}(t) < 0$ and $\ddot{y}(t) < 0$ for $t \in (t_c, 1]$. Thus, $y_2 = y(1) < 0$ and $y_0 y_2 > 0$. Similarly, if $y_0 > 0$ then $y_1 > 0$.

If $y_0 = 0$ then $\dot{y}(t) = 0$ for $t \in [0,1]$ and $y_2 = 0$. Therefore, $y_0 y_2 = 0$. □

The geometrical meaning of Lemma 2 is that for given a crossing point and the tangent vector on it, ${}^j\mathbf{Q}_0^0$ and ${}^j\mathbf{Q}_2^0$ can not be located on different sides with respect to the line passing through the crossing point and co-linear with the tangent vector.

We are to calculate ${}^j\mathbf{Q}_1^0$ depending on whether $y_0 y_2$ is nonzero. For simplicity, superscript j is dropped from now on.

Without loss of generality, suppose that $y_0 < 0$ and $y_2 < 0$. Recall that $\mathbf{Q}_0^2$ is at the origin and that $\mathbf{Q}_0^1$ and $\mathbf{Q}_1^1$ are on the X axis with respect to ${}^jT$. $\mathbf{Q}_0^2$ is represented as

$$\mathbf{Q}_0^2 = (1-\tau^*)\mathbf{Q}_0^1 + \tau^*\mathbf{Q}_1^1$$

by applying (1). So the coordinates of $\mathbf{Q}_0^1$ and $\mathbf{Q}_1^1$ can be represented as

$$\mathbf{Q}_0^1 = \big(-\alpha\tau^*, 0\big),\ \mathbf{Q}_1^1 = \big(\alpha(1-\tau^*), 0\big), \quad \alpha > 0 \tag{15}$$

where $\alpha > 0$ is some constant. Applying (1) with $i = 0$ and $j = 1$ and arranging the result with respect to $\mathbf{Q}_1^0$ by using (15) gives

$$\mathbf{Q}_1^0 = \big(-\alpha - \frac{1-\tau^*}{\tau^*}x_0, -\frac{1-\tau^*}{\tau^*}y_0\big) \tag{16}$$

Similarly, applying (1) with $i = 1$ and $j = 1$ and arranging the result with respect to $\mathbf{Q}_1^0$ yields

$$\mathbf{Q}_1^0 = \big(\alpha - \frac{\tau^*}{1-\tau^*}x_2, -\frac{\tau^*}{1-\tau^*}y_2\big) \tag{17}$$

where α and τ^* are obtained by equating (16) and (17):

$$\tau^* = \frac{1}{1+\sqrt{y_2/y_0}},\ \alpha = \frac{x_0 y_2 - y_0 x_2}{2y_0\sqrt{y_2/y_0}} \tag{18}$$

Notice that τ^* and α have the square root of y_2/y_0. So, if $y_0 y_2 < 0$ then τ^* and α are not determined, hence, $\mathbf{Q}(t)$ is infeasible. That is, (18) ends up with Lemma 2.

If $y_0 = y_2 = 0$ then all control points of ${}^j\mathbf{Q}$ are on X axis (See proof of Lemma 2). In the geometric relation of control points and the points computed by applying the de Casteljau algorithm as shown in Figure 7, we obtain

$$\begin{aligned} x_0 &= -(\alpha+\beta)\tau \\ x_2 &= (\alpha+\gamma)(1-\tau) \\ \alpha &= \beta(1-\tau) + \gamma\tau \end{aligned} \tag{19}$$

where $\alpha > 0$, $\beta > 0$, $\gamma > 0$ are some constants. Using (19), $\mathbf{Q}_1^0 = (x_1, 0)$ is represented in terms of arbitrary $\tau \in (0,1)$:

$$x_1 = -\frac{1}{2}\big(\frac{1-\tau}{\tau}x_0 + \frac{\tau}{1-\tau}x_2\big) \tag{20}$$

Then Bézier curves ${}^i\mathbf{P}(t)$ for $i \in \{1, 2, \ldots, N-1\}$ are used within each segment G_i so that ${}^{j-1}\mathbf{P}$ and ${}^j\mathbf{Q}$ are C^2

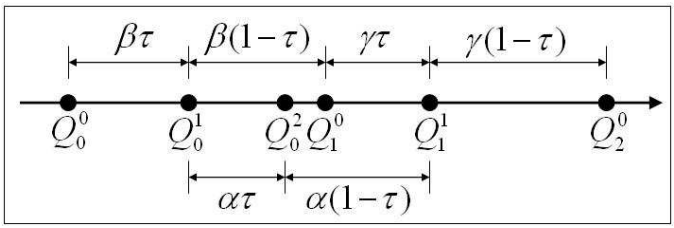


Figure 7: The geometry of ${}^j\mathbf{Q}(t)$ with respect to ${}^jT$ when $y_0 = y_2 = 0$.

continuous at ${}^j\mathbf{Q}_0$, ${}^j\mathbf{P}$ and ${}^j\mathbf{Q}$ are C^2 continuous at ${}^j\mathbf{Q}_2$. The degree of ${}^i\mathbf{P}(t)$, n_i is determined by the minimum number of control points to satisfy the constraint:

$$\begin{cases} n_i = 3, & i \in \{1, N-1\} \\ n_i = 5, & i \in \{2, \ldots, N-2\} \end{cases}$$

The constraints imposed on the planned path are as follows:

- The beginning and end point of $\mathbf{P}$ are $\mathbf{W}_1$ and $\mathbf{W}_N$.

$$ {}^1\mathbf{P}_0 = \mathbf{W}_1, \quad {}^{N-1}\mathbf{P}_{n_{N-1}} = \mathbf{W}_N \tag{21}$$

- ${}^{j-1}\mathbf{P}(t)$ and ${}^j\mathbf{Q}(t)$ are C^2 continuous at ${}^j\mathbf{Q}_0$.

$$\begin{aligned} &{}^{j-1}\mathbf{P}^0_{n_{j-1}} = {}^j\mathbf{Q}^0_0 \\ &n_{j-1}({}^{j-1}\mathbf{P}^0_{n_{j-1}} - {}^{j-1}\mathbf{P}^0_{n_{j-1}-1}) = 2({}^j\mathbf{Q}^0_1 - {}^j\mathbf{Q}^0_0) \\ &n_{j-1}(n_{j-1}-1)({}^{j-1}\mathbf{P}^0_{n_{j-1}} - 2\cdot{}^{j-1}\mathbf{P}^0_{n_{j-1}-1} + \\ &{}^{j-1}\mathbf{P}^0_{n_{j-1}-2}) = 2\cdot 1\cdot({}^j\mathbf{Q}^0_2 - 2{}^j\mathbf{Q}^0_1 + {}^j\mathbf{Q}^0_0) \end{aligned} \tag{22}$$

- ${}^j\mathbf{P}(t)$ and ${}^j\mathbf{Q}(t)$ are C^2 continuous at ${}^j\mathbf{Q}_2$.

$$\begin{aligned} &{}^j\mathbf{P}^0_0 = {}^j\mathbf{Q}^0_2 \\ &n_j({}^j\mathbf{P}^0_1 - {}^j\mathbf{P}^0_0) = 2({}^j\mathbf{Q}^0_2 - {}^j\mathbf{Q}^0_1) \\ &n_j(n_j-1)({}^j\mathbf{P}^0_2 - 2\cdot{}^j\mathbf{P}^0_1 + {}^j\mathbf{P}^0_0) \\ &= 2\cdot 1\cdot({}^j\mathbf{Q}^0_2 - 2{}^j\mathbf{Q}^0_1 + {}^j\mathbf{Q}^0_0) \end{aligned} \tag{23}$$

- The crossing points are bounded within the corridor.

$$|d_j| < \frac{1}{2}\min(w_{j-1}, w_j) \tag{24}$$

- The tangent vectors at the crossing points have the same direction as $\mathbf{W}_{j+1}$ is with respect to $\hat{\mathbf{b}}_j$.

$$\begin{aligned} &\text{mod}\,(\angle(\mathbf{W}_{j+1} - \mathbf{W}_j) - \angle\hat{\mathbf{b}}_j, 2\pi) > \pi \\ \Rightarrow\quad &\text{mod}\,(\theta_j - \angle\hat{\mathbf{b}}_j, 2\pi) > \pi, \\ &\text{mod}\,(\angle(\mathbf{W}_{j+1} - \mathbf{W}_j) - \angle\hat{\mathbf{b}}_j, 2\pi) < \pi \\ \Rightarrow\quad &\text{mod}\,(\theta_j - \angle\hat{\mathbf{b}}_j, 2\pi) < \pi \end{aligned} \tag{25}$$

- ${}^j\mathbf{Q}^0_0$ and ${}^j\mathbf{Q}^0_2$ with respect to ${}^jT$ satisfies Lemma 2.

$$y_0 y_2 \geq 0 \tag{26}$$

where y_0 and y_2 are with respect to ${}^jT$.

- ${}^j\mathbf{Q}^0_0$ and ${}^j\mathbf{Q}^1_0$ lie within G_{j-1}. ${}^j\mathbf{Q}^0_2$ and ${}^j\mathbf{Q}^1_1$ lie within G_j.

$$ {}^j\mathbf{Q}^0_0 \in G_{j-1},\ {}^j\mathbf{Q}^1_0 \in G_{j-1},\ {}^j\mathbf{Q}^0_2 \in G_j,\ {}^j\mathbf{Q}^1_1 \in G_j \tag{27}$$

- $\{{}^i\mathbf{P}_1, \ldots, {}^i\mathbf{P}_{n_i-1}\}$ always lie within the area of G_i.

$$ {}^i\mathbf{P}_1 \in G_i,\ \ldots,\ {}^i\mathbf{P}_{n_i-1} \in G_i \tag{28}$$

Then $6(N-2)$ free variables $\mathbf{Q} = \{{}^j\mathbf{Q}_0\}$, $\mathbf{d} = \{d_j\}$ and $\theta = \{\theta_j\}$ for $j \in \{2, \ldots, N-1\}$ are computed by minimizing the constrained optimization problem:

$$\min_{\mathbf{Q},\mathbf{d},\theta} J = \sum_{i=1}^{N-1} J_i \tag{29}$$

subject to (24), (25), (26), (27), and (28).

Notice that the convex hull property is tested for ${}^j\mathbf{Q}^1_0$ and ${}^j\mathbf{Q}^1_1$ of the divided curves instead of ${}^j\mathbf{Q}^0_1$ in (27). As the result, it comes up with more tight condition for curves against the corridor constraint.

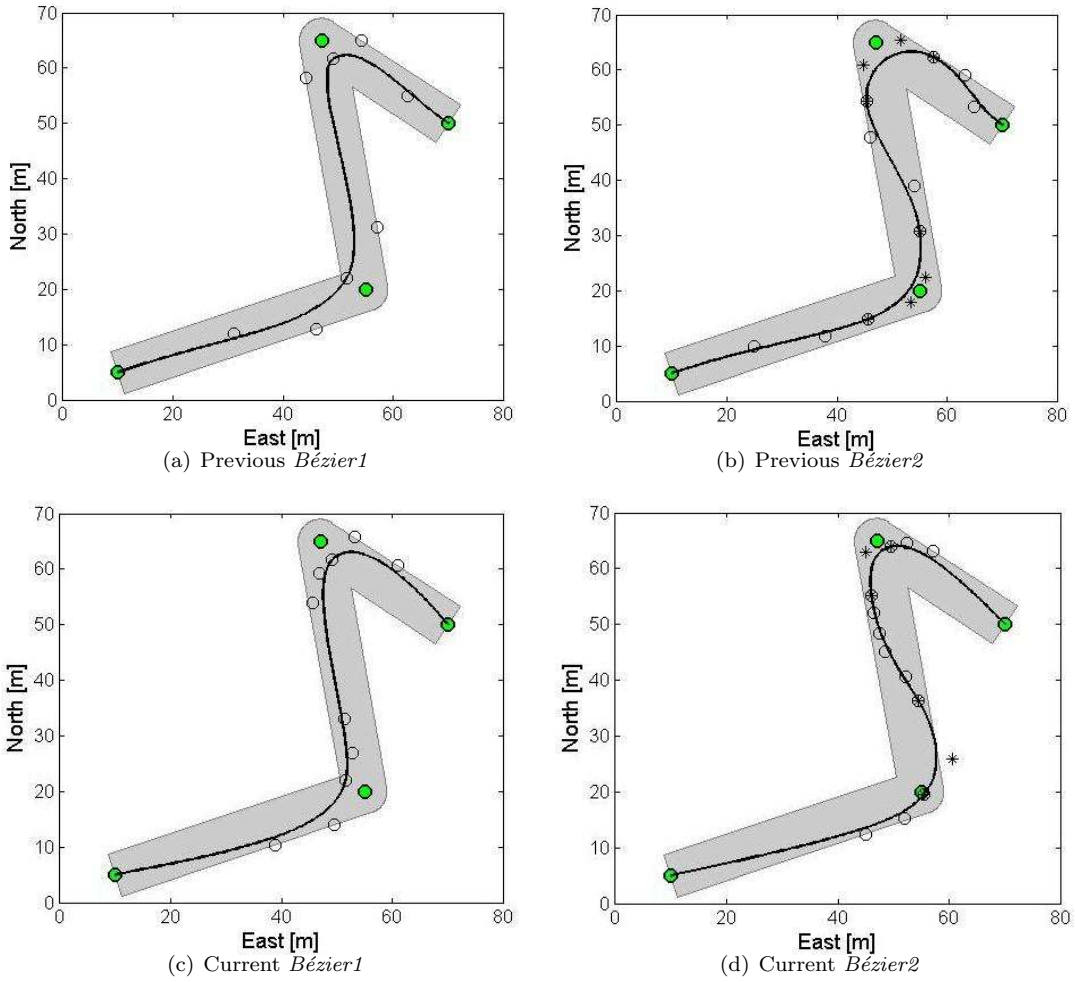
5 Simulation Results

Simulations performed in this paper use the course with waypoints $\mathbf{W} = \{\mathbf{W}_k\}$, $k \in \{1, \ldots, N\}$ and corridor width $w_i = 8$, $i \in \{1, \ldots, N-1\}$ for $N = 4$ as illustrated in Figure 4. The location of waypoints are given by two-dimensional world coordinates (X, Y) in *meter* scale: $\mathbf{W}_1 = (10, 5)$, $\mathbf{W}_2 = (55, 20)$, $\mathbf{W}_3 = (47, 65)$, $\mathbf{W}_4 = (70, 50)$. The initial position is assumed to fit to the first waypoint of the reference path. The constant longitudinal velocity $v(t) = 10\ m/s$ is used. The magnitude of ω is bounded within $|\omega|_{max} = 25\ rpm$. The PID gains are given by: $k_p = 2$, $k_d = 1$, and $k_i = 0.1$.

Path planning methods based on Section 4.1, and 4.2 are denoted as *Bézier1* and *Bézier2* respectively. Figure 8(a) and 8(c) show the path planned by *Bézier1* of [2] and of this paper, respectively. Figure 8(b) and 8(d) are ones by *Bézier2* of the two methods. Circles indicate the location of control points of Bézier curve segments, ${}^i\mathbf{P}$. In Figure 8(b) and 8(d), control points of ${}^j\mathbf{Q}$ are marked by stars. All of them are obtained by solving Equation (12) or (29) with

$$J_i = \int_0^1 \left[(a_i \left|{}^i\kappa(t)\right|^2 + b_i \left|{}^i\dot{\kappa}(t)\right|^2 \right] dt$$

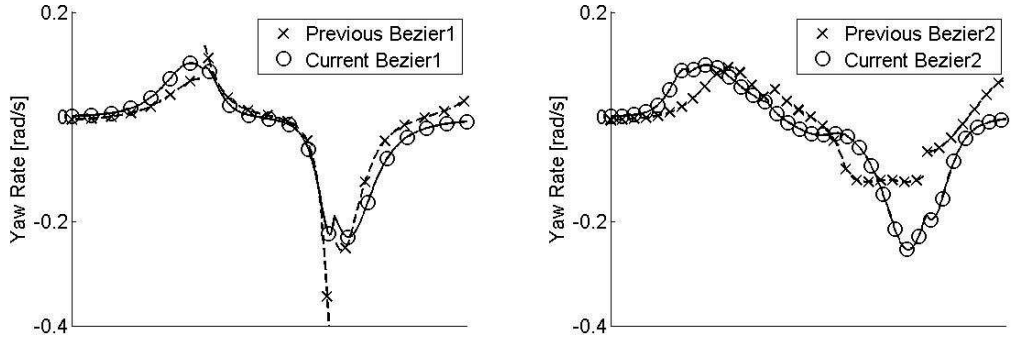
Where $a_i = b_i = 1$. The cost function leads to resulting paths with larger radii of curvature for Bézier curves. Compared to the paths generated by [2], the proposed algorithm generated smoother paths in the turning area.



(a) Previous *Bézier1* (b) Previous *Bézier2*

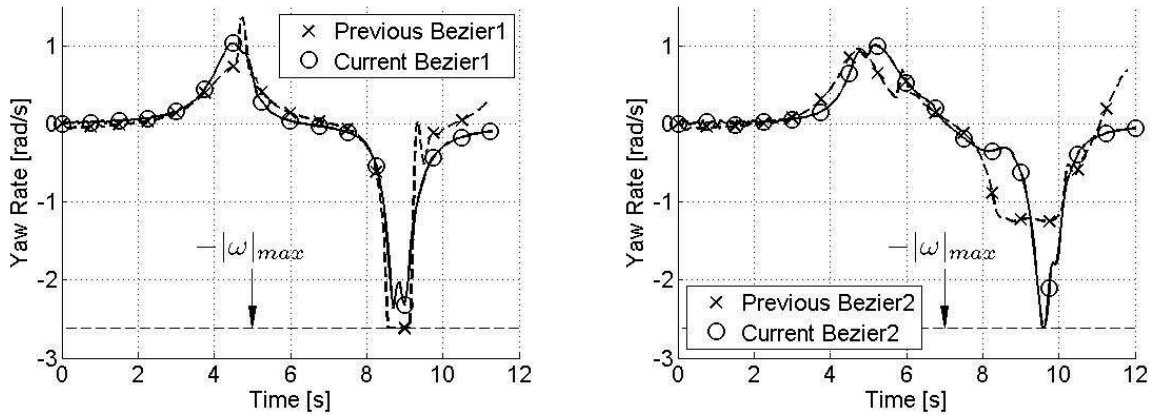
(c) Current *Bézier1* (d) Current *Bézier2*

Figure 8: The planned paths by previous methods of [2] (top) and by current method (bottom). The paths at left column are planned by applying *Bézier1*. Those at right column are by *Bézier2*. Control points of ${}^{i}P$ are indicated as 'o' and those of ${}^{j}Q$ are '$*$'.



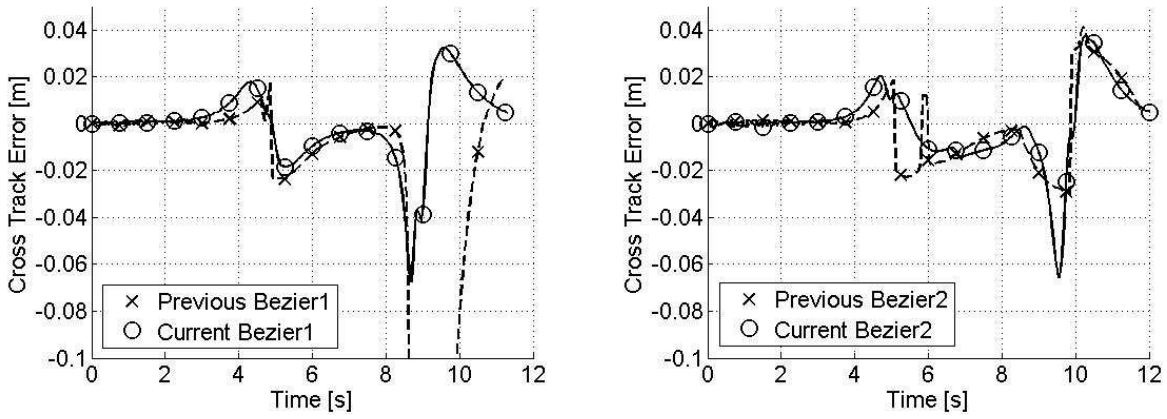
(a) κ of *Bézier1*. (b) κ of *Bézier2*.

Figure 9: The curvature κ by method of [2] (x) and the proposed method (o). The results of *Bézier1* is at left, that of *Bézier2* is at right.



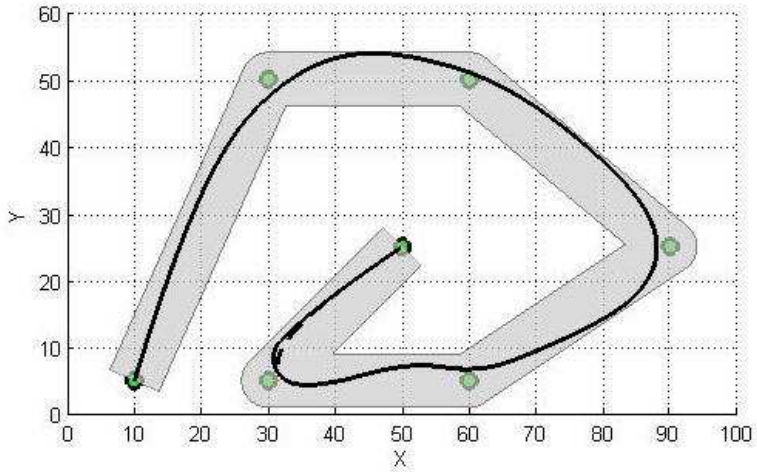
(a) ω of *Bézier1*. (b) ω of *Bézier2*.

Figure 10: The steering control ω by previous method of [2] (x) and by current method (o). The results of *Bézier1* is at left, that of *Bézier2* is at right.

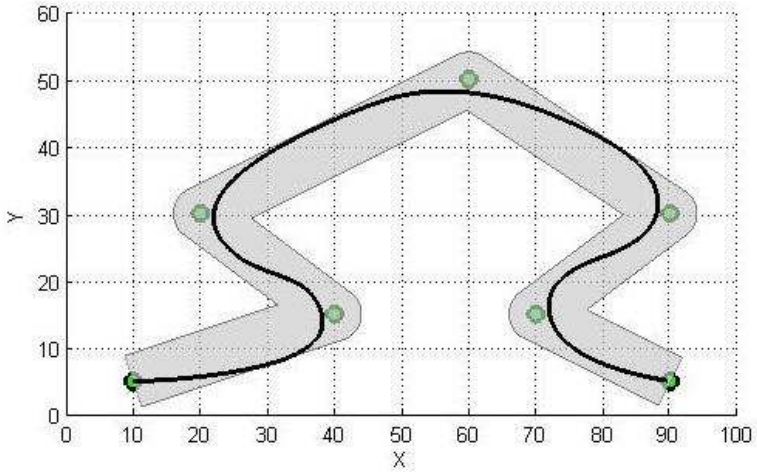


(a) y_{cerr} of *Bézier1*. (b) y_{cerr} of *Bézier2*.

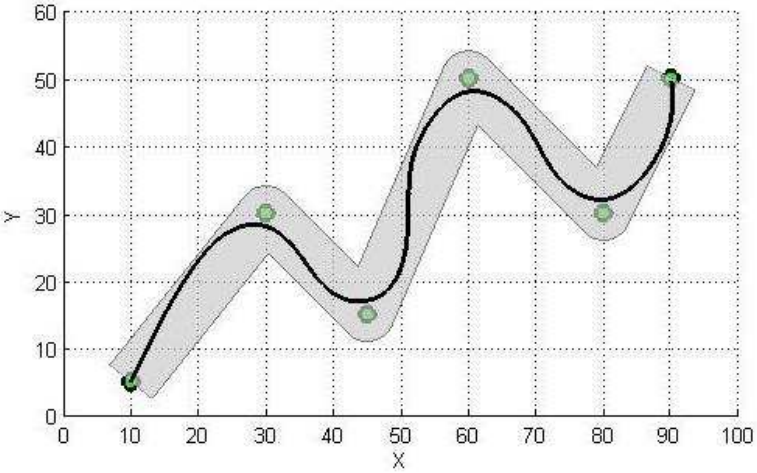
Figure 11: The cross track error y_{cerr} by previous method of [2] (x) and by current method (o). The results of *Bézier1* is at left, that of *Bézier2* is at right.



(a) Arbitrary course 1



(b) Arbitrary course 2



(c) Arbitrary course 3

Figure 12: Tracking simulation results over arbitrary courses using *Bézier2* method. The planned paths are indicated as dotted lines and the actual trajectories are solid lines.

Figure 13: The Overbot, the autonomous ground vehicle at Autonomous Systems Lab at UCSC.

Main difference of the proposed algorithms from the previous ones of [2] is the degree of continuity at the junction nodes: C^2. We can verify that the paths planned by the proposed algorithms have continuous curvature for every point on it (See Lemma 1) in Figure 9. On other hand, the paths planned by the previous algorithms, in which curve segments are constrained to connect each other by only C^1 continuity are discontinuous at junction nodes.

Recall that

$$\omega = \kappa v.$$

Assuming that v is continuous, if κ is continuous then ω is continuous. Especially when v is constant as this simulation, since ω is proportional to κ, continuity character of ω tends to follow that of κ. We can verify this in the graphs of the steering control ω by two methods in Figure 10. The path planned by the proposed algorithms in Figure 10(b) has smoother steering at junction nodes, compared to the ones obtained by [2] in Figure 10(a). The discontinuity of ω by that method imposes large forces and large changes in forces on the vehicle in the lateral direction. Moreover, the proposed algorithms result in smaller cross track error in Figure 11(a) over the one by [2].

More tracking results over arbitrary and longer courses are shown in Figure 12. The results are obtained by applying *Bézier2* method with the same parameters used above. These simulation results demonstrate generation of successful routes for vehicles using the algorithm as well as control results.

6 Conclusions and Future Work

This paper presents two path planning algorithms based on Bézier curves for autonomous vehicles with waypoints and corridor constraints. Bézier curves provide an efficient way to generate the optimized path and satisfy the constraints at the same time. The simulation results also show that the trajectory of the vehicle follows the planned path within the constraints.

These path planning algorithms will be implemented on the Overbot as shown in Figure 13, the autonomous ground vehicle at Autonomous Systems Lab at UCSC.

Enabling autonomous vehicles to detect unknown obstacles and safely avoid them is essential to future operations. Future work will employ receding horizon control methods to generate real-time Bézier-based optimal trajectories while avoiding obstacles.

References

[1] Choi, J. and Elkaim, G.H., "Bézier Curves for Trajectory Guidance," *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2008*, WCECS 2008, 22-24 October, 2008, San Francisco, USA pp625-630.

[2] Choi, J., Curry, R.E., and Elkaim, G.H., "Path Planning based on Bézier Curve for Autonomous Ground Vehicles," *IAENG Transactions on Electrical and Electronics Engineering Volume I - Special Edition of the World Congress on Engineering and Computer Science 2008*, pp. 158-166, IEEE Computer Society, 2009.

[3] Choi, J., Curry, R.E., and Elkaim, G.H., "Smooth Path Generation Based on Bézier Curves for Autonomous Vehicles," *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2009*, WCECS 2009, 20-22 October, 2009, San Francisco, USA pp668-673.

[4] Elkaim, G.H., Connors, J., and Nagel, J., "The Overbot: An off-road autonomous ground vehicle testbed," *ION Global Navigation Satellite Systems Conference (ION-GNSS 2006)*, pp. 22-24, 9/06.

[5] Lizarraga, M. and Elkaim, G.H., "Spatially Deconflicted Path Generation for Multiple UAVs in a Bounded Airspace," *ION/IEEE Position, Location, and Navigation Symposium, ION/IEEE PLANS 2008*, Monterey, CA, 5/08

[6] Sederber, T.W., "Computer aided geometric design," *CAGD Course Notes*, Brigham Young University, Provo, UT, 2007.

[7] Skrjanc, I. and Klancar, G., "Cooperative Collision Avoidance between Multiple Robots Based on Bézier Curves," *Information Technology Interfaces, 2007 (ITI 2007)*, pp. 451-456, 6/07.

[8] *The 2005 DARPA Grand Challenge,* V36, pp. 363-405, Springer Berlin / Heidelberg, 2007.