Bézier Curve for Trajectory Guidance

Ji-wung Choi *, 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. The paper describes how the algorithms apply these properties to generate the reference trajectory for vehicles to satisfy the path constraints. Both algorithms join cubic Bézier curve segments smoothly to generate the path. Additionally, we discuss the constrained optimization problem that optimizes the resulting path for user-defined cost function. The simulation shows the generation of successful routes for autonomous vehicles using these algorithms as well as control results for a simple kinematic vehicle. Extensions of these algorithms towards navigating through an unstructured environment with limited sensor range are discussed.**

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

1. Introduction

Exploiting the versatility of autonomous vehicles for academic, industrial, and military applications will have a profound effect on future applications. Current research on control systems for autonomous vehicles demonstrates that trajectory generation is hardly a "solved" problem. For vehicle viability, it is imperative to be able to generate safe paths in real time.

Many path planning techniques for autonomous vehicles have been discussed 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 5. 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.

Connors and Elkaim [1] previously presented a method for developing feasible paths through complicated environments using a kernel function based on cubic splines. This method iteratively refines the path to compute a feasible path and thus find a collision free path in real time through an unstructured environment. This method, when implemented in a receding horizon fashion, becomes the basis for high level control. This previous method, however, result in an incomplete path planning algorithm to satisfy the computational requirements in a complicated environment. A new approach has been developed based on using Bézier curves as the seed function for the path planning algorithm as an alternative to cubic splines. The resulting path is manipulated by the control points of the bounding polygon. Though the optimization function for collision avoidance is non-linear, it can be solved quickly and efficiently. The results of the new algorithm demonstrate the generation of higher performance, more efficient, and successful routes for autonomous vehicles. Feedback control is used to track the planned path. The new algorithm is validated using simulations, and demonstrates a successful tracking result of a vehicle.

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 dynamics, and vehicle control algorithms. Section 4 proposes four path planning methods of which two are 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 and future work.

2. Bézier Curve

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]. A Bézier Curve of degree n can be represented as

$$P_{[t_0,t_1]}(t) = \sum_{i=0}^{n} B_i^n(t) P_i$$

*Ph.D. Student, Autonomous Systems Lab, Computer Engineering Department, University of California, Santa Cruz, 95064, Tel: 831-428-2146, Email: jwchoi@soe.ucsc.edu

†Assistant Professor, Autonomous Systems Lab, Computer Engineering Department, University of California, Santa Cruz, 95064, Tel: 831-459-3054, Email: elkaim@soe.ucsc.edu

Where P_i are control points such that $P(t_0) = P_0$ and $P(t_1) = P_n$, $B_i^n(t)$ is a Bernstein polynomial given by

$$B_i^n(t) = \binom{n}{i} (\frac{t_1 - t}{t_1 - t_0})^{n-i} (\frac{t - t_0}{t_1 - t_0})^i, \quad i \in \{0, 1, \ldots, n\}$$

Bézier Curves have useful properties for path planning:

- They always passes through P_0 and P_n.
- They are always tangent to the lines connecting $P_0 \rightarrow P_1$ and $P_n \rightarrow P_{n-1}$ at P_0 and 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 $P_{[t_0,t_2]}(t)$ into two segments $P_{[t_0,t_1]}(t)$ and $P_{[t_1,t_2]}(t)$ [6]. Let $\{P_0^0, P_1^0, \ldots, P_n^0\}$ denote the control points of $P_{[t_0,t_2]}(t)$. The control points of $P_{[t_0,t_1]}(t)$ and $P_{[t_1,t_2]}(t)$ can be computed by

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

where $\tau = \frac{t_1 - t_0}{t_2 - t_0}$. Then, $\{P_0^0, P_0^1, \ldots, P_0^n\}$ are the control points of $P_{[t_0,t_1]}$ and $\{P_0^n, P_1^n - 1, \ldots, P_n^0\}$ are the control points of $P_{[t_1,t_2]}$.

Remark 1. A Bézier curve $P_{[t_0,t_2]}$ always passes through the point $P(t_1) = P_0^n$ computed by applying the de Casteljau algorithm to subdivide itself into $P_{[t_0,t_1]}$ and $P_{[t_1,t_2]}$. Also, it is always tangent to $\overline{P_0^{n-1} P_1^{n-1}}$ at $P(t_1)$.

The path planning method introduced in the Section 4.3.2 is motivated by this property.

2.2. Derivatives

The derivatives of a Bézier curve, referred to as the hodograph, can be determined by its control points [6]. For a Bézier curve $P_{[t_0,t_1]}(t) = \sum_{i=0}^{n} B_i^n(t) P_i$, the first derivative can be represented as:

$$\dot{P}_{[t_0,t_1]}(t) = \sum_{i=0}^{n-1} B_i^{n-1}(t) D_i \tag{2}$$

Where D_i, control points of $\dot{P}_{[t_0,t_1]}(t)$ is

$$D_i = \frac{n}{t_1 - t_0}(P_{i+1} - Pi)$$

The higher order derivative of a Bézier curve can be obtained by using the relationship of Equation (2).

2.3. Curvature

The curvature of a n degree Bézier curve $P_{[t_0,t_1]}$ at its endpoint is given by [6]

$$\kappa(t_0) = \frac{n-1}{n} \frac{h_0}{|P_0 - P_1|^2} \tag{3}$$

$$\kappa(t_1) = \frac{n-1}{n} \frac{h_1}{|P_{n-1} - P_n|^2} \tag{4}$$

Where h_0 is the distance from P_2 to the line segment $\overline{P_0 P_1}$, h_1 is the distance from P_{N-2} to the line segment $\overline{P_{n-1} P_n}$.

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 $W_i \in \mathbb{R}^2$ 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, j-th widths of each segment between two waypoints, $j \in \{1, \ldots, N-1\}$.

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. 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 velocity. 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 feed forward path planning with feedback corrections as illustrated in Figure 1 [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 c is computed with the current longitudinal velocity and heading of the vehicle from the current position. c is projected onto the reference trajectory at point p such that $\overline{cp}$ is normal to the tangent at p. The cross track error y_{err} is defined by the distance between c and p. The steering control ω uses PID controller with respect to cross track error y_{err}.

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

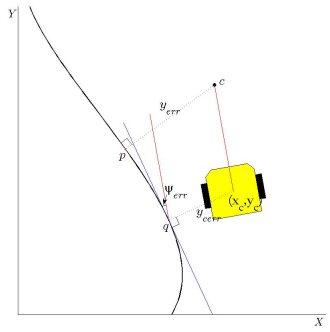


Figure 1: The position error of the vehicle is measured from a point c, projected in front of the vehicle, and unto the ideal curve to point p.

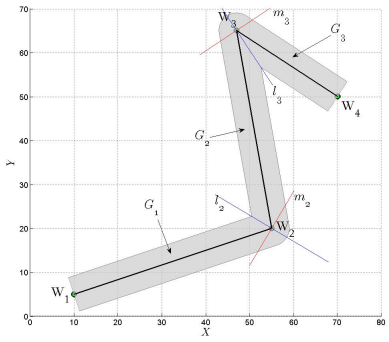


Figure 2: The course with four waypoints. Gray area is the permitted area for vehicles under a corridor constraint.

4. Path Planning Algorithm

In this section, four path planning methods are proposed. To describe the methods, we consider the course with waypoints $\mathbf{W} = \{W_i\}$, $i \in \{1, \ldots, N\}$ and corridor width $w_k = 8$, $k \in \{1, \ldots, N-1\}$ for $N = 4$ as illustrated in Figure 2. The location of waypoints are given by two-dimensional world coordinates (X, Y) in $meter$ scale: $W_1 = (10, 5)$, $W_2 = (55, 20)$, $W_3 = (47, 65)$, $W_4 = (70, 50)$.

To describe the algorithms, let us denote l_j as the bisector of $\angle W_{j-1}W_jW_{j+1}$ for $j \in \{2, \ldots, N-1\}$ and m_j as the normal line to l_j at the intersect of the planned path and l_j. The course is divided into segments G_k by l_{k+1}. G_k indicates the permitted area for vehicles under corridor constraint w_k, from W_k to W_{k+1}.

4.1. Path Planning Based on Center Lines

The simplest path planning method is to follow a series of center lines that connect neighboring waypoints, plotted as a black line in Figure 2. Although this method has advantages of simplicity and optimality in terms of the shortest path, the sharp turns at waypoints are not feasible for vehicles in practice. It will result in a position error varying as a function of the longitudinal velocity and the maximum steer angle. Section 5 details the simulation and Figure 6(a) shows the result of this path.

4.2. Path Planning Using Circular Arcs for Turns

To reduce position error at segment transitions with sharp turns, another path planning method is proposed with circular arc. Under the assumption that the slip angle of the vehicle is zero in lateral motion, the ideal shape of a path with a turn is a circular arc as it has constant radius of curvature. Figure 3 shows the potential trajectories around W_2. They consist of a circular arc and its tangents. It is interesting to note the following property from the geometry.

Remark 2. The tangent of the trajectory on the bisector l_j is normal to l_j.

This property will be used as a constraint of the path planning methods introduced in the Section 4.3.1 and 4.3.2.

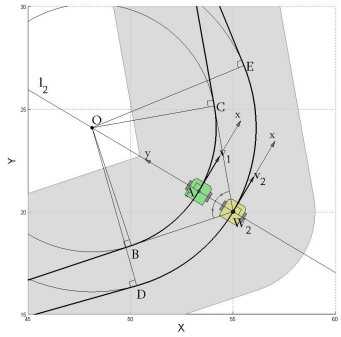


Figure 3: An enlarged path based on circular arc for a turn at W_2.

4.3. Path Planning Based on Bézier Curves

In this section, two methods of path planning based on Bézier curves are proposed. Both methods are motivated by Remark 2 and are developed to preserve it.

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]. The basic requirement of path planning is to pass through the beginning point and the end

point with different desired velocities. The least degree of the Bézier curve that can satisfy this requirement is three (See properties of Bézier curves in Section 2). Both methods use a set of cubic Bézier curves.

The cubic Bézier curves used for the path planning are denoted as $P_{[t_{i-1},t_i]}(t) = \sum_{k=0}^{3} B_k^3(t) P_{k,i}$ for $i \in \{1, \ldots, M\}$ where M is the total number of the Bézier curves. The planned path $P(t)$ for $t \in [t_0, t_M]$ is represented as

$$P(t) = \{P_{[t_{i-1},t_i]}(t)\}, \quad i \in \{1, \ldots, M\}$$

4.3.1 Path Planning Placing Bézier Curves within Segments

In this path planning method, the cubic Bézier curves $P_{[t_{i-1},t_i]}(t) = \sum_{k=0}^{3} B_k^3(t) P_{k,i}$ for $i \in \{1, \ldots, N-1\}$ are used within each segment G_i as shown in Figure 4. The planned path $P(t)$ are designed such that it begins from W_1 with the heading of $\overrightarrow{W_1 W_2}$ and ends to W_N with the heading of $\overrightarrow{W_{N-1} W_N}$. Furthermore, the corridor constraint and Remark 2 are satisfied.

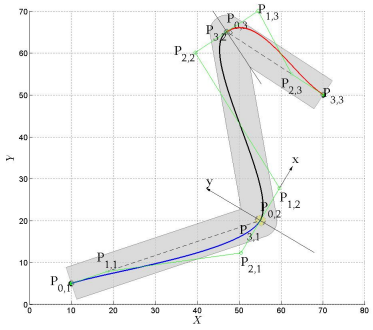


Figure 4: The planned path based on cubic Bézier curves used within segments when $T_i = W_i$. The Bézier curves are plotted with different colors: blue, black, and red.

The control points of $P_{[t_{i-1},t_i]}(t)$, $P_{k,i}$ are determined to maintain these conditions. The beginning point of $P_{[t_0,t_1]}(t)$, $P_{0,1}$ is W_1. The end point of $P_{[t_{N-2},t_{N-1}]}(t)$, $P_{3,N-1}$ is W_N. Other beginning/end points denoted as T_j are chosen on the bisectors l_j for $j \in \{2, \ldots, N-1\}$ and bounded within the corridor. Then it is assigned to $P_{3,j-1}$ and $P_{0,j}$. $\{P_{0,i}, P_{1,i}, P_{2,i}, P_{3,i}\}$ always lie within the area of G_i so that the resulting Bézier curve satisfies the corridor constraint by the convex hull property. Also, $\{P_{1,i}, P_{2,i}\}$ are chosen such that the derivatives of $P_{[t_{j-2},t_{j-1}]}(t)$ and $P_{[t_{j-1},t_j]}(t)$ at T_j are continuous and are normal to l_j.

These can be formulated as the following constraints:

$$P_{[t_0,t_1]}(t_0) = P_{0,1} = W_1 \tag{5}$$

$$P_{[t_{N-2},t_{N-1}]}(t_{N-1}) = P_{3,N-1} = W_N \tag{6}$$

$$\frac{3}{t_1 - t_0}(P_{1,1} - W_1) = c_i(W_2 - W_1) \tag{7}$$

$$\frac{3}{t_{N-1} - t_{N-2}}(W_N - P_{2,N-1}) = c_f(W_N - W_{N-1}) \tag{8}$$

$$P_{3,j-1} = P_{0,j} \tag{9}$$

$$\frac{3}{t_{j-1} - t_{j-2}}(P_{3,j-1} - P_{2,j-1}) = \frac{3}{t_j - t_{j-1}}(P_{1,j} - P_{0,j}) \tag{10}$$

$$T_j \in l_j \tag{11}$$

$$|T_j - W_j| < \frac{1}{2} w_{j-1}, \quad |T_j - W_j| < \frac{1}{2} w_j \tag{12}$$

$$P_{1,i} \in G_i, \quad P_{2,i} \in G_i \tag{13}$$

$$\frac{3}{t_{j-1} - t_{j-2}}(P_{1,j-1} - P_{0,j-1}) \cdot l_j = 0 \tag{14}$$

Where $c_i \in \mathbb{R}$, $c_f \in \mathbb{R}$ and velocity constraints (7), (8), and (14) are represented by applying (2). $[t_{i-1}, t_i]$ are assumed to be given. Then $\mathbf{P} = \{P_{1,i}, P_{2,i}\}$ and $\mathbf{T} = \{T_j\}$ are computed by minimizing the constrained optimization problem:

$$\min_{\mathbf{P},\mathbf{T}} J = \sum_{i=1}^{N-1} J_i \tag{15}$$

subject to (7), (8), (10), (11), (12), (13), and (14).

Where J_i is the cost function of $P_{[t_{i-1},t_i]}(t)$ which is defined in Section 5. As the result, the planned trajectory passes through $\{W_1, T_2, \ldots, T_{N-2}, W_{N-1}\}$ with Remark 2 preserved.

4.3.2 Path Planning Placing Mid-points of Bézier Curves on Bisectors of Turns

In the section 4.3.1, a Bézier curve is used within each segment. Another path planning method places the mid-points of the Bézier curves on the bisectors l_j, $j \in \{2, \ldots, N-1\}$ using the de Casteljau algorithm. In this method, cubic Bézier curves $P_{[t_{i-1},t_i]}(t) = \sum_{k=0}^{3} B_k^3(t) P_{k,i}^0$ for $i \in \{1, \ldots, 2N-3\}$ are used.

The local area of the course around W_j can be seen as symmetric with respect to l_j. Thus the cubic Bézier curves $P_{[t_{i'-1},t_{i'}]}(t)$, $i' \in \{2, 4, \ldots, 2N-4\}$ used for this area will also be symmetric with respect to $l_{i'/2+1}$. In other words, $P_{0,i'}^0$ and $P_{3,i'}^0$ are symmetric with respect to $l_{i'/2+1}$. So are $P_{1,i'}^0$ and $P_{2,i'}^0$. After applying the de Casteljau algorithm with $\tau = 0.5$ to the curve, the mid-point $P_{0,i'}^3$ is on $l_{i'/2+1}$. Then Bézier curves $P_{[t_{i''-1},t_{i''}]}(t)$ for $i'' \in \{1, 3, \ldots, 2N-$

3} are used for the rest part of the path. The constraints imposed on the planned path are as follows:

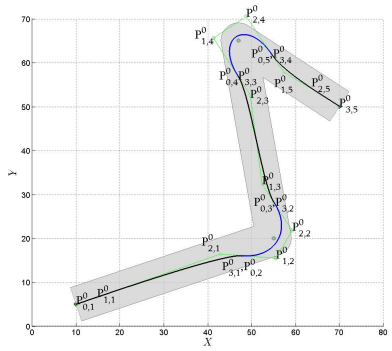


Figure 5: The path based on cubic Bézier curves placed such that their mid-point lies on the bisectors.

$$P^0_{0,1} = W_1 \tag{16}$$

$$P^0_{3,2N-3} = W_N \tag{17}$$

$$\frac{3}{t_1 - t_0}(P^0_{1,1} - W_1) = c_i(W_2 - W_1) \tag{18}$$

$$\frac{3}{t_{2N-4} - t_{2N-3}}(W_N - P^0_{2,2N-3}) = c_f(W_N - W_{N-1}) \tag{19}$$

$$P^0_{3,i} = P^0_{0,i+1} \tag{20}$$

$$\frac{3}{t_i - t_{i-1}}(P^0_{3,i} - P^0_{2,i}) = \frac{3}{t_{i+1} - t_i}(P^0_{1,i+1} - P^0_{0,i+1}) \tag{21}$$

$$\left.\begin{array}{ll} (P^0_{0,i'} + P^0_{3,i'})/2 & \in l_{i'/2+1} \\ (P^0_{1,i'} + P^0_{2,i'})/2 & \in l_{i'/2+1} \end{array}\right\} \tag{22}$$

$$P^0_{3,i'} \in G_{i'/2} \tag{23}$$

$$\left.\begin{array}{ll} P^0_{0,i'} & \in G_{i'/2} \\ \frac{1}{2}P^0_{0,i'} + \frac{1}{2}P^0_{1,i'} & \in G_{i'/2} \\ \frac{1}{4}P^0_{0,i'} + \frac{1}{2}P^0_{1,i'} + \frac{1}{4}P^0_{2,i'} & \in G_{i'/2} \end{array}\right\} \tag{24}$$

$$\left.\begin{array}{ll} |\frac{1}{8}P^0_{0,i'} + \frac{3}{8}P^0_{1,i'} + \frac{3}{8}P^0_{2,i'} + \frac{1}{8}P^0_{3,i'}| & < \frac{1}{2}w_{i'/2} \\ |\frac{1}{8}P^0_{0,i'} + \frac{3}{8}P^0_{1,i'} + \frac{3}{8}P^0_{2,i'} + \frac{1}{8}P^0_{3,i'}| & < \frac{1}{2}w_{i'/2+1} \end{array}\right\} \tag{25}$$

$$P^0_{1,i''} \in G_{(i''+1)/2}, \quad P^0_{2,i''} \in G_{(i''+1)/2} \tag{26}$$

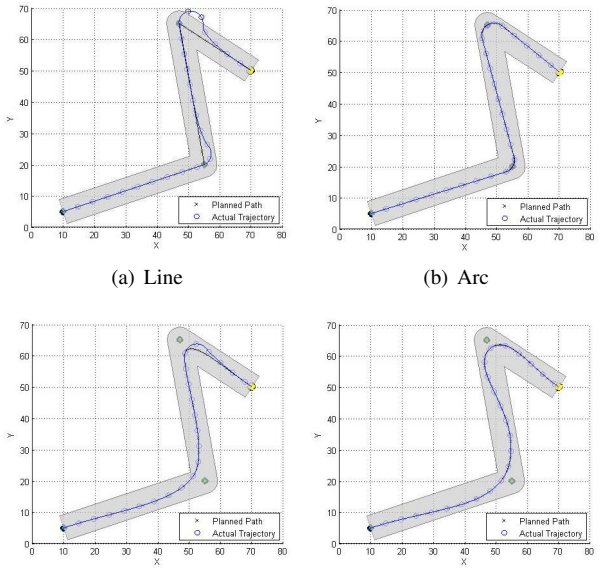
Where $c_i \in \mathbb{R}$, $c_f \in \mathbb{R}$, and boundary constraints (24) and (25) are represented by applying (1). $[t_{i-1}, t_i]$ are assumed to be given. Then $\mathbf{P} = \{P^0_{0,i'}, P^0_{1,i'}\}$ and $\mathbf{Q} = \{P^0_{1,i''}, P^0_{2,i''}\}$ are computed by minimizing the constrained optimization problem:

$$\min_{\mathbf{P},\mathbf{Q}} J = \sum_{i=1}^{2N-3} J_i \tag{27}$$

subject to (18), (19), (21), (22), (24), (25), and (26).

Remarkable feature of this method is the fact that it poses desirable position and velocity of the mid-point of a Bézier curve as well as those of the beginning and the end points. Furthermore, the convex hull property is tested for $\{P^0_{0,i'}, P^1_{0,i'}, P^2_{0,i'}, P^3_{0,i'}\}$ of the divided curves instead of $\{P^0_{0,i'}, P^0_{1,i'}, P^0_{2,i'}, P^0_{3,i'}\}$. Since the curve and the course is locally symmetrical, we do not need to check $\{P^3_{0,i'}, P^2_{1,i'}, P^1_{2,i'}, P^0_{3,i'}\}$. As the result, it comes up with more tight condition for curves against the corridor constraint than checking $\{P^0_{0,i'}, P^0_{1,i'}, P^0_{2,i'}, P^0_{3,i'}\}$ by performing the same number of tests.

5. Simulation Results



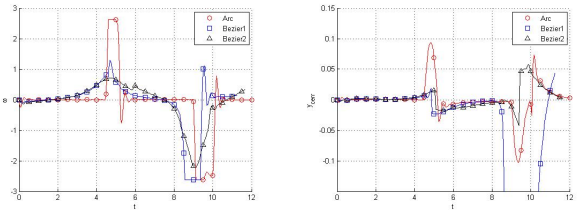
(a) Line (b) Arc

(c) Bézier1 (d) Bézier2

Figure 6: Tracking simulation results (blue) over the planned paths (black).

Simulation performed in this paper uses the course as shown in Figure 2. Initial position and heading are assumed to fit to the first waypoint and the direction to the second waypoint respectively. The constant longitudinal velocity $v(t) = 10\ m/s$ is used. The magnitude of ω is bounded within $|\omega|_{max} = 2.618\ rad/s$. 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, 4.2, 4.3.1, and 4.3.2 are denoted as *Line*, *Arc*, *Bézier1*, and *Bézier2* respectively. The simulated trajectory of a vehicle that tracks



(a) The steering control ω. (b) The cross track error y_{cerr}.

Figure 7: The steering control ω and the cross track error y_{cerr}.

the planned path by *Line* is shown in Figure 6(a). Although the vehicle tracks the straight parts of the planned path accurately, it overshoots in turns resulting in a large position error due to maximum steer angle limitation. Decreasing speed as the vehicle approaches to the turning area can reduce the position error. However, in this simulation, the longitudinal velocity v was kept constant which magnifies the tracking errors with this method of path planning.

Figure 6(b) shows the tracking result of *Arc*. Where uniform radius $R = 5$ is used. It satisfies the condition of $R \geq \frac{v}{|\omega|_{max}} = 3.8197$ for vehicles to converge to the planned circular arc path.

Figure 6(c) is the tracking result over the path planned by *Bézier1*. This path obtained by solving Equation (15) with

$$J_i = \int_{t_{i-1}}^{t_i} \left[(\dot{P}_{[t_{i-1},t_i]}(t))^2 + (\ddot{P}_{[t_{i-1},t_i]}(t))^2 \right] dt$$

Where $t_i = i$. J_i penalizes $\dot{P}$ and $\ddot{P}$ to give smoothness to each Bézier curve.

Figure 6(d) is the tracking result over the path planned by *Bézier2*. This path obtained by solving Equation (27) with

$$J_i = \int_{t_{i-1}}^{t_i} \left[(\dot{P}_{[t_{i-1},t_i]}(t))^2 + (\ddot{P}_{[t_{i-1},t_i]}(t))^2 \right] dt, \quad i \in \{1, 3, \ldots, 2N-3\} \tag{28}$$

$$J_i = \int_{t_{i-1}}^{t_i} \left[(\dot{P}_{[t_{i-1},t_i]}(t))^2 + (\ddot{P}_{[t_{i-1},t_i]}(t))^2 \right] dt + \frac{20}{\kappa(t_i)}, \quad i \in \{2, 4, \ldots, 2N-4\} \tag{29}$$

Note that two different J_i are used. Bézier curves used in turns are additionally penalized inversely to the curvature at their endpoints in Equation (29). This leads to a resulting path with smooth turns.

6. Summary and Conclusions

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 converges to the planned path successfully.

These path planning algorithms will be implemented on the Overbot [4], the autonomous ground vehicle at Autonomous Systems Lab in UCSC.

In this work, proposed algorithms only generate nominal path. We will extend and expand our work in receding horizon obstacle avoidance for an autonomous ground vehicle. Enabling autonomous vehicles to detect unknown obstacles and safely avoid them is essential to future operations. Additionally, receding horizon control methods will be applied to generate real-time Bézier-based optimal trajectories.

References

[1] J. Connors, G. Elkaim, "Analysis of a Spline Based, Obstacle Avoiding Path Planning Algorithm," *IEEE Vehicle Technology Conference, IEEE VTC 2007,* Dublin, Ireland, Apr. 22-25, 2007

[2] J. Connors, G. Elkaim, "Experimental Results for Spline Based Obstacle Avoidance of an Off-Road Ground Vehicle," *ION Global Navigation Satellite Systems Conference, ION GNSS 2007,* Fort Worth, TX, Sept. 25-28, 2007

[3] J. Connors, G. Elkaim, "Manipulating B-Spline Based Paths for Obstacle Avoidance in Autonomous Ground Vehicles," *ION National Technical Meeting, ION NTM 2007,* San Diego, CA, Jan. 22-24, 2007

[4] G. Elkaim, J. Connors, and J. Nagel, "The Overbot: An off-road autonomous ground vehicle testbed," *ION Global Navigation Satellite Systems Conference (ION-GNSS 2006)*, 1, Sept. p.22-24, 2006.

[5] M. Lizarraga, G. Elkaim, "Spatially Deconflicted Path Generation for Multiple UAVs in a Bounded Airspace," *ION/IEEE Position, Location, and Navigation Symposium, ION/IEEE PLANS 2008,* Monterey, CA, May 5-8, 2008

[6] T. W. Sederber, "Computer aided geometric design," *CAGD Course Notes,* Brigham Young University, Provo, UT, 84602, April 2007.

[7] I. Skrjanc, G. Klancar "Cooperative Collision Avoidance between Multiple Robots Based on Bzier Curves," *Information Technology Interfaces, 2007 (ITI 2007),* p.451-456, June 25-28, 2007.

[8] *The 2005 DARPA Grand Challenge,* vol. 36/2007, p.363-405, Springer Berlin / Heidelberg, 2007.