Real-Time Obstacle-Avoiding Path Planning for Mobile Robots

Ji-wung Choi* and Renwick E. Curry † and Gabriel Hugh Elkaim ‡
University of California, Santa Cruz, CA, 95064, USA

In this paper a computationally effective trajectory generation algorithm of mobile robots is proposed. The algorithm plans a reference path based on Voronoi diagram and Bézier curves, that meet obstacle avoidance criteria. Bézier curves defining the path are created such that the circumference convex polygon of their control points miss all obstacles. To give smoothness, they are connected under C_1 continuity constraint. In addition, the first Bézier curve is created to satisfy the initial heading constraint and to minimize the maximum curvature of the curve. For the mission, this paper analyzes the algebraic condition of control points of a quadratic Bézier curve to minimize the maximum curvature. The numerical simulations demonstrate smooth trajectory generation with satisfaction of obstacle avoidance in an unknown environment by applying the proposed algorithm in a receding horizon fashion.

Nomenclature

$\mathbf{Q}$	Bezier curve
$\mathbf{q}_i$	$i+1$-th control points determining $\mathbf{Q}$, $i = 0, 1, \ldots$
x_i	x-coordinate value of $\mathbf{q}_i$
y_i	y-coordinate value of $\mathbf{q}_i$
α	Length between the first and the second control points: $\|\mathbf{q}_1 - \mathbf{q}_0\|$
β	Length between the second and the third control points: $\|\mathbf{q}_2 - \mathbf{q}_1\|$
θ	Heading difference from $\mathbf{q}_1 - \mathbf{q}_0$ to $\mathbf{q}_2 - \mathbf{q}_1$
λ	Parameter defining a Bezier curve
κ	Curvature
$\mathbf{o}_i$	Obstacle, $i = 1, 2, \ldots$
$\mathbf{s}$	Start point
$\mathbf{t}$	Target point
$\mathbf{p}_i$	Vertices of a Dijkstra's shortest path, $i = 0, 1, \ldots$

Subscript

max	maximum value

I. Introduction

Many path planning techniques for robots have been discussed in the literature. The algorithms can be categorized as mainly three: roadmap (*visibility graph*, *Voronoi diagram*), cell decomposition (*exact* and *approximate*), and potential field.[1]

Nilsson[2] investigated the visibility graph algorithm for motion planning of a mobile robot system. Hsu and Latombe[3] introduced *expansiveness* to characterize a family of robot configuration spaces whose connectivity can be effectively captured by a roadmap of randomly-sampled milestones and developed a new randomized planning algorithm based on analysis of the expansiveness. Lozano-Pérez[4] introduced the approximate cell decomposition approach for the automatic planning of manipulator transfer movements. Chatila[5] applied an exact decomposition of the workspace into convex cells for motion planning with incomplete knowledge for a mobile robot. Khatib[6] implemented

*Ph.D. Candidate, Department of Computer Engineering, UCSC, 1156 High St., Santa Cruz, CA, 95064, USA, and AIAA Student Member.
†Adjunct Professor, Department of Computer Engineering, UCSC, 1156 High St., Santa Cruz, CA, 95064, USA, and AIAA Member.
‡Associate Professor, Department of Computer Engineering, UCSC, 1156 High St., Santa Cruz, CA, 95064, USA.

a real-time collision avoidance module in a robot controller by applying the potential field approach. Barraquand and Latombe[7] incorporated a potential field method and randomized planning algorithm to escape from local minima.

This paper uses Bézier curves for path smoothing. Since Bézier curves have useful properties for path generation problem, they have been applied to generate the reference trajectory. The Cornell University Team for the 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. Choi[9] has presented path planning algorithms based on Bézier curves for autonomous vehicles with waypoints and corridor constraints. The algorithms join Bézier curve segments smoothly to generate the path. Additionally, the constrained optimization problem that optimizes the resulting path for a user-defined cost function is discussed. Sahraei[10] has presented a real-time motion planning algorithm for mobile robots by applying Voronoi diagram roadmap and two Bézier curves to path smoothing. The paper has claimed that the algorithm satisfies obstacle avoidance as well as time optimality given in discrete time system.

This paper shows that Sahraei's algorithm is problematic. To resolve the problems, a new real-time motion planning algorithm is proposed, which also satisfies obstacle avoidance. The numerical simulations provided in this paper demonstrate a better solution to the problem of motion planning by the proposed algorithm than Sahraei's. Furthermore, the safe path generation is demonstrated in unknown environment by applying the proposed algorithm in a receding horizon fashion.

A. Background

1. Bézier Curve

A Bézier Curve of degree n, $\mathbf{Q}$ is defined by $n+1$ control points $\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_n$:

$$\mathbf{Q}(\lambda) = \sum_{i=0}^{n} B_i^n(\lambda)\mathbf{q}_i, \quad \lambda \in [0,1] \tag{1}$$

where $B_i^n(\lambda)$ is Bernstein polynomial given by

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

Bézier Curves have useful properties for path planning:

- They always pass through $\mathbf{q}_0$ and $\mathbf{q}_n$.
- They are always tangent to $\overline{\mathbf{q}_0\mathbf{q}_1}$ and $\overline{\mathbf{q}_{n-1}\mathbf{q}_n}$ at $\mathbf{q}_0$ and $\mathbf{q}_n$ respectively.
- They always lie within the convex hull of control points.

The derivatives of a Bézier curve can be determined by its control points:

$$\dot{\mathbf{Q}}(\lambda) = \sum_{i=0}^{n-1} B_i^{n-1}(\lambda)\mathbf{d}_i \tag{2}$$

Where $\mathbf{d}_i$, control points of $\dot{\mathbf{Q}}$ is

$$\mathbf{d}_i = n(\mathbf{q}_{i+1} - \mathbf{q}_i) \tag{3}$$

The curvature of a Bézier curve $\mathbf{Q}(\lambda) = \big(x(\lambda), y(\lambda)\big)$ at λ is given by

$$\kappa(\lambda) = \frac{\dot{x}(\lambda)\ddot{y}(\lambda) - \dot{y}(\lambda)\ddot{x}(\lambda)}{(\dot{x}^2(\lambda) + \dot{y}^2(\lambda))^{\frac{3}{2}}} \tag{4}$$

The de Casteljau algorithm describes a recursive process to subdivide a Bézier curve $\mathbf{Q}$ into two segments. The subdivided segments are also Bézier curves. Let $\{\mathbf{q}_0^0, \mathbf{q}_1^0, \ldots, \mathbf{q}_n^0\}$ denote the control points of $\mathbf{Q}$. The control points of the segments can be computed by

$$\mathbf{q}_i^j = (1-\tau)\mathbf{q}_i^{j-1} + \tau\mathbf{q}_{i+1}^{j-1}, \quad \tau \in (0,1), \;\; j = 1,\ldots,n, \;\; i = 0,\ldots,n-j \tag{5}$$

Then, $\{\mathbf{q}_0^0, \mathbf{q}_0^1, \ldots, \mathbf{q}_0^n\}$ are the control points of one segment and $\{\mathbf{q}_0^n, \mathbf{q}_1^{n-1}, \ldots, \mathbf{q}_n^0\}$ are the another.

2. *Voronoi Diagram*

A Voronoi diagram is the partitioning of a plane with n distinct points, called the sites, into n cells. The partitioning is made such that each cell includes one site and every point in a given cell is closer to the captured site than to any other site. We denote the Voronoi diagram of a set of n sites $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n\}$ by $Vor(\mathbf{W})$. $\mathscr{V}(\mathbf{w}_i)$ denotes the Voronoi cell of the site $\mathbf{w}_i$.[11]

Proposition 1. *$\mathscr{V}(\mathbf{w}_i)$ is a convex polygon.*[11]

II. Sahraei's Algorithm

Sahraei[10] proposed a trajectory generation algorithm for mobile robots. To describe this algorithm, let obstacles be denoted by $\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_{n_o}$, where n_o is the number of obstacles. The first step is to construct the Voronoi diagram of obstacles to find a path that avoids obstacles: $Vor(\mathbf{O})$, $\mathbf{O} = \{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_{n_o}\}$. After constructing the Voronoi diagram, the start and the target point, $\mathbf{s}$ and $\mathbf{t}$ are added to the Voronoi graph with corresponding edges which connect these two points to their cell vertices except for the ones that collide with obstacles. Then Dijkstra's shortest path algorithm is run. The resulting path is the shortest path (*SP*) whose edges are in the Voronoi graph. Two Bézier curves are used to find a smooth path near *SP* with regards to initial and final conditions. Let $\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_n$ denote the vertices of *SP* and $\mathbf{p}_0$, and $\mathbf{p}_n$ denote $\mathbf{s}$ and $\mathbf{t}$, respectively. The first Bézier curve, $\mathbf{Q}_a(\lambda)$ for $\lambda \in [0,1]$, is constructed by $\mathbf{p}_0$, $\mathbf{q}$, $\mathbf{r}$, and $\mathbf{p}_1$, where control points $\mathbf{q}$ and $\mathbf{r}$ are introduced to satisfy slope of initial velocity ($\mathbf{v}_0$) constraint and continuity of curve and its slope in $\mathbf{p}_1$. The second Bézier curve $\mathbf{Q}_b(\lambda)$ is constructed by $\mathbf{p}_1, \ldots, \mathbf{p}_n$. Following equations describe boundary conditions:

$$\frac{\dot{\mathbf{Q}}_a(0)}{\|\dot{\mathbf{Q}}_a(0)\|} = \frac{\mathbf{v}_0}{\|\mathbf{v}_0\|}, \quad \frac{\dot{\mathbf{Q}}_a(1)}{\|\dot{\mathbf{Q}}_a(1)\|} = \frac{\dot{\mathbf{Q}}_b(0)}{\|\dot{\mathbf{Q}}_b(0)\|}.$$

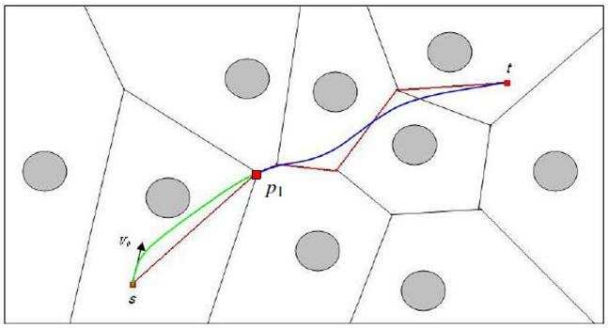


Figure 1. A path resulted by Sahraei's method.

Figure 1 shows an example of the paths.

Sahraei's paper described the constraint to have $\mathbf{Q}_a$ miss the obstacles: $\|\dot{\mathbf{Q}}_a(0)\|$ and $\|\dot{\mathbf{Q}}_a(1)\|$ are constrained as the circumferential convex polygon of $\mathbf{p}_0$, $\mathbf{q}$, $\mathbf{r}$, and $\mathbf{p}_1$ would not collide with any obstacle.[10] However, it did not explain how to select the length of $\mathbf{q}-\mathbf{p}_0$ and $\mathbf{r}-\mathbf{p}_1$. In addition, it did not do any calculation to ensure that $\mathbf{Q}_b$ misses the obstacles. There is a possibility that $\mathbf{Q}_b$ intersects obstacles as presented in section IV.

III. Proposed Algorithm

To resolve drawbacks of Sahraei's method, this paper proposes new algorithm to generate a collision-free path.

A. Voronoi Cell Decomposition

The first step of the algorithm is to construct a Voronoi diagram. While Sahraei's algorithm constructs the Voronoi diagram of obstacles, the proposed algorithm constructs the one of obstacles, $\mathbf{s}$ and $\mathbf{t}$: $Vor(\mathbf{S})$, $\mathbf{S} = \{\mathbf{o}_1, \ldots, \mathbf{o}_{n_o}, \mathbf{s}, \mathbf{t}\}$. As the result, $\mathbf{s}$ and $\mathbf{t}$ are separated from obstacles by cells. Then the graph $G = \{V, E\}$ is constructed, where V is a set of the Voronoi vertices, E is a set of the Voronoi edges plus edges which connect $\mathbf{s}$ and $\mathbf{t}$ to vertices of $\mathscr{V}(\mathbf{s})$ and $\mathscr{V}(\mathbf{t})$, respectively. Dijkstra's algorithm for the graph G results in *SP*, the collision-free sketch path $\mathbf{p}_0, \ldots, \mathbf{p}_n$, where $\mathbf{p}_0 = \mathbf{s}$ and $\mathbf{p}_n = \mathbf{t}$.

Note that *SP* is discontinuous in the first derivative at the joint points and hence do not guarantee kinematic feasibility of the robot tracking it. So multiple Bézier curves $\mathbf{Q}_0, \mathbf{Q}_1, \ldots, \mathbf{Q}_m$ are used to smooth *SP*. Let $\mathbf{q}_j(i)$ denote control points of $\mathbf{Q}_i$ for $j = 0, 1, \ldots, l_i$, where l_i is the degree of $\mathbf{Q}_i$. All control points of the curves are located on *SP* except for one additional control point $\mathbf{q}_1(0)$ to satisfy the initial velocity heading constraint. $\mathbf{q}_0(0)$ and $\mathbf{q}_{l_m}(m)$ are located on $\mathbf{p}_0$ and $\mathbf{p}_n$ to satisfy the initial and the final position constraints.

Every Bézier curve is constructed such that the circumferential convex polygon of its control points misses obstacles. Since cells of $Vor(\mathbf{S})$ have at most one obstacle, they can be good guidance to compute the convex polygon. In other words, for collision detection, line segments of the polygons divided by the Voronoi cells are needed to be tested only over the obstacles captured in the cells. This is more detailed in next subsections.

B. Choosing Control Points for Initial Velocity Constraint

For the initial velocity heading constraint, let us calculate the control points of the first Bézier curve $\mathbf{Q}_0$. In this section, index of 0 for $\mathbf{Q}_0$ is dropped for simplicity. As described earlier, its first control point is set to $\mathbf{p}_0$: $\mathbf{q}_0 = \mathbf{p}_0$. We are to find the location of $\mathbf{q}_1$ on the direction of $\mathbf{v}_0$ from $\mathbf{p}_0$ so that $\dot{\mathbf{Q}}(0)$ is collinear to $\mathbf{v}_0$. It can be represented as

$$\mathbf{q}_1 = \mathbf{p}_0 + \alpha \frac{\mathbf{v}_0}{\|\mathbf{v}_0\|} = \mathbf{q}_0 + \alpha \frac{\mathbf{v}_0}{\|\mathbf{v}_0\|}, \quad \alpha > 0 \tag{6}$$

where α is the length between the first and the second control points, $\|\mathbf{q}_1 - \mathbf{q}_0\|$. Also, in order for $\mathbf{Q}$ to approach *SP*, two control points are selected on $\overline{\mathbf{p}_1\mathbf{p}_2}$. The number and location of the control points are chosen depending on the configuration of $\mathbf{p}_0$, $\mathbf{v}_0$ and obstacles. It can be categorized as following.

- Case I : $\mathbf{p}_0$, $\mathbf{p}_1$ and $\mathbf{p}_2$ are collinear. $\mathbf{v}_0$ is co-directional to $\mathbf{p}_1 - \mathbf{p}_0$.
- Case II : $\mathbf{q}_1$, given by (6), is to the opposite side (left or right) of $\overline{\mathbf{p}_0\mathbf{p}_1}$ as $\mathbf{p}_2$.
- Case III : $\mathbf{q}_1$ is to the same side of $\overline{\mathbf{p}_0\mathbf{p}_1}$ as $\mathbf{p}_2$. $\overrightarrow{\mathbf{p}_0\mathbf{q}_1}$ can not intersect $\overline{\mathbf{p}_1\mathbf{p}_2}$ without intersecting an obstacle.
- Case IV : $\overrightarrow{\mathbf{p}_0\mathbf{q}_1}$ intersects $\overline{\mathbf{p}_1\mathbf{p}_2}$ without intersecting an obstacle.

Following subsections present how to compute control points of $\mathbf{Q}$ for each case. If $\mathbf{p}_2$ is taken as the control point of $\mathbf{Q}$, then $\mathbf{Q}$ can be extended as taking in more control points on the rest segments of *SP* (See section C).

1. *Case I*

This case is the simplest one. $\mathbf{Q}$ takes in $\mathbf{p}_0$ and $\mathbf{p}_2$ as its control points.

2. *Case II*

Refer to figure 2 that illustrates an example of the case, in which the transparent grey area indicates circumferential convex polygon of the control points of $\mathbf{Q}$: $\{\mathbf{p}_0 = \mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3\}$.

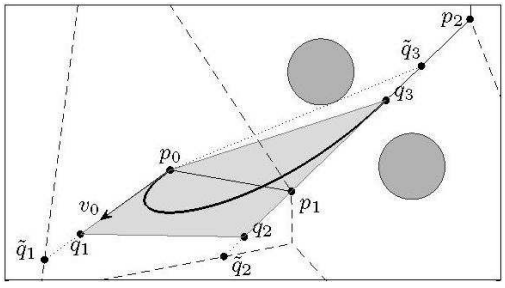


Figure 2. An example of *Case II*.

$\mathbf{q}_1$ is bounded by $\mathscr{V}(\mathbf{p}_0)$. Let $\tilde{\mathbf{q}}_1$ denote the farthest $\mathbf{q}_1$ from $\mathbf{p}_0$. To have $\mathbf{Q}$ approach to *SP*, two another control points $\mathbf{q}_2$ and $\mathbf{q}_3$ are selected such that the points, $\mathbf{p}_1$ and $\mathbf{p}_2$ are collinear and that $\mathbf{q}_3$ is closer to $\mathbf{p}_2$ than $\mathbf{q}_2$ is. One of the boundary points for them, $\tilde{\mathbf{q}}_2$ is defined depending on configuration of $\overline{\mathbf{p}_2\mathbf{p}_1}$ and $\mathscr{V}(\mathbf{p}_0)$. If $\overline{\mathbf{p}_1\mathbf{p}_2}$ is extended to inside of $\mathscr{V}(\mathbf{p}_0)$ as shown in the figure, then $\tilde{\mathbf{q}}_2$ is the intersection of the line and $\mathscr{V}(\mathbf{p}_0)$. Otherwise, $\tilde{\mathbf{q}}_2$ is $\mathbf{p}_1$. The other boundary point $\tilde{\mathbf{q}}_3$, on $\overline{\mathbf{p}_1\mathbf{p}_2}$, is defined such that $\triangle\mathbf{p}_0\mathbf{p}_1\tilde{\mathbf{q}}_3$ is the biggest triangle not to contain nor to intersect any obstacle. (It is sufficient to test only the obstacles whose Voronoi cells have the vertex $\mathbf{p}_1$.) If $\tilde{\mathbf{q}}_3$ is $\mathbf{p}_2$ then $\mathbf{Q}$ takes in $\{\mathbf{p}_0, \tilde{\mathbf{q}}_1, \mathbf{p}_1, \mathbf{p}_2\}$ as its control points and extends by adding more on the rest of *SP* as described in section C. Otherwise, $\mathbf{Q}$ is the cubic Bézier curve constructed by $\{\mathbf{p}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3\}$ in which $\mathbf{q}_1$ is selected on $\overline{\mathbf{p}_0\tilde{\mathbf{q}}_1}$, and $\mathbf{q}_2$ and $\mathbf{q}_3$ on $\overline{\tilde{\mathbf{q}}_2\tilde{\mathbf{q}}_3}$. Under the constraint, the convex hull of $\mathbf{p}_0\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3$ lies inside of the convex hull of $\mathbf{p}_0\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_2\tilde{\mathbf{q}}_3$, which misses any obstacles and, thus, misses obstacles by the convex hull property of Bézier curves.

To give the resulting path smoothness, $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$ are computed to minimize $|\kappa|_{max}$, maximum magnitude of curvature of $\mathbf{Q}$. Since κ and $\dot{\kappa}$ of a cubic Bézier curve consist of high degree of polynomials, it is difficult to compute

$|\kappa|_{max}$. The alternative way is to approximate the cubic curve by two quadratic curves. Firstly, the cubic curve is subdivided into two cubic curves $\mathbf{E}$ and $\mathbf{F}$. Their control points are computed by using (5) with $\tau = \frac{1}{2}$ for simplicity.

$$\begin{cases} \mathbf{e}_0 = \mathbf{p}_0 \\ \mathbf{e}_1 = \frac{1}{2}\mathbf{p}_0 + \frac{1}{2}\mathbf{q}_1 \\ \mathbf{e}_2 = \frac{1}{4}\mathbf{p}_0 + \frac{1}{2}\mathbf{q}_1 + \frac{1}{4}\mathbf{q}_2 \\ \mathbf{e}_3 = \frac{1}{8}\mathbf{p}_0 + \frac{3}{8}\mathbf{q}_1 + \frac{3}{8}\mathbf{q}_2 + \frac{1}{8}\mathbf{q}_3 \end{cases} \quad \begin{cases} \mathbf{f}_0 = \frac{1}{8}\mathbf{p}_0 + \frac{3}{8}\mathbf{q}_1 + \frac{3}{8}\mathbf{q}_2 + \frac{1}{8}\mathbf{q}_3 \\ \mathbf{f}_1 = \frac{1}{4}\mathbf{q}_1 + \frac{1}{2}\mathbf{q}_2 + \frac{1}{4}\mathbf{q}_3 \\ \mathbf{f}_2 = \frac{1}{2}\mathbf{q}_2 + \frac{1}{2}\mathbf{q}_3 \\ \mathbf{f}_3 = \mathbf{q}_3 \end{cases} \tag{7}$$

Then the subdivided curves $\mathbf{E}$ and $\mathbf{F}$ are approximated by quadratic curves $\mathbf{G}$ and $\mathbf{H}$ respectively. End points of $\mathbf{G}$ and $\mathbf{H}$ are set equal to ones of $\mathbf{E}$ and $\mathbf{F}$:

$$\mathbf{g}_0 = \mathbf{e}_0, \;\; \mathbf{g}_2 = \mathbf{e}_3, \;\; \mathbf{h}_0 = \mathbf{f}_0, \;\; \mathbf{h}_2 = \mathbf{f}_3 \tag{8}$$

The second control points of the quadratic curves are obtained by minimizing the position difference from corresponding cubic curves:

$$\begin{aligned} \mathbf{g}_1 &= \arg\min \int_0^1 \|\mathbf{E}(\lambda) - \mathbf{G}(\lambda)\|^2 d\lambda = \frac{1}{4}(-\mathbf{e}_0 + 3\mathbf{e}_1 + 3\mathbf{e}_2 - \mathbf{e}_3) \\ \mathbf{h}_1 &= \arg\min \int_0^1 \|\mathbf{F}(\lambda) - \mathbf{H}(\lambda)\|^2 d\lambda = \frac{1}{4}(-\mathbf{f}_0 + 3\mathbf{f}_1 + 3\mathbf{f}_2 - \mathbf{f}_3) \end{aligned} \tag{9}$$

To determine the optimal location of $\mathbf{q}_1$, $\mathbf{q}_2$, and $\mathbf{q}_3$, $\overline{\mathbf{p}_0\tilde{\mathbf{q}}_1}$ and $\overline{\tilde{\mathbf{q}}_2\tilde{\mathbf{q}}_3}$ are discretized into a finite number of equally spaced points. So they are represented as

$$\begin{aligned} &\bar{\mathbf{q}}_1(i) = \mathbf{p}_0 + \frac{i}{M-1}(\tilde{\mathbf{q}}_1 - \mathbf{p}_0), \quad i = 1,\ldots,M-1 \\ &\left.\begin{aligned} \bar{\mathbf{q}}_2(j) &= \tilde{\mathbf{q}}_2 + \tfrac{j}{M-1}(\tilde{\mathbf{q}}_3 - \tilde{\mathbf{q}}_2) \\ \bar{\mathbf{q}}_3(k) &= \tilde{\mathbf{q}}_2 + \tfrac{k}{M-1}(\tilde{\mathbf{q}}_3 - \tilde{\mathbf{q}}_2) \end{aligned}\right\}, \quad (j,k) \in \{(j,k) \mid j = 0,\ldots,M-1,\ k = 1,\ldots,M-1,\ k > j\} \end{aligned} \tag{10}$$

where $\bar{\mathbf{q}}_a(b)$ denote the b-th sample point of $\mathbf{q}_a$ for $a = 1,2,3$. M is the total number of sample points on each line segment. For any combination of (i,j,k), corresponding control points of $\mathbf{G}$ and $\mathbf{H}$ are represented by incorporating Eq. (7), (8) and (9) with substitution of $\bar{\mathbf{q}}_1(i)$, $\bar{\mathbf{q}}_2(j)$, and $\bar{\mathbf{q}}_3(k)$ for $\mathbf{q}_1$, $\mathbf{q}_2$, and $\mathbf{q}_3$.

$$\begin{aligned} \mathbf{g}_0(i,j,k) &= \mathbf{p}_0 \\ \mathbf{g}_1(i,j,k) &= \frac{1}{32}\left[(30 - \frac{21i}{M-1})\mathbf{p}_0 + \frac{21i}{M-1}\tilde{\mathbf{q}}_1 + (2 - \frac{3j-k}{M-1})\tilde{\mathbf{q}}_2 + \frac{3j-k}{M-1}\tilde{\mathbf{q}}_3\right] \\ \mathbf{g}_2(i,j,k) &= \frac{1}{8}\left[(4 - \frac{3i}{M-1})\mathbf{p}_0 + \frac{3i}{M-1}\tilde{\mathbf{q}}_1 + (4 - \frac{3j+k}{M-1})\tilde{\mathbf{q}}_2 + \frac{3j+k}{M-1}\tilde{\mathbf{q}}_3\right] \\ \mathbf{h}_0(i,j,k) &= \frac{1}{8}\left[(4 - \frac{3i}{M-1})\mathbf{p}_0 + \frac{3i}{M-1}\tilde{\mathbf{q}}_1 + (4 - \frac{3j+k}{M-1})\tilde{\mathbf{q}}_2 + \frac{3j+k}{M-1}\tilde{\mathbf{q}}_3\right] \\ \mathbf{h}_1(i,j,k) &= \frac{1}{32}\left[(2 - \frac{3i}{M-1})\mathbf{p}_0 + \frac{3i}{M-1}\tilde{\mathbf{q}}_1 + (30 - \frac{21j+9k}{M-1})\tilde{\mathbf{q}}_2 + \frac{21j+9k}{M-1}\tilde{\mathbf{q}}_3\right] \\ \mathbf{h}_2(i,j,k) &= \tilde{\mathbf{q}}_3 \end{aligned} \tag{11}$$

Then $|\kappa|_{max}(\mathbf{g}_0,\mathbf{g}_1,\mathbf{g}_2)$ and $|\kappa|_{max}(\mathbf{h}_0,\mathbf{h}_1,\mathbf{h}_2)$, the maximum curvatures of $\mathbf{G}$ and $\mathbf{H}$ determined by the control points, are calculated by using Eq. (12) which is derived in Appendix A.

$$|\kappa|_{max}(\mathbf{q}_0,\mathbf{q}_1,\mathbf{q}_2) = \begin{cases} \frac{|x_0(y_1-y_2)+x_1(y_2-y_0)+x_2(y_0-y_1)|}{2[(x_0-x_1)^2+(y_0-y_1)^2]^{\frac{3}{2}}}, & \text{if } (x_0-x_1)(x_0-2x_1+x_2)+(y_0-y_1)(y_0-2y_1+y_2) \le 0 \\ \frac{|x_0(y_1-y_2)+x_1(y_2-y_0)+x_2(y_0-y_1)|}{2[(x_1-x_2)^2+(y_1-y_2)^2]^{\frac{3}{2}}}, & \text{else if } (x_1-x_2)(x_0-2x_1+x_2)+(y_1-y_2)(y_0-2y_1+y_2) \ge 0 \\ \frac{[(x_0-2x_1+x_2)^2+(y_0-2y_1+y_2)^2]^{\frac{3}{2}}}{2|x_0(y_1-y_2)+x_1(y_2-y_0)+x_2(y_0-y_1)|^2}, & \text{else} \end{cases} \tag{12}$$

where $\mathbf{q}_0 = (x_0,y_0)$, $\mathbf{q}_1 = (x_1,y_1)$, and $\mathbf{q}_2 = (x_2,y_2)$. Finally, the optimal control points $\bar{\mathbf{q}}_1(i^*)$, $\bar{\mathbf{q}}_2(j^*)$, and $\bar{\mathbf{q}}_3(k^*)$ are determined by the combination of indices (i^*,j^*,k^*) that minimize sum of them.

$$(i^*,j^*,k^*) = \arg\min_{i,j,k}\left[|\kappa|_{max}\big(\mathbf{g}_0(i,j,k),\mathbf{g}_1(i,j,k),\mathbf{g}_2(i,j,k)\big) + |\kappa|_{max}\big(\mathbf{h}_0(i,j,k),\mathbf{h}_1(i,j,k),\mathbf{h}_2(i,j,k)\big)\right] \tag{13}$$

3. *Case III*

This case is similar to *Case II* as shown in figure 3. The boundary point $\tilde{\mathbf{q}}_1$ is defined as the same way as *Case II*. $\tilde{\mathbf{q}}_3$, on $\overline{\mathbf{p}_1\mathbf{p}_2}$, is defined as the point such that $\triangle\tilde{\mathbf{q}}_1\mathbf{p}_1\tilde{\mathbf{q}}_3$ is the biggest triangle that misses any obstacles. If $\tilde{\mathbf{q}}_3$ is $\mathbf{p}_2$ then $\mathbf{Q}$ takes in $\{\mathbf{p}_0, \tilde{\mathbf{q}}_1, \mathbf{p}_1, \mathbf{p}_2\}$ as its control points and extends by the method of section C. Otherwise, $\mathbf{Q}$ is the cubic Bézier curve constructed by $\{\mathbf{p}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3\}$.

$\mathbf{q}_1$ is selected among finite number of sample points on $\overline{\mathbf{p}_0\tilde{\mathbf{q}}_1}$, and so are $\mathbf{q}_2$ and $\mathbf{q}_3$ on $\overline{\mathbf{p}_1\tilde{\mathbf{q}}_3}$. It is important to note the following lemma.

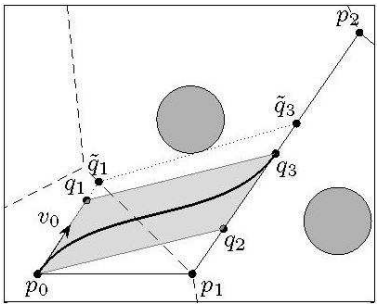


Figure 3. An example of *Case III*.

Lemma 1. *The polygon* $\mathbf{p}_0\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3\mathbf{p}_1$ *is a convex.*

Proof. See Appendix C. □

So any line segment connecting two of $\mathbf{p}_0, \mathbf{q}_1, \mathbf{q}_2$, and $\mathbf{q}_3$ always lies inside of the polygon $\mathbf{p}_0\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3\mathbf{p}_1$ which misses any obstacles. So does the convex hull of $\mathbf{p}_0\mathbf{q}_1\mathbf{q}_2\mathbf{q}_3$. Therefore, the Bézier curve constructed by $\{\mathbf{p}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3\}$ misses any obstacles by the convex hull property.

The optimal control points are obtained by the method presented in *Case II*: curve approximation by two quadratic curves and minimizing the sum of maximum curvatures of the two curves.

4. *Case IV*

Figure 4 shows an example of *case II*. In this case, $\mathbf{q}_1$ is located at the intersection of $\mathbf{p}_0 + \alpha\frac{\mathbf{v}_0}{\|\mathbf{v}_0\|}$ and $\overline{\mathbf{p}_1\mathbf{p}_2}$. Then another control point $\mathbf{q}_2$ is selected on $\overline{\mathbf{q}_1\mathbf{p}_2}$ as $\triangle\mathbf{p}_0\mathbf{q}_1\mathbf{q}_2$ misses any obstacles. It is sufficient that $\triangle\mathbf{p}_0\mathbf{q}_1\mathbf{q}_2$ misses all obstacles whose cell has $\mathbf{p}_1$ as its vertex for the constraint. Let $\tilde{\mathbf{q}}_2$ denote the farthest point from $\mathbf{q}_1$ to satisfy the constraint. If $\tilde{\mathbf{q}}_2$ is $\mathbf{p}_2$, then $\mathbf{Q}$ takes in $\{\mathbf{p}_0, \mathbf{q}_1, \mathbf{q}_2\}$ and extends by adding more on the rest of *SP* as described in section C. Otherwise, $\mathbf{Q}$ is the quadratic Bézier curve constructed by $\{\mathbf{p}_0, \mathbf{q}_1, \mathbf{q}_2\}$. Since $\mathbf{q}_2$ is on $\overline{\mathbf{q}_1\tilde{\mathbf{q}}_2}$, it is determined by a scaler $\beta > 0$ as

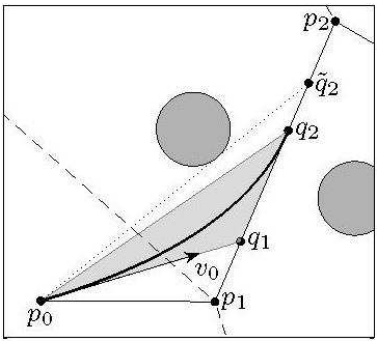


Figure 4. An example of *Case IV*.

$$\mathbf{q}_2 = \mathbf{q}_1 + \beta\frac{\tilde{\mathbf{q}}_2 - \mathbf{q}_1}{\|\tilde{\mathbf{q}}_2 - \mathbf{q}_1\|}, \quad \beta \in (0, \tilde{\beta}] \tag{14}$$

where $\tilde{\beta} = \|\tilde{\mathbf{q}}_2 - \mathbf{q}_1\|$. We are to find $\mathbf{q}_2^*$, the optimal $\mathbf{q}_2$ to minimize $|\kappa|_{max}$ of $\mathbf{Q}$ determined by $\{\mathbf{p}_0, \mathbf{q}_1, \mathbf{q}_2\}$. In other words, we compute β^*, the β corresponding to the $\mathbf{q}_2^*$. In Appendix B, theorem 2, we proved that

$$\beta^* = \arg\min_{\beta\in(0,\tilde{\beta}]} |\kappa|_{max} = \min\Big(\tilde{\beta}, \frac{-\cos\theta + \sqrt{\cos^2\theta + 8}}{2}\alpha\Big)$$

where $\theta \in (0, \pi)$ is the heading difference from $\mathbf{q}_1 - \mathbf{q}_0$ to $\mathbf{q}_2 - \mathbf{q}_1$ and α is $\|\mathbf{q}_1 - \mathbf{q}_0\|$.

C. Control Points on the Shortest Path

Next step is to compute control points of each Bézier curve on the rest of *SP* segments, $\mathbf{p}_2, \mathbf{p}_3, \ldots, \mathbf{p}_n$. The current Bézier curve $\mathbf{Q}_k$, $k = 0, 1, \ldots$ takes in $\mathbf{p}_i$, $i = 2, 3, \ldots$ as its control points, in order, if the circumferential convex polygon of its control points and $\mathbf{p}_i$ misses any obstacles. Once the convex polygon does not, $\mathbf{Q}_k$ takes in the last control point $\mathbf{q}$ on segments before $\mathbf{p}_i$ such that the convex polygon of its control points and $\mathbf{q}$ is free from obstacles. Then another Bézier curve is constructed and extended, and so on.

This method is summarized as pseudo code in Algorithm 1. To present the method, we need to introduce some notions. $C(\mathbf{Q}_k) = \{\mathbf{q}_0(k), \ldots, \mathbf{q}_{l_k}(k)\}$ is the set of control points of a Bézier curve $\mathbf{Q}_k$. $H_{convex}(\mathbf{P})$ is a convex hull of a set of points $\mathbf{P}$.

Algorithm 1 Calculate control points on *SP*.

Require: $\mathbf{Q}_0$ {The result of Section B}
Ensure: $\mathbf{Q}_1, \ldots, \mathbf{Q}_m$
  if $\mathbf{q}_{l_0}(0) \neq \mathbf{p}_2$ **then**
    $C(\mathbf{Q}_1) \Leftarrow \{\mathbf{q}_{l_0}(0), \mathbf{p}_2\}$
    $k \Leftarrow 1$,
  else
    $k \Leftarrow 0$
  end if
  $i \Leftarrow 3$
  while $\mathbf{p}_i \neq \mathbf{p}_n$ **do**
    if $H_{convex}(C(\mathbf{Q}_k) \cup \{\mathbf{p}_i\})$ intersects any obstacles **then**
      if $l_k > 1$ **then**
        Calculate $\mathbf{q}$ on between $\overline{\mathbf{p}_{i-2}\mathbf{p}_{i-1}}$ such that $H_{convex}(C(\mathbf{Q}_k) \cup \{\mathbf{q}\})$ misses any obstacles.
        $C(\mathbf{Q}_k) \Leftarrow \{\mathbf{q}_0(k), \ldots, \mathbf{q}_{l_k-1}(k), \mathbf{q}\}$
        $C(\mathbf{Q}_{k+1}) \Leftarrow \{\mathbf{q}, \mathbf{p}_{i-1}, \mathbf{p}_i\}$
      else
        Calculate $\mathbf{q}$ on between $\overline{\mathbf{p}_{i-1}\mathbf{p}_i}$ such that $H_{convex}(C(\mathbf{Q}_k) \cup \{\mathbf{q}\})$ misses any obstacles.
        $C(\mathbf{Q}_k) \Leftarrow C(\mathbf{Q}_k) \cup \{\mathbf{q}\}$
        $C(\mathbf{Q}_{k+1}) \Leftarrow \{\mathbf{q}, \mathbf{p}_i\}$
      end if
      $k \Leftarrow k+1$
    else
      $C(\mathbf{Q}_k) \Leftarrow C(\mathbf{Q}_k) \cup \{\mathbf{p}_i\}$
    end if
    $i \Leftarrow i+1$
  end while

In the procedure to detect an intersection between convex polygon and obstacles and to compute $\mathbf{q}$, its Voronoi diagram is useful to reduce the number of obstacles to be tested. First of all, suppose that the number of control points of $\mathbf{Q}_k$ is two: $\mathbf{q}_0(k)$ and $\mathbf{p}_{i-1}$ as shown in figure 5. (For simplicity, let us drop the index k.) In this case, $H_{convex}(C(\mathbf{Q}_k) \cup \{\mathbf{p}_i\})$ is $\triangle \mathbf{q}_0\mathbf{p}_{i-1}\mathbf{p}_i$. Since $\overline{\mathbf{q}_0\mathbf{p}_{i-1}}$ and $\overline{\mathbf{p}_{i-1}\mathbf{p}_i}$ are segments of *SP*, they do not intersect any obstacles. Thus, for $\triangle \mathbf{q}_0\mathbf{p}_{i-1}\mathbf{p}_i$ to miss any obstacles, $\overline{\mathbf{q}_0\mathbf{p}_i}$ should miss those. Let $o(\mathbf{p}_{i-1})$ denote the obstacle such that $\mathscr{V}(o(\mathbf{p}_{i-1}))$ is bounded by $\overline{\mathbf{p}_{i-2}\mathbf{p}_{i-1}}$ and $\overline{\mathbf{p}_{i-1}\mathbf{p}_i}$. Note that $\overline{\mathbf{q}_0\mathbf{p}_i}$ lies inside of $\mathscr{V}(o(\mathbf{p}_{i-1}))$ by Proposition 1. Thus, it is sufficient to test if $\overline{\mathbf{q}_0\mathbf{p}_i}$ intersects $o(\mathbf{p}_{i-1})$ or $\triangle \mathbf{q}_0\mathbf{p}_{i-1}\mathbf{p}_i$ contains $o(\mathbf{p}_{i-1})$, in order to detect collision between $H_{convex}(C(\mathbf{Q}_k) \cup \{\mathbf{p}_i\})$ and obstacles. If the collision happens, $\mathbf{q}$ can be chosen such that $\overline{\mathbf{q}_0\mathbf{q}}$ is the tangent of $o(\mathbf{p}_{i-1})$ closer to $\mathbf{p}_{i-1}$. This joint node will be adjusted in Section D.

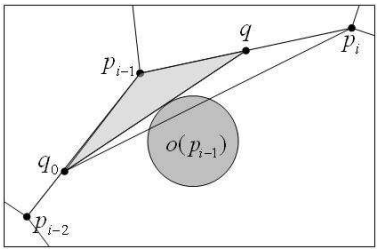
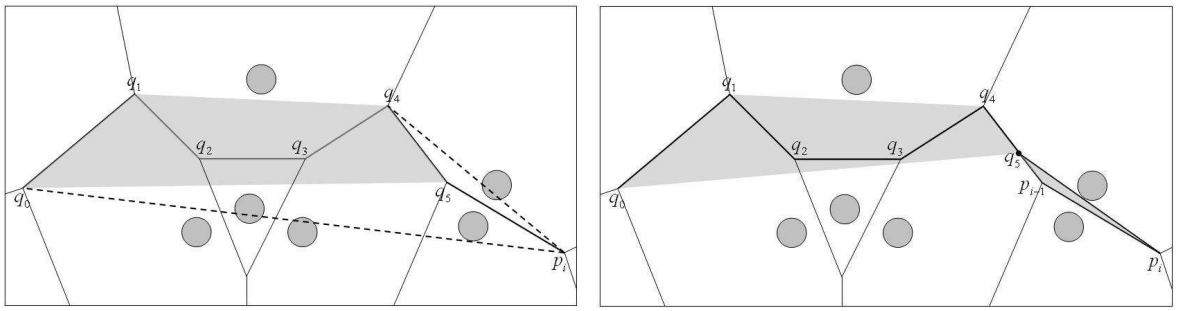


Figure 5. $\mathbf{Q}_k$ has two control points: $\mathbf{q}_0$ and $\mathbf{p}_{i-1}$.

The other case is that the number of control points of $\mathbf{Q}_k$ is more than two. In Figure 6(a), $\mathbf{Q}_k$ has convex hull of control points, which miss any obstacles until $\mathbf{p}_i$ was added. Note that $\mathbf{p}_{i-1}$ is the last control point of $\mathbf{Q}_k$: $\mathbf{q}_5$ in this example. However, adding $\mathbf{p}_i$ to $\mathbf{Q}_k$ causes collision between obstacles and the convex hull. The collision detection is tested over several obstacles. Let $\mathbf{q}_j$ and $\mathbf{q}_h$ be two control points of $H_{convex}(\{\mathbf{q}_0, \ldots, \mathbf{q}_{l_k}, \mathbf{p}_i\})$, that connect to $\mathbf{p}_i$. It is sufficient to test if $\overline{\mathbf{q}_j\mathbf{p}_i}$ intersect or contain any of $o(\mathbf{q}_{j+1}), \ldots, o(\mathbf{q}_{l_k})$ or $\overline{\mathbf{q}_h\mathbf{p}_i}$ does any of $o(\mathbf{q}_{h+1}), \ldots, o(\mathbf{q}_{l_k})$ for collision detection. In case a collision happens, the last control point of $\mathbf{Q}_k$, $\mathbf{q}_{l_k}$ will be re-chosen on $\overline{\mathbf{p}_{i-2}\mathbf{p}_{i-1}}$ such that $\overline{\mathbf{q}_{l_k}\mathbf{p}_i}$ misses the obstacle $o(\mathbf{p}_{i-1})$, as shown in Figure 6(b). $\triangle \mathbf{q}_{l_k}\mathbf{p}_{i-1}\mathbf{p}_i$ is free from collisions by the same reason as the above example. For every new point of $\mathbf{q}_{l_k}$ on $\overline{\mathbf{p}_{i-2}\mathbf{p}_{i-1}}$, the convex hull $H_{convex}(\{\mathbf{q}_0, \ldots, \mathbf{q}_{l_k}\})$ is free from collision, since it lies inside of $H_{convex}(\{\mathbf{q}_0, \ldots, \mathbf{q}_{l_{k-1}}, \mathbf{p}_{i-1}\})$. Then $\mathbf{Q}_{k+1}$ is created by three control points, $\mathbf{q}_{l_k}$, $\mathbf{p}_{i-1}$, $\mathbf{p}_i$ and is extended if possible.



(a) Adding $\mathbf{p}_i$ to $\mathbf{Q}_k$ causes collision.

(b) The last control point of $\mathbf{Q}_k$ is re-chosen.

Figure 6. $\mathbf{Q}_k$ has more than two control points.

D. Adjusting Joint Nodes

Every pair of neighboring Bézier curves created in Section C has continuous tangents at their junction node, since the node is located on a line segment of *SP*. To connect them more smoothly, let us adjust the nodes such that each pair of curves are C_1 continuous at the junction if possible.

Suppose that Section C has ended up with that $\mathbf{Q}_{k-1}$ and $\mathbf{Q}_k$ share a junction node $\mathbf{q}$ on $\overline{\mathbf{p}_{i-1}\mathbf{p}_i}$. Let $\gamma \in (0,1)$ is the ratio such that

$$\overline{\mathbf{p}_{i-1}\mathbf{q}} : \overline{\mathbf{q}\mathbf{p}_i} = \gamma : 1-\gamma \tag{15}$$

We are to find, if possible, the new joint node $\mathbf{q}_{new}$ such that $\mathbf{Q}_{k-1}$ and $\mathbf{Q}_k$ are C_1 continuous at the node. $\mathbf{q}_{new}$ is represented by using Eq. (2) and (3):

$$l_{k-1}(\mathbf{q}_{new} - \mathbf{p}_{i-1}) = l_k(\mathbf{p}_i - \mathbf{q}_{new}) \tag{16}$$

Incorporating the definition of γ in Eq. (15) and the constraint of $\mathbf{q}_{new}$ in Eq. (16), the new ratio γ_{new} corresponding to $\mathbf{q}_{new}$ is represented as

$$\gamma_{new} = \frac{l_k}{l_{k-1}+l_k} \tag{17}$$

Notice that, in Section C, $\mathbf{q}$ is chosen to minimize γ if $l_{k-1} > 2$ and to maximize otherwise. Considering the range of γ with Eq. (17), possible γ_{new} is given by

$$\gamma_{new} = \begin{cases} \max(\frac{l_k}{l_{k-1}+l_k}, \gamma), & \text{if } l_{k-1} > 2 \\ \min(\frac{l_k}{l_{k-1}+l_k}, \gamma), & \text{if } l_{k-1} = 2 \end{cases} \tag{18}$$

Finally, $\mathbf{q}_{new}$ is obtained by using γ_{new} calculated in Eq. (18):

$$\mathbf{q}_{new} = (1-\gamma_{new})\mathbf{p}_{i-1} + \gamma_{new}\mathbf{p}_i \tag{19}$$

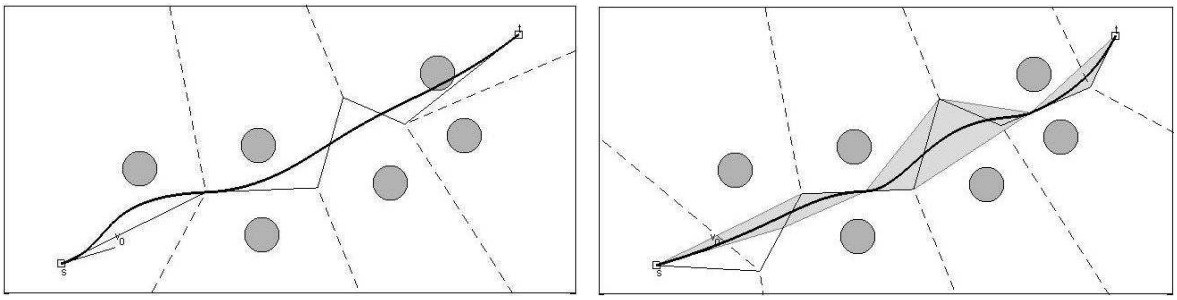
IV. Numerical Simulations

Simulations provided in this section are implemented in Matlab programming language and tested in Windows environment using a Pentium IV 1800 MHz processor.

The first simulation demonstrates obstacle avoiding trajectory generation by the proposed algorithm as opposed to the colliding trajectory generated by Sahraei's algorithm. Figure 7(a) shows that a path planned by Sahraei's algorithm intersects an obstacle. The path planned by the proposed algorithm is collision free by the convex hull property as shown in Figure 7(b).

The second simulation is to demonstrate applicability of the proposed algorithm for obstacle avoiding path planning in an unknown environment. In the real world, many path planning problems are for an unknown environment due to the limitation of the sensors. Let us consider the control problem of an autonomous ground vehicle that has sensors capable of detecting obstacles in a limited range. In the simulation, the reference path is generated against obstacles detected in a receding horizon fashion.

For the dynamics of the vehicle, the state and the control vector are denoted $\mathbf{X}(t) = \big(x(t), y(t), \psi(t)\big)^T$ and $\mathbf{u}(t) = \big(v(t), \omega(t)\big)^T$ respectively, where (x,y) represents the position of the center of gravity of the vehicle. The vehicle yaw



(a) The path by Sahraei's algorithm intersects an obstacle.

(b) The path by the proposed algorithm. The transparent grey areas are obstacle free convex hulls.

Figure 7. The dashed lines are the Voronoi diagram and the solid lines are *SP*. The bold solid lines are the resulting paths by two algorithms.

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 rate. We assume the discrete time system to follow the state equation below.

$$\begin{aligned} x\big((k+1)T\big) &= x\big(kT\big) + Tv\big(kT\big)\cos\psi\big(kT\big) \\ y\big((k+1)T\big) &= y\big(kT\big) + Tv\big(kT\big)\sin\psi\big(kT\big) \qquad ,k = 0,1,\ldots \\ \psi\big((k+1)T\big) &= \psi\big(kT\big) + T\omega\big(kT\big) \end{aligned}$$

In the simulation, we use 0.05 for the sample interval T. The vehicle uses path following with feedback corrections.[9] A position and orientation error is computed every T second. A point $\mathbf{z}$ is computed one sample ahead with the current longitudinal velocity and heading of the vehicle from the current position. $\mathbf{z}$ is projected onto the reference trajectory at point $\mathbf{p}$ such that $\overline{\mathbf{zp}}$ is normal to the tangent at $\mathbf{p}$. The cross track error y_{err} is defined by the distance between $\mathbf{z}$ and $\mathbf{p}$. The steering control ω uses a PID controller with respect to cross track error y_{err}.

$$\omega\big((k+1)T\big) = \omega(kT) + k_p y_{err}(kT) + k_d \frac{dy_{err}(kT)}{dt} + k_i \int_0^{kT} y_{err}(kT)dt \tag{20}$$

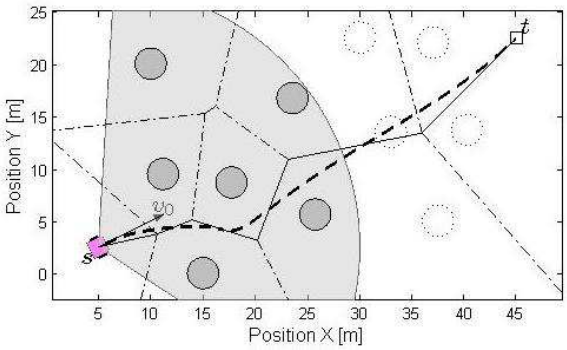
Figure 8 shows the snapshots of the path planning and following process in the unknown environment consisting of 11 obstacles in meter scale. In the simulation, $\mathbf{s} = (5, 2.5)$, $\mathbf{t} = (45, 22.5)$, and $v_0 = 10m/s$ are given. We assume that sensor detecting range is defined as circular sector with radius of 20 m and a central angle of 120° such that the heading of the vehicle is at the bisector of the angle. It is illustrated as the light shaded region in the figure. An obstacle is assumed to be detected if the center point of it lies inside of the range. The magnitude of ω is bounded within 25 rpm or 2.618 rad/s. The PID gains are given by: $k_p = 2$, $k_d = 1$, and $k_i = 0.1$.

In the initial position $\mathbf{s}$, the vehicle sensor detects obstacles in the environment (Figure 8(a)). The reference path is planned for the detected obstacles by applying the proposed algorithm. The path satisfies the initial heading and obstacle avoidance in the Voronoi diagram constructed by $\mathbf{s}$, $\mathbf{t}$ and detected obstacles. When undetected obstacles are detected as the vehicle tracks the reference path, the vehicle calculates if the path intersects the obstacles. If so, the path is replanned by applying the algorithm, again (Figure 8(b)). In the replanned process, $\mathbf{s}$ and $\mathbf{v}_0$ are replaced with the current position and velocity of the vehicle, respectively, and the newly detected obstacle is added to the Voronoi diagram so that the vehicle not only avoids detected obstacles but also keeps tracking the replanned path smoothly (Figure 8(c)). This process is recursively done until the vehicle reaches to $\mathbf{t}$ (Figure 8(f)).

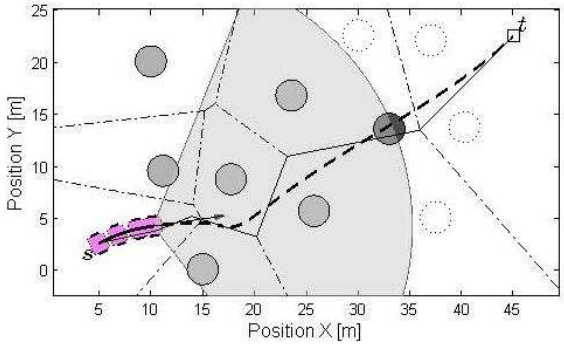
The computational cost of the proposed algorithm is extremely low. In this simulation, the path planning function was called three times. The average time spent in the function of each call was 0.2 **s**.

V. CONCLUSIONS

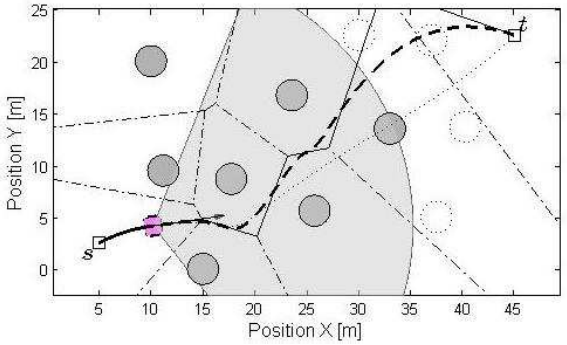
This paper proposes a collision-free real-time path planning algorithm for mobile robots. It has been shown that planned path of the robot is a computationally effective way to satisfy obstacle avoidance as well as short arc length. Numerical simulations demonstrate the improvement of the path planning compared to Sahraei's algorithm and safe path generation in unknown environment.



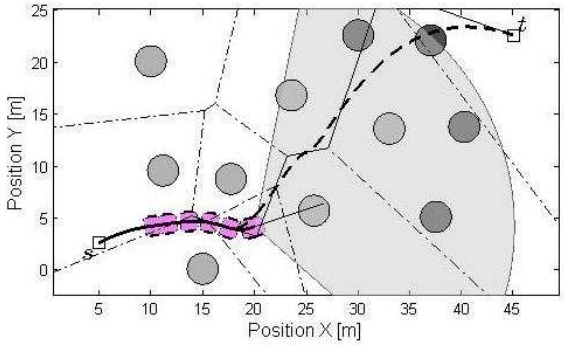
(a) In the initial position **s**, the vehicle sensor detects 6 out of 11 obstacles. The reference path is planned for the detected obstacles by applying the proposed algorithm.



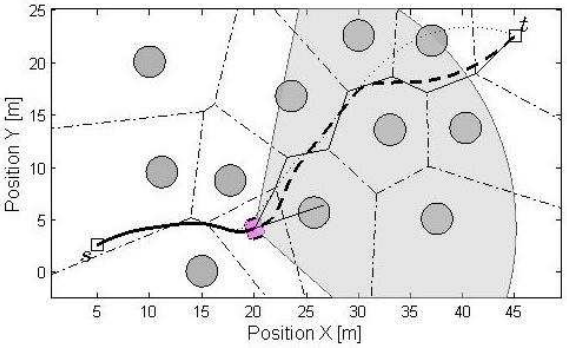
(b) As the vehicle tracks the reference path, a new obstacle (dark shaded circle) is detected. The vehicle decides that the path needs to be replanned, because it intersects the obstacle.



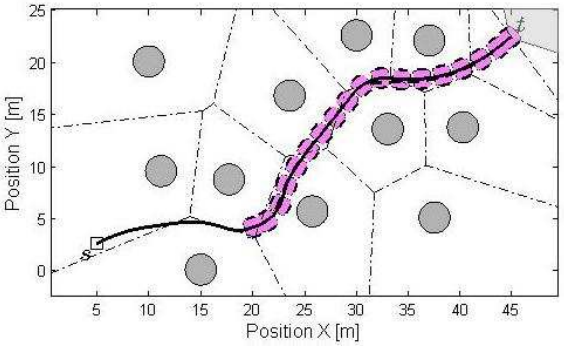
(c) The new path is calculated by substituting the current position and velocity of the vehicle for **s** and $\mathbf{v}_0$, respectively, and adding the newly detected obstacle to the Voronoi diagram.



(d) Again, undetected obstacles are detected. Since the first three obstacles do not intersect the path, the path is not modified.



(e) The path is replanned since the last detected obstacle intersects the path. The replanning is done in the same way in Figure 8(c).



(f) Since the new path does not intersect any obstacle, the vehicle keeps tracking the path until it reaches to **t**.

Figure 8. Snapshots of the path planning in receding horizon fashion in unknown environment.The dash-dot lines are the Voronoi diagram and the solid lines are the shortest paths by Dijkstra's algorithm (*SP*). The bold dashed lines are the reference paths planned by the proposed algorithms. The light shaded circular sector illustrates obstacles detecting range of the vehicle sensor. The undetected obstacles are illustrated as the dotted lines.

Appendix A: Maximum Curvature of a Quadratic Bézier Curve

In 1992, Sapidis and Frey firstly solved the problem of computing the maximum curvature of quadratic Bézier curves.[12] The maximum value is formulated by interpreting geometry of the control points and the hodograph of the curves. We expand their algorithm to generalize the formula in terms of the locations of the control points.

A quadratic Bézier curve $\mathbf{Q}(\lambda) = \big(x(\lambda), y(\lambda)\big)$ is represented as the following by using Eq. (1) with $n = 2$:

$$\mathbf{Q}(\lambda) = (1-\lambda)^2\mathbf{q}_0 + 2\lambda(1-\lambda)\mathbf{q}_1 + \lambda^2\mathbf{q}_2, \quad \lambda \in [0,1] \tag{21}$$

where $\mathbf{q}_0 = (x_0, y_0)$, $\mathbf{q}_1 = (x_1, y_1)$, and $\mathbf{q}_2 = (x_2, y_2)$ are control points. We assume that $\mathbf{q}_0$, $\mathbf{q}_1$ and $\mathbf{q}_2$ are distinct

$$\mathbf{q}_0 \neq \mathbf{q}_1, \;\; \mathbf{q}_1 \neq \mathbf{q}_2, \;\; \mathbf{q}_2 \neq \mathbf{q}_0$$

and not collinear

$$x_0(y_1 - y_2) + x_1(y_2 - y_0) + x_2(y_0 - y_1) \neq 0$$

Plugging the first and the second derivative of Eq. (21) into Eq. (4) yields

$$|\kappa(\lambda)| = \frac{4|x_0(y_1 - y_2) + x_1(y_2 - y_0) + x_2(y_0 - y_1)|}{(a\lambda^2 + b\lambda + c)^{\frac{3}{2}}} \tag{22}$$

where

$$\begin{aligned} a &= 4[(x_0 - 2x_1 + x_2)^2 + (y_0 - 2y_1 + y_2)^2] \\ b &= -8[(x_0 - x_1)(x_0 - 2x_1 + x_2) + (y_0 - y_1)(y_0 - 2y_1 + y_2)] \\ c &= 4[(x_0 - x_1)^2 + (y_0 - y_1)^2] \end{aligned} \tag{23}$$

Note that, in the equation above, $a \geq 0$ and that equality is if and only if $\mathbf{q}_1$ is the midpoint of $\mathbf{q}_0$ and $\mathbf{q}_2$, which contradicts our assumption that $\mathbf{q}_0$, $\mathbf{q}_1$, and $\mathbf{q}_2$ are not collinear. So a is always greater than zero. Also, note that the discriminant of the quadratic polynomial with respect to λ in Eq. (22), $a\lambda^2 + b\lambda + c$ is less than zero:

$$D = b^2 - 4ac = -64[(x_0 - 2x_1 + x_2)^2(y_0 - y_1)^2 + (y_0 - 2y_1 + y_2)^2(x_0 - x_1)^2] < 0$$

So

$$a\lambda^2 + b\lambda + c > 0, \quad \forall\lambda \in [0,1]$$

This conforms the fact that the sign of the curvature of the quadratic Bézier curve is unchanged for all λ. Differentiating Eq. (22) with respect to λ gives

$$\frac{d|\kappa(\lambda)|}{d\lambda} = \frac{-6(2a\lambda + b)|x_0(y_1 - y_2) + x_1(y_2 - y_0) + x_2(y_0 - y_1)|}{(a\lambda^2 + b\lambda + c)^{\frac{5}{2}}} \tag{24}$$

Thus, $|\kappa(\lambda)|$ is monotone if and only if

$$f(\lambda) = 2a\lambda + b \neq 0, \quad \forall\lambda \in (0,1) \tag{25}$$

Since $f(\lambda)$ is the first order polynomial and the first order coefficient $2a$ is greater than zero,

$$\begin{aligned} f(\lambda) > 0,\ \forall\lambda \in (0,1) \quad &\Leftrightarrow\ f(0) = b \geq 0 \\ f(\lambda) < 0,\ \forall\lambda \in (0,1) \quad &\Leftrightarrow\ f(1) = 2a + b \leq 0 \end{aligned}$$

So Eq. (25) can be rewritten as

$$b \geq 0 \ \text{ or } \ 2a + b \leq 0 \tag{26}$$

Incorporating Eq. (26) with (23) yields the necessary and sufficient condition of curvature monotonicity:

$$\begin{aligned} &(x_0 - x_1)(x_0 - 2x_1 + x_2) + (y_0 - y_1)(y_0 - 2y_1 + y_2) \leq 0 \ \text{ or} \\ &(x_1 - x_2)(x_0 - 2x_1 + x_2) + (y_1 - y_2)(y_0 - 2y_1 + y_2) \geq 0 \end{aligned} \tag{27}$$

This can be rewritten as

$$\left(x_1-\frac{3x_0+x_2}{4}\right)^2+\left(y_1-\frac{3y_0+y_2}{4}\right)^2\le\frac{(x_2-x_0)^2+(y_2-y_0)^2}{16}\quad\text{or}$$
$$\left(x_1-\frac{x_0+3x_2}{4}\right)^2+\left(y_1-\frac{y_0+3y_2}{4}\right)^2\le\frac{(x_2-x_0)^2+(y_2-y_0)^2}{16} \tag{28}$$

Eq. (28) parallels with the geometric interpretation that Sapidis and Frey described in the following theorem.

Theorem 1. *Let* $\mathbf{m}$ *be the midpoint of* $\mathbf{q}_0\mathbf{q}_2$*. Then,* $\mathbf{Q}(\lambda)$ *has monotone curvature if and only if one of the angles* $\angle\mathbf{q}_0\mathbf{q}_1\mathbf{m}$ *and* $\angle\mathbf{m}\mathbf{q}_1\mathbf{q}_2$ *is equal to or larger than* $\pi/2$*. In other words,* $\mathbf{Q}(\lambda)$ *has monotone curvature if and only if* $\mathbf{q}_1$ *lies on or inside one of the two circles having as diameter* $\mathbf{q}_0\mathbf{m}$ *and* $\mathbf{m}\mathbf{q}_2$*.*[12]

If $\{\mathbf{q}_0,\mathbf{q}_1,\mathbf{q}_2\}$ meets the first inequality of Eq. (27) then $\frac{d|\kappa(\lambda)|}{d\lambda}<0$, $\forall\lambda\in(0,1)$ and thus $|\kappa|_{max}(\lambda)=|\kappa(0)|$. If $\{\mathbf{q}_0,\mathbf{q}_1,\mathbf{q}_2\}$ meets the second then $\frac{d|\kappa(\lambda)|}{d\lambda}>0$, $\forall\lambda\in(0,1)$ and thus $|\kappa|_{max}(\lambda)=|\kappa(1)|$. On the other hand, if $\{\mathbf{q}_0,\mathbf{q}_1,\mathbf{q}_2\}$ does not meet Eq. (27), then $|\kappa(\lambda)|$ has a unique global maxima at $\hat{\lambda}=-\frac{b}{2a}\in(0,1)$ and thus $|\kappa|_{max}(\lambda)=|\kappa(\hat{\lambda})|$. This is because $\hat{\lambda}$ is the unique solution to $\frac{d|\kappa(\lambda)|}{d\lambda}=0$.

$|\kappa(0)|$, $|\kappa(1)|$, and $|\kappa(\hat{\lambda})|$ are given by substituting 0, 1, and $\hat{\lambda}=-\frac{b}{2a}$ for λ in Eq. (22):

$$\kappa(0)=\frac{x_0(y_1-y_2)+x_1(y_2-y_0)+x_2(y_0-y_1)}{2[(x_0-x_1)^2+(y_0-y_1)^2]^{\frac{3}{2}}} \tag{29}$$
$$\kappa(1)=\frac{x_0(y_1-y_2)+x_1(y_2-y_0)+x_2(y_0-y_1)}{2[(x_1-x_2)^2+(y_1-y_2)^2]^{\frac{3}{2}}} \tag{30}$$
$$\kappa(\hat{\lambda})=\frac{[(x_0-2x_1+x_2)^2+(y_0-2y_1+y_2)^2]^{\frac{3}{2}}}{2[x_0(y_1-y_2)+x_1(y_2-y_0)+x_2(y_0-y_1)]^2} \tag{31}$$

In sum, Eq. (12) summarizes how to compute $|\kappa|_{max}$ of $\mathbf{Q}$ in terms of its control points, $\{\mathbf{q}_0,\mathbf{q}_1,\mathbf{q}_2\}$.

Appendix B: Optimal Control Length of a Quadratic Bézier Curve

In this section, we solve the problem of computing the optimal control length of a quadratic Bézier Curve to minimize its maximum curvature.

To specify the problem, we need to introduce some notation. Suppose a quadratic Bézier curve $\mathbf{Q}$ is constructed by control points $\mathbf{q}_0$, $\mathbf{q}_1$ and $\mathbf{q}_2$. Let $\alpha>0$ and $\beta>0$ be the control lengths between the points, given by

$$\alpha=\|\mathbf{q}_1-\mathbf{q}_0\|,\ \ \beta=\|\mathbf{q}_2-\mathbf{q}_1\|$$

Let $\theta\in(0,\pi)$ denote the heading difference from $\mathbf{q}_1-\mathbf{q}_0$ to $\mathbf{q}_2-\mathbf{q}_1$:

$$\theta=\pi-\angle\mathbf{q}_0\mathbf{q}_1\mathbf{q}_2$$

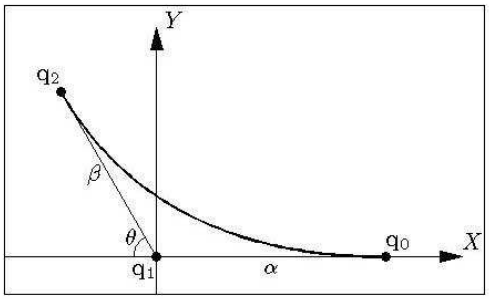


Figure 9. Coordinate transformation.

Without loss of generality, the control points can be translated, rotated, and if necessary, reflected such that $\mathbf{q}_1$ is at the origin, $\mathbf{q}_0$ is on the positive x-axis and $\mathbf{q}_2$ is above the x-axis (See figure 9). The control points may now be written as

$$\mathbf{q}_0=(\alpha,0),\ \mathbf{q}_1=(0,0),\ \mathbf{q}_2=(-\beta\cos\theta,\beta\sin\theta). \tag{32}$$

Substituting these into Eq. (12) yields the maximum curvature expressed in terms of α, β, and θ.

$$|\kappa|_{max}(\alpha,\beta,\theta)=\begin{cases}\frac{\beta\sin\theta}{2\alpha^2}, & \text{if } \alpha\le\beta\cos\theta\\ \frac{\alpha\sin\theta}{2\beta^2}, & \text{else if } \beta\le\alpha\cos\theta\\ \frac{(\beta^2-2\alpha\beta\cos\theta+\alpha^2)^{\frac{3}{2}}}{2\alpha^2\beta^2\sin^2\theta}, & \text{else}\end{cases} \tag{33}$$

We assume that θ and one of the control lengths are given. Without loss of generality, let α be given. The problem to solve is to compute $\beta^* \in (0, \tilde{\beta}]$ that leads to the minimum of $|\kappa|_{max}$ of $\mathbf{Q}$ determined by θ, α, and all $\beta \in (0, \tilde{\beta}]$, where $\tilde{\beta}$ is the maximum boundary value for β.

$$\beta^* = \arg \min_{\beta \in (0,\tilde{\beta}]} |\kappa|_{max}(\alpha, \beta, \theta)$$

To present the theorem to compute β^*, we need to introduce more notation. Let Ω denote the set of β such that $\mathbf{Q}$ determined by θ, α, and β is non-monotone. The set is obtained by substituting Eq. (32) into Eq. (27):

$$\Omega = \{\beta \mid \beta > \alpha\cos\theta \text{ or } \beta\cos\theta < \alpha\} = \begin{cases} \{\beta \mid \beta \in (0,\infty)\}, & \forall\theta \in [\frac{\pi}{2}, \pi) \\ \{\beta \mid \beta \in (\alpha\cos\theta, \frac{\alpha}{\cos\theta})\}, & \forall\theta \in (0, \frac{\pi}{2}) \end{cases} \tag{34}$$

Lemma 2. *Suppose θ and α are given. $|\kappa|_{max}(\beta)$, $\beta \in \Omega$ has a unique global minimum at $\beta = \frac{-\cos\theta + \sqrt{\cos^2\theta + 8}}{2}\alpha$.*

Proof. Incorporating Eq. (34) with Eq. (33) yields

$$|\kappa|_{max}(\beta) = \frac{(\beta^2 - 2\alpha\beta\cos\theta + \alpha^2)^{\frac{3}{2}}}{2\alpha^2\beta^2\sin^2\theta}, \quad \forall\beta \in \Omega \tag{35}$$

Differentiating $|\kappa|_{max}(\beta)$ with respect to β yields

$$\frac{d}{d\beta}|\kappa|_{max}(\beta) = \frac{\sqrt{\beta^2 - 2\alpha\beta\cos\theta + \alpha^2}(\beta^2 + \alpha\beta\cos\theta - 2\alpha^2)}{2\alpha^2\beta^3\sin^2\theta}, \quad \beta \in \Omega$$

Note that the sign of $\frac{d}{d\beta}|\kappa|_{max}(\beta)$ relies on that of $\beta^2 + \alpha\beta\cos\theta - 2\alpha^2$. We denote $f(\beta)$ the quadratic polynomial:

$$f(\beta) = \beta^2 + \alpha\cos\theta\beta - 2\alpha^2, \quad \beta \in \Omega$$

The root of $f(\beta) = 0$ is

$$\beta_{root} = \frac{-\cos\theta + \sqrt{\cos^2\theta + 8}}{2}\alpha \tag{36}$$

The other root $\frac{-\cos\theta - \sqrt{\cos^2\theta + 8}}{2}\alpha < 0$ is infeasible for β. So

$$\frac{d}{d\beta}|\kappa|_{max}(\beta) \begin{cases} < 0, & \forall\beta \in (0, \beta_{root}) \\ > 0, & \forall\beta \in (\beta_{root}, \infty) \end{cases}$$

We need to see if β_{root} is in Ω according to θ.

1. Given $\theta \in [\frac{\pi}{2}, \pi)$: Since $\Omega = (0, \infty)$ from Eq. (34), $\beta_{root} \in \Omega$.
2. Given $\theta \in (0, \frac{\pi}{2})$: Note that $\Omega = (\alpha\cos\theta, \frac{\alpha}{\cos\theta})$ from Eq. (34) and that

$$\cos\theta < \frac{-\cos\theta + \sqrt{\cos^2\theta + 8}}{2} < \frac{1}{\cos\theta}, \tag{37}$$

Multiplying α to (37) yields

$$\alpha\cos\theta < \beta_{root} < \frac{\alpha}{\cos\theta} \;\Rightarrow\; \beta_{root} \in \Omega$$

Therefore, $|\kappa|_{max}(\beta)$, $\beta \in \Omega$ has a unique global minimum at $\beta = \beta_{root}(\theta, \alpha) = \frac{-\cos\theta + \sqrt{\cos^2\theta + 8}}{2}\alpha$. □

Theorem 2. *Suppose θ and α are given. $|\kappa|_{max}(\beta)$, $\beta \in (0, \infty)$ has a unique global minimum at $\beta = \beta_{root}(\theta, \alpha) = \frac{-\cos\theta + \sqrt{\cos^2\theta + 8}}{2}\alpha$. So β^* is given by*

$$\beta^* = \min\left(\tilde{\beta}, \frac{-\cos\theta + \sqrt{\cos^2\theta + 8}}{2}\alpha\right) \tag{38}$$

Proof. The proof is divided into two parts according to θ.

1. Given $\theta \in [\frac{\pi}{2}, \pi)$: Since $\Omega = (0, \infty)$ from Eq. (34), all $\beta \in (0, \infty)$ satisfies $\beta \in \Omega$. So $|\kappa|_{max}(\beta)$, $\beta \in (0, \infty)$ has a unique global minimum at $\beta = \beta_{root}(\theta, \alpha)$ by lemma 2.

2. Given $\theta \in (0, \frac{\pi}{2})$: Let us divide the range of β into three sections: $(0, \alpha\cos\theta]$, $(\alpha\cos\theta, \frac{\alpha}{\cos\theta})$, and $[\frac{\alpha}{\cos\theta}, \infty)$.

   (a) $0 < \beta \leq \alpha\cos\theta$: From Eq. (33), $|\kappa|_{max}(\beta)$ is given by

   $$|\kappa|_{max}(\beta) = \frac{\alpha\sin\theta}{2\beta^2}, \quad \beta \in (0, \alpha\cos\theta]. \tag{39}$$

   So $|\kappa|_{max}(\beta)$ decreases monotonically as β increases.

   (b) $\alpha\cos\theta < \beta < \frac{\alpha}{\cos\theta}$: From Eq. (33), $|\kappa|_{max}(\beta)$ is given by

   $$|\kappa|_{max}(\beta) = \frac{(\beta^2 - 2\alpha\beta\cos\theta + \alpha^2)^{\frac{3}{2}}}{2\alpha^2\beta^2\sin^2\theta}, \quad \beta \in (\alpha\cos\theta, \frac{\alpha}{\cos\theta}), \tag{40}$$

   By lemma 2, $|\kappa|_{max}(\beta)$ has a unique global minimum at $\beta = \beta_{root}(\theta, \alpha)$.

   (c) $\beta \geq \frac{\alpha}{\cos\theta}$: From Eq. (33), $|\kappa|_{max}(\beta)$ is given by

   $$|\kappa|_{max}(\beta) = \frac{\beta\sin\theta}{2\alpha^2}, \quad \beta \in [\frac{\alpha}{\cos\theta}, \infty). \tag{41}$$

   So $|\kappa|_{max}(\beta)$ increases monotonically as β increases.

   All $|\kappa|_{max}(\beta)$ in each section above, (39), (40), and (41) are continuous function of β. Also, $|\kappa|_{max}(\beta)$ is continuous at the junctions, $\alpha\cos\theta$ and $\frac{\alpha}{\cos\theta}$:

$$|\kappa|_{max}(\alpha\cos\theta + 0^-) = |\kappa|_{max}(\alpha\cos\theta) = \frac{\alpha\sin\theta}{2\beta^2}\Big|_{\beta=\alpha\cos\theta} = \frac{\sin\theta}{2\alpha\cos^2\theta}$$
$$= |\kappa|_{max}(\alpha\cos\theta + 0^+) = \frac{(\beta^2 - 2\alpha\beta\cos\theta + \alpha^2)^{\frac{3}{2}}}{2\alpha^2\beta^2\sin^2\theta}\Big|_{\beta=\alpha\cos\theta}$$

$$|\kappa|_{max}(\frac{\alpha}{\cos\theta} + 0^-) = \frac{(\beta^2 - 2\alpha\beta\cos\theta + \alpha^2)^{\frac{3}{2}}}{2\alpha^2\beta^2\sin^2\theta}\Big|_{\beta=\frac{\alpha}{\cos\theta}} = \frac{\sin\theta}{2\alpha\cos\theta}$$
$$= |\kappa|_{max}(\frac{\alpha}{\cos\theta}) = |\kappa|_{max}(\frac{\alpha}{\cos\theta} + 0^+) = \frac{\beta\sin\theta}{2\alpha^2}\Big|_{\beta=\frac{\alpha}{\cos\theta}}$$

   Incorporating the results above yields the shape of $|\kappa|_{max}(\beta)$ as shown in figure 10. So $|\kappa|_{max}(\beta)$ has a unique global minimum at $\beta = \beta_{root}(\theta, \alpha) = \frac{-\cos\theta + \sqrt{\cos^2\theta + 8}}{2}\alpha$.

Therefore, β^* is given by (38). □

Appendix C: Proof of Lemma 1

Note that both of internal angles of $\angle\tilde{\mathbf{q}}_1\mathbf{p}_0\mathbf{p}_1$ and $\angle\mathbf{p}_0\mathbf{p}_1\tilde{\mathbf{q}}_3$ are less than π by definition of case III.

Suppose that internal angle of $\angle\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3\mathbf{p}_1$ is greater than or equal to π as illustrated in 11(a). So $\tilde{\mathbf{q}}_3$ lies inside of $\triangle\mathbf{p}_0\mathbf{p}_1\tilde{\mathbf{q}}_1$. $\overline{\mathbf{p}_1\tilde{\mathbf{q}}_1}$ lies inside of $\mathscr{V}(\mathbf{p}_0)$, since $\mathbf{p}_1$ is a vertex of $\mathscr{V}(\mathbf{p}_0)$ and $\tilde{\mathbf{q}}_1$ is on a edge of $\mathscr{V}(\mathbf{p}_0)$. Thus, subsequently, $\tilde{\mathbf{q}}_3$ lies inside of $\mathscr{V}(\mathbf{p}_0)$. That is, a Voronoi edge $\overline{\mathbf{p}_1\mathbf{p}_2}$ pass through a Voronoi cell $\mathscr{V}(\mathbf{p}_0)$. This is contradictory against the definition of the Voronoi diagram. As the result, internal angle of $\angle\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3\mathbf{p}_1$ is less than π, as well.

Suppose $\angle\mathbf{p}_0\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3$ is greater than π as illustrated in 11(b). Since $\triangle\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3\mathbf{p}_1$ does not contain any obstacle in it, the extension of $\overline{\mathbf{p}_0\tilde{\mathbf{q}}_1}$ intersects $\overline{\mathbf{p}_1\mathbf{p}_2}$ without colliding any obstacle. It is contradictory because this is *Case II*. Therefore, all internal angles are less than π. In other words, $\mathbf{p}_0\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3\mathbf{p}_1$ is a convex.

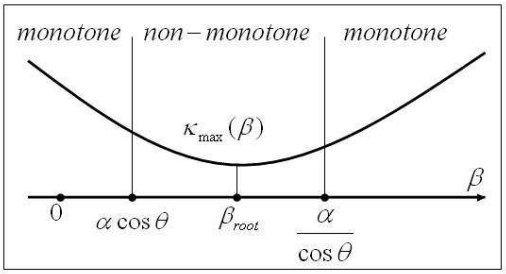
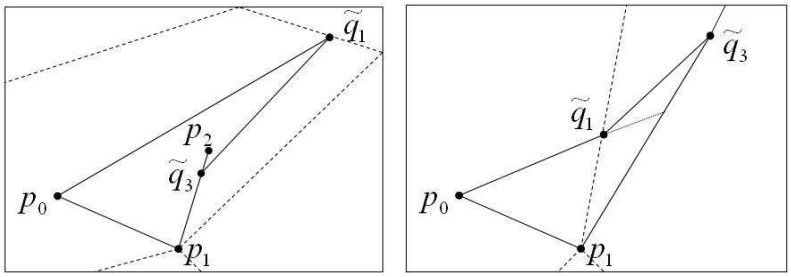


Figure 10. $|\kappa|_{max}(\beta)$ **versus** β**.**



(a) $\angle\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3\mathbf{p}_1 > \pi$ (b) $\angle\mathbf{p}_0\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3 > \pi$

Figure 11. Geometry when an internal angle of $\mathbf{p}_0\tilde{\mathbf{q}}_1\tilde{\mathbf{q}}_3\mathbf{p}_1$ **is greater than** π**.**

References

[1]Latombe, J.-C., *Robot Motion Planning*, Kluwer Academic Publishers, 1991.

[2]Nilsson, N. J., "A Mobile Automation: An Application of Artificial Intelligence Techniques," *The 1st International Joint Conference on Artificial Intelligence*, 1969.

[3]Hsu, D., Latombe, J.-C., and Motwani, R., "Path Planning in Expansive Configuration Spaces," *IEEE International Conference on Robotics and Automation*, 1997.

[4]Lozano-Pérez, T., "Automatic Planning of Manipulator Transfer movements," *IEEE Transactions on Systems*, Vol. 11, No. 10, 1981, pp. 681–698.

[5]Chatila, R., "Path Planning and Environment Learning in a Mobile Robot System," *European Confernece on Artificial Intelligence*, 1982.

[6]Khatib, O., "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *International Journal of Robotics Research*, Vol. 5, No. 1, 1986, pp. 90–98.

[7]Barraquand, J., Langlois, B., and Latombe, J.-C., "Robot Motion Planning with Many Degrees of Freedom and Dynamic Constraints," *Preprints of the Fifth International Symposium of Robotics Research*, 1989.

[8]Miller, I., Lupashin, S., Zych, N., Moran, P., Schimpf, B., Nathan, A., and Garcia, E., *Cornell University's 2005 DARPA Grand Challenge Entry*, Vol. 36, chap. 12, Springer Berlin / Heidelberg, 2007, pp. 363–405.

[9]Choi, J.-W., 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*, San Francisco, CA, USA, 2009, pp. 668–673.

[10]Sahraei, A., Manzuri, M. T., Razvan, M. R., Tajfard, M., and Khoshbakht, S., "Real-Time Trajectory Generation for Mobile Robots," *Proceedings of the 10th Congress of the Italian Association for Artificial Intelligence on AI*IA 2007: Artificial Intelligence and Human-Oriented Computing*, Rome, Italy, 2007.

[11]de Berg, M., van Kreveld, M., Overmars, M., and Schwarzkopf, O., *Computational Geometry: Algorithms and Applications*, Springer, 2000.

[12]Sapidis, N. and Frey, W. H., "Controlling the curvature of a quadratic Bézier curve," *Computer Aided Geometric Design*, Vol. 9, 1992, pp. 85–91.