Robotics: Science and Systems 2023
Daegu, Republic of Korea, July 10-July 14, 2023

G^*: A New Approach to Bounding Curvature Constrained Shortest Paths through Dubins Gates

Satyanarayana G Manyam
Infoscitex Corp.
Dayton, OH 45431
Email: msngupta@gmail.com

Abhishek Nayak
Texas A&M University
College Station, TX 77843
Email: nykabhishek@tamu.edu

Sivakumar Rathinam
Texas A&M University
College Station, TX 77843
Email: srathinam@tamu.edu

***Abstract*—We consider a Curvature-constrained Shortest Path (CSP) problem on a 2D plane for a robot with minimum turning radius constraints in the presence of obstacles. We introduce a new bounding technique called Gate* (G*) that provides optimality guarantees to the CSP. Our approach relies on relaxing the obstacle avoidance constraints but allows a path to travel through some restricted sets of configurations called gates which are informed by the obstacles. We also let the path to be discontinuous when it reaches a gate. This approach allows us to pose the bounding problem as a least-cost problem in a graph where the cost of traveling an edge requires us to solve a new motion planning problem called the Dubins gate problem. In addition to the theoretical results, our numerical tests show that G* can significantly improve the lower bounds with respect to the baseline approaches, and by more than 60% in some instances.**

I. Introduction

Finding a collision-free path for a robot in the midst of obstacles is a fundamental problem in Robotics [1]–[3]. In this paper, we consider a Curvature-constrained Shortest Path (CSP) problem on a 2D plane for a robot with minimum turning radius constraints. Specifically, given an initial and a final configuration[1], the minimum turning radius ($\rho > 0$) of the robot and a set of obstacles in the 2D plane, the objective is to find a shortest, collision-free path from the initial to the final configuration such that the radius of curvature at any point on the path is at least equal to ρ (refer to Fig. 1). This is a central problem that arises in applications for mobile robots controlled by steering mechanisms or for fixed-wing aerial robots with turn-rate constraints. There has been considerable work on the CSP and related problems under the general area of non-holonomic motion planning in the Robotics literature [4]–[13]. Our focus in this paper is on the optimality guarantees for the CSP.

In the *absence* of obstacles, the CSP problem reduces to the classic shortest path problem considered by L.E. Dubins in [14]. Dubins showed that the shortest path between two configurations on a 2D plane belongs to a family of 6 paths where each path is a concatenation of at most 3 pieces, each of which is a straight line or a circular arc [14]. This shortest path problem can also be formulated as an optimal control problem and solved using Pontryagin's minimum principle, as shown in [7].

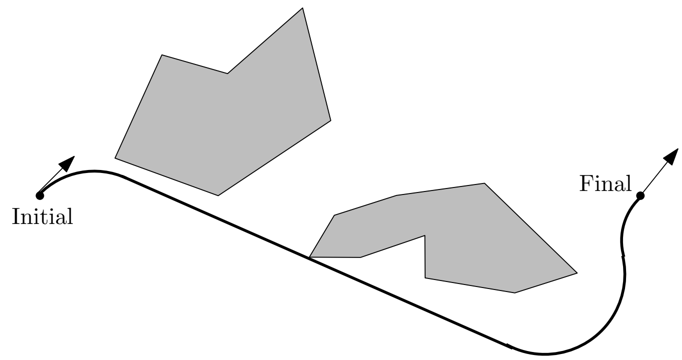


Fig. 1: Illustration of a CSP for an instance with two obstacles.

In the *presence* of obstacles, it is much harder to compute a CSP. In [10], Forture and Wilfong develop an exact algorithm that can decide if the shortest path exists (but do not find such a path) when the obstacles are polygons. Reif and Wang [11] show that finding a CSP is NP-hard when the obstacles are polygons with a total of k vertices and the vertex positions are specified within $O(k^2)$ bits. Apart from the special case addressed in [9] for disjoint, convex obstacles with boundaries consisting of line segments or circular arcs of unit radius, we are not aware of any exact algorithms for finding a CSP. Since finding the optimum is difficult, there are two ways of generating optimality guarantees (*a-priori* and *a-posteriori*) for the CSP. *A-priori* guarantees obtained through approximation algorithms provide theoretical upper bounds on the length of the paths found by the algorithms with respect to the optimum in polynomial time; they are theoretical worst-case bounds, generally true for any instance of the problem. Given length l, and a factor $\epsilon > 0$, an approximation algorithm is presented in [12] which either outputs that no feasible path with length at most equal to l exists or finds such a path whose length is at most $(1+\epsilon)$ times the optimum. This algorithm runs in time polynomially bounded in n (the total number of obstacle vertices and edges), m (the bit precision of the input), $\frac{1}{\epsilon}$, and l. More approximation results are presented in [13] for a scenario with moderate[2] obstacles. Robust variants of the CSP with polygonal obstacles [15], [16], and CSPs inside a polygon [17] have also been addressed. While all the existing approximation

[1] A configuration is defined by a location and orientation (or heading angle) of the robot on a 2D plane.

[2] An obstacle is defined as moderate if it is convex and its boundary is a differential curve whose radius of curvature is everywhere at least equal to 1.

algorithms for a CSP in the presence of polygonal or moderate obstacles provide theoretical guarantees, to the best of our knowledge, we are not aware of any implementations of these algorithms on any test instance.

Given a problem instance, there are also other ways (sampling-based methods [18]–[20], heuristics [21], [22]) for obtaining feasible solutions to the CSP problem. We are then interested in addressing the following question in this paper: ***Given a feasible solution to the CSP problem, how do we know how good the solution is?*** The only way to answer this question is to compare the length of the feasible solution to the optimum. Since we do not know how to find the optimum, we develop algorithms that can find tight lower bounds or underestimates to the optimum, which then provide us with *a-posteriori*[3] guarantees. With respect to the lower bounds, there are two of them that are readily available for the CSP. The first lower bound can be obtained by finding a CSP ignoring the obstacles (referred to as the Dubins lower bound), and the second lower bound can be obtained by finding a shortest, Euclidean path in the presence of obstacles while ignoring the curvature constraints (referred to as the Euclidean lower bound). Other than these two lower bounds, we are not aware of any other lower bound for the CSP problem available in the literature. While these two lower bounds are relatively easy to compute, they may not be tight. For example, in Fig. 2, we show a comparison between these lower bounds and the length of the feasible paths obtained by some of the best sampling-based methods for 30 instances. On average, the deviation of the feasible solutions from these lower bounds is ≈60%, and it gets as worse as ≈80% for some instances. Our main objective in this paper is to improve on these lower bounds (or *a-posteriori* guarantees) and, as a result, provide more accurate estimates of the quality of the feasible solutions for the CSP.

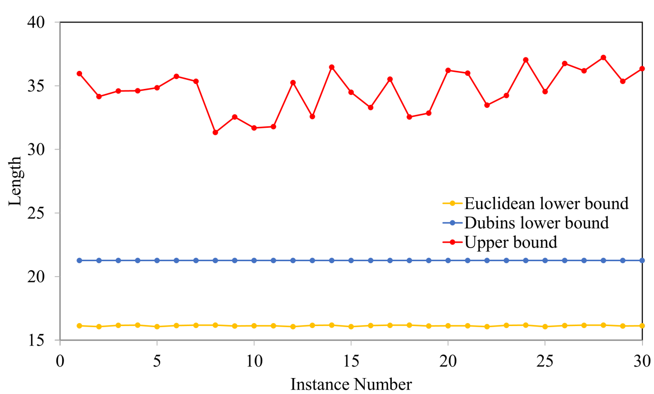


Fig. 2: Comparison between the upper bound (length of the best feasible solution) generated by sampling-based methods (RRT*, BIT*, FMT*) and the two lower bounds for 30 instances. Instances consist of 10-20 obstacles in a 16x9 map. Also, ρ is set to 3.

To obtain lower bounds, we can relax some constraints in the CSP. The choice of which constraint to relax is critical

[3]Measures the deviation of a feasible solution from a lower bound.

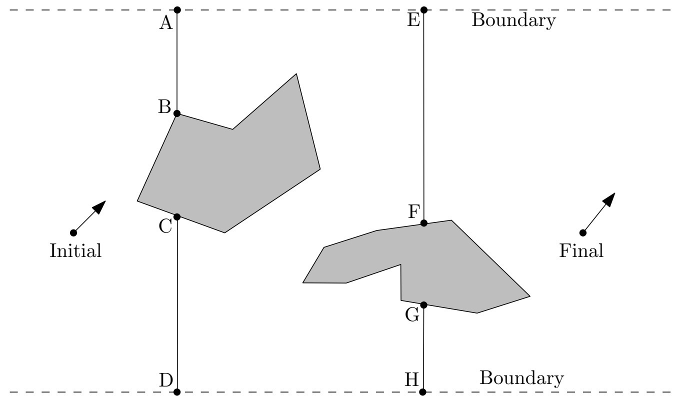


Fig. 3: Illustration of Gates.

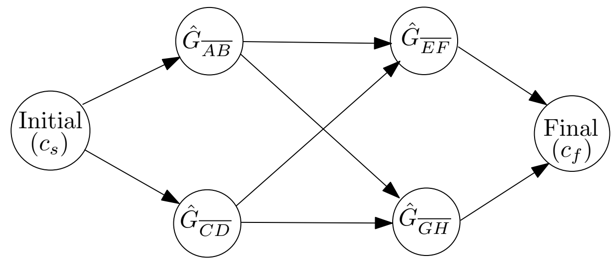


Fig. 4: Graph construction using gates.

because otherwise, we may end up with either poor lower bounds or relaxations that remain challenging to solve. In this paper, we relax the obstacle avoidance constraints but allow a path to travel through some restricted sets of configurations called gates which are informed by the obstacles. We also allow for the paths to be discontinuous when it enters a gate. Similar ideas on relaxing the continuity of the paths have been successfully applied to the Dubins Traveling Salesman Problem and its extensions in [23], [24] to obtain optimality guarantees. To illustrate our ideas, for any line segment $\overline{XY}$, consider $\hat{G}_{\overline{XY}} := \{(x, y, \theta) : (x, y) \in \overline{XY}, \theta \in [-\frac{\pi}{2}, +\frac{\pi}{2}]\}$, a set of configurations referred to as a gate associated with line segment $\overline{XY}$. For an example scenario shown in Fig. 3, it is clear that any feasible path (if it exists) must first pass through $\hat{G}_{\overline{AB}}$ or $\hat{G}_{\overline{CD}}$, and then through either $\hat{G}_{\overline{EF}}$ or $\hat{G}_{\overline{GH}}$.

Allowing a path to be discontinuous when it traverses through a gate enables us to pose the lower bounding problem as a shortest path problem in the following way: The gates and the initial/final configurations are represented as vertices in a newly constructed directed and acyclic graph $\mathcal{G}$ as shown in Fig. 4. The minimum length of the curvature-constrained shortest path between any two adjacent gates or vertices is obtained by formulating and solving a new motion planning problem called the *Dubins Gate* problem. This length is set as the cost of the corresponding edge in $\mathcal{G}$. Once we compute the costs of all the edges in the graph, we solve for a least-cost path from the initial to the final configuration in $\mathcal{G}$, the cost of which is a lower bound to the CSP. The procedure for adding gates to $\mathcal{G}$ is accomplished through an iterative process. Each iteration of our approach adds a new set of gates, and the updated lower bounding solution (least-cost

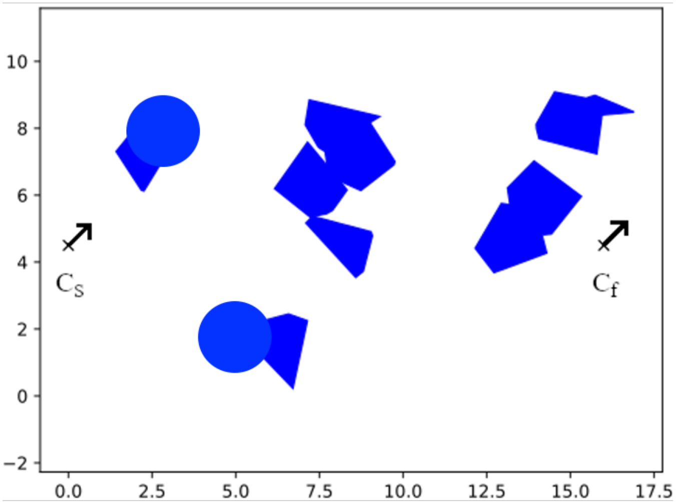


Fig. 5: Obstacle Map.

path) further informs us on the choice of the gates to add in the next iteration. This iterative procedure terminates when we reach the computational time limit or when we cannot add any more gates (based on the parameters we specify). We refer to our approach as Gate* (G*). We also present extensive numerical tests to show that G* can significantly improve the lower bounds with respect to the baseline approaches, and by more than 60% in some instances.

II. Problem Statement

The configuration of the robot at time t is represented as $(x(t), y(t), \theta(t))$ where $(x(t), y(t))$ denotes the position and $\theta(t)$ denotes the heading angle of the robot at time t. Without loss of generality, we assume both the initial and final configurations of the robot lie on the $x-$axis. That is, we let $c_s := (0, 0, \theta_s)$ denote the initial configuration of the robot at time $t = 0$. The final (desired) configuration of the robot is denoted by $c_f := (x_f, 0, \theta_f)$. Also, without loss of generality, we assume the robot travels at unit speed; therefore, the time elapsed is the same as the distance traversed along the path. Let Ω denote a set of obstacles in a 2D plane. We assume each obstacle is a either a convex polygon[4] or a disc but they can intersect allowing for non-convex regions where obstacles are present (Fig. 5). Any path between c_s and c_f is feasible if it does not intersect with the interior of any obstacle and the radius of curvature at any point on the path is at least equal to ρ. The objective of the CSP is to find a shortest, feasible path from c_s to c_f.

III. Preliminaries and Notations

A gate consists of a set of all the configurations (x, y, θ) such that (x, y) lies on a line segment and θ is any angle in a given sector of angles. Specifically, if the line segment connecting two points A and B is denoted as $\overline{AB}$ and the sector of angles is denoted as $[\theta_{min}, \theta_{max}]$, then the corresponding gate

[4]The approach presented in this paper is generic and can be extended to other shapes.

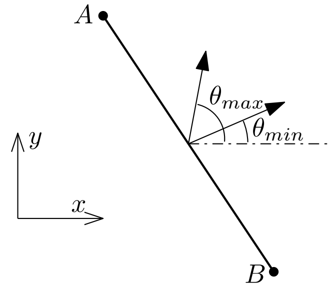


Fig. 6: Illustration of the gate $G_{\overline{AB}}(\theta_{min}, \theta_{max})$ corresponding to line segment $\overline{AB}$. As usual, angles are measured in the counter-clockwise direction with respect to the $x-$axis.

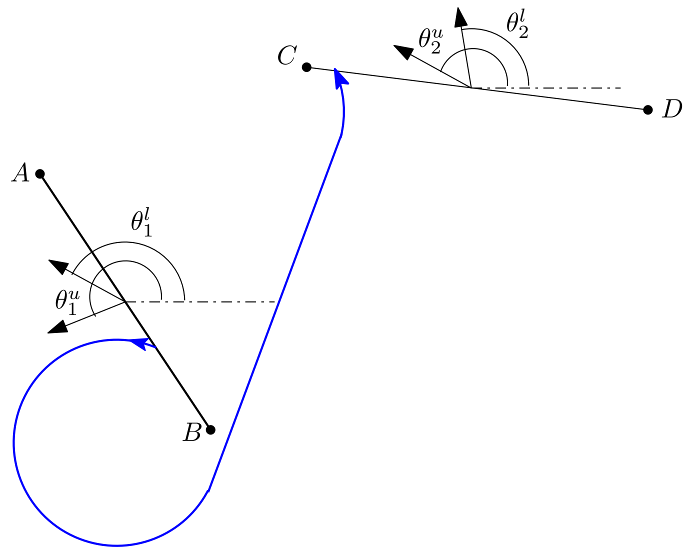


Fig. 7: An example of a feasible path for the Dubins Gate Problem (DGP). Note that the departure and the arrival configurations of Dubins path (in blue color) must satisfy the heading angle constraints.

$G_{\overline{AB}}(\theta_{min}, \theta_{max})$ is defined as $\{(x, y, \theta) : (x, y) \in \overline{AB}, \theta \in [\theta_{min}, \theta_{max}]\}$. Refer to Fig. 6 for an illustration.

The initial and the final configurations of the robot can be also viewed as special cases of gates where the line segments and the sectors reduce to points and angles respectively. To simplify the presentation, we interchangeably refer to any vertex in the graph $\mathcal{G}$ as a gate, and vice-versa. While there are several ways of choosing and adding gates to $\mathcal{G}$, in this paper, we only add gates corresponding to vertical line segments. Also, we only add a gate if the $x-$coordinate of any configuration in the gate lies strictly between 0 and x_f (the initial and final $x-$coordinates of the robot)[5]. This allows us to generate a simpler graph (directed and acyclic) like the one shown in Fig. 4. Other possibilities for generating gates will be considered in future work. Since we only add

[5]This gate construction procedure in G* doesn't impose any restriction on the location of the obstacles; certainly, obstacles can lie anywhere. We used this approach to make the graph construction in G* simpler. Ideally, we would like to allow for all the gates, and also have back edges going from gates with larger x coordinates to gates with smaller x coordinates. If we add all the gates and allow back edges, we certainly expect the bounds to get better.

gates corresponding to vertical line segments, the gates in $\mathcal{G}$ can be partitioned into disjoint subsets $V_i, i = 1 \cdots, l$ ($l \geq 2$) such that the x-coordinate of any configuration in any gate of V_i is the same (lets call this x-coordinate as $\bar{x}(V_i)$), and $\bar{x}(V_1) < \bar{x}(V_2) \leq \bar{x}(V_3) \cdots \leq \bar{x}(V_{l-1}) < \bar{x}(V_l)$. By our gate construction process, note that $V_1 = \{c_s\}$ and $V_l = \{c_f\}$. For example, in Fig. 4, the six gates (or vertices) of $\mathcal{G}$ can be partitioned into $V_1 = \{c_s\}$, $V_2 = \{\hat{G}_{\overline{AB}}, \hat{G}_{\overline{CD}}\}$, $V_3 = \{\hat{G}_{\overline{EF}}, \hat{G}_{\overline{GH}}\}$, $V_4 = \{c_f\}$.

Given two gates $G_{\overline{AB}}(\theta_1^l, \theta_1^u)$ and $G_{\overline{CD}}(\theta_2^l, \theta_2^u)$, the **Dubins Gate Problem (DGP)** aims to find the shortest curvature constrained path from a configuration in $G_{\overline{AB}}(\theta_1^l, \theta_1^u)$ to a configuration in $G_{\overline{CD}}(\theta_2^l, \theta_2^u)$. This problem is new and has not been addressed in the literature. However, in the special case when the line segments $\overline{AB}$, $\overline{CD}$ reduce to points, the DGP simplifies to the *Dubins Interval Problem (DIP)* which has been solved in the literature [25]. Even though the gates generated in this paper correspond to vertical line segments, we make no such assumptions while solving the DGP (Fig. 7).

We will briefly review the main result for the Dubins interval problem as it will be used to solve DGP. Suppose L and R represent the left (counter-clockwise) and the right (clockwise) circular arcs with radius equal to ρ, and let S represent a straight line segment. Also, let L_ψ or R_ψ denote left or right circular arcs with an arc angle equal to ψ. A three segment path for the Dubins interval problem, say $LSR(\theta_1, \theta_2)$, follows the sequence L, S and R, and starts with heading equal to $\theta_1 \in [\theta_1^l, \theta_1^u]$ and ends with heading equal to $\theta_2 \in [\theta_2^l, \theta_2^u]$. Other three segment paths can be defined similarly. For two segment paths, the initial or the final heading angle is specified while the other heading is derived based on the path type. For example, $LS(\theta_1^u, \theta_2(\theta_1^u))$ denotes a LS path that starts at heading equal to θ_1^u and ends at a heading equal to $\theta_2(\theta_1^u)$ which is a function of θ_1^u. The initial and final headings for single segment paths can be specified directly based on the path type. The main result in [25] states that the shortest path for the Dubins interval problem must be one of the following candidate paths[6] or a degenerate form of these:

- *Paths with three segments*: $LSR(\theta_1^u, \theta_2^u)$, $LSL(\theta_1^u, \theta_2^l)$, $LRL(\theta_1^u, \theta_2^l)$, $RSL(\theta_1^l, \theta_2^l)$, $RSR(\theta_1^l, \theta_2^u)$ and $RLR(\theta_1^l, \theta_2^u)$.
- *Paths with two segments*: $LS(\theta_1^u, \theta_2(\theta_1^u))$, $RS(\theta_1^l, \theta_2(\theta_1^u))$, $SL(\theta_1(\theta_2^l), \theta_2^l)$, $SR(\theta_1(\theta_2^u), \theta_2^u)$, $LR(\theta_1^u, \theta_2(\theta_1^u))$, $LR(\theta_1(\theta_2^u), \theta_2^u)$, $RL(\theta_1^l, \theta_2(\theta_1^u))$ and $RL(\theta_1(\theta_2^l), \theta_2^l)$.
- *Paths with one segment*: S, L_ψ and R_ψ, where $\psi > \pi$.

IV. G* Algorithm

The overall pseudo-code of G* is given in Algorithm 1. G* first initializes $\mathcal{G}$ with just two vertices c_s (initial configuration) and c_f (final configuration) and an edge between them (line 14 of Alg. 1). The cost of traveling the edge (c_s, c_f) is set

[6]Note that some of these paths may not be feasible (may not satisfy the heading angle constraints) and, therefore, can be ignored.

Algorithm 1: G*

1 **Inputs:**
2 Ω `// Set of obstacles`
3 $size(obs)$ $\forall$ $obs \in \Omega$ `// sizes of obstacles`
4 c_s, c_f `// Initial, final configurations`
5 τ_i `// Obstacle intersection tolerance`
6 τ_p `// Position continuity tolerance`
7 τ_θ `// Angle continuity tolerance`
8 T_m `// Computational time limit`
9 **Output:**
10 l_{lb} `// Lower bound for CSP`
11 $path_{lb}$ `// Lower bounding path`
12 **Initialization:**
13 $TimeElapsed \leftarrow 0$ `// Running time G*`
14 $V \leftarrow \{c_s, c_f\}, E \leftarrow \{(c_s, c_f)\}$ $\mathcal{G} \leftarrow (V, E)$
15 $cost(c_s, c_f) \leftarrow$ Dubins path length between c_s and c_f
16 $path^* \leftarrow (c_s, c_f)$
17 $path_{lb} \leftarrow$ Dubins path between c_s and c_f
18 **Main Loop:**
19 **while** $path_{lb}$ *is infeasible &* $TimeElapsed \leq T_m$ **do**
20   **for** $obs \in \Omega$ **do**
21   **if** $path_{lb}$ intersects obs **then**
22             $l_c \leftarrow$ chord length of the intersection of $path_{lb}$ with obs
23             $r \leftarrow \frac{l_c}{size(obs)}$
24   **if** $r > \tau_i$ *&* $0 \leq x_c \leq x_f$ **then**
25   Add new gates to $\mathcal{G}$ as in Fig. 8(c)
26   Update the set of edges in $\mathcal{G}$
27   **end**
28   **end**
29   **end**
30     $\mathcal{S}_{dc} \leftarrow$ Set of all the discontinuities in $path_{lb}$
31   **if** $|\mathcal{S}_{dc}| \geq 1$ **then**
32   **for** $\bar{C} \in \mathcal{S}_{dc}$ **do**
            `// Let` $\bar{C} := \{(x_a, y_a, \theta_a), (x_d, y_d, \theta_d)\}$
33   **if** $|y_a - y_d| \geq \tau_p$ *or* $|\theta_a - \theta_d| \geq \tau_\theta$ **then**
34   Delete and add new gates to $\mathcal{G}$ as shown in Fig. 9(b) and Fig. 9(c)
35   Update the set of edges in $\mathcal{G}$
36   **end**
            `// If both` $|y_a - y_d| \geq \tau_p$ `and` $|\theta_a - \theta_d| \geq \tau_\theta$ `are true, we first add new gates as in Fig. 9(b) and then, add more sectors to each new gate as in Fig. 9(c)`
37   **end**
38   **end**
39   **for** *any new edge* (u, v) *added in* $\mathcal{G}$ **do**
40         $cost(u, v) \leftarrow$ optimal Dubins path length from gate u to gate v (solve corresponding DGP).
41   **end**
42     $path^* \leftarrow$ a least-cost path in $\mathcal{G}$
43     $path_{lb} \leftarrow$ The Dubins lower bounding path corresponding to $path^*$
44 **end**
45 $l_{G*} \leftarrow$ sum of the edge costs in $path^*$
46 **return** $l_{G*}, path_{lb}$

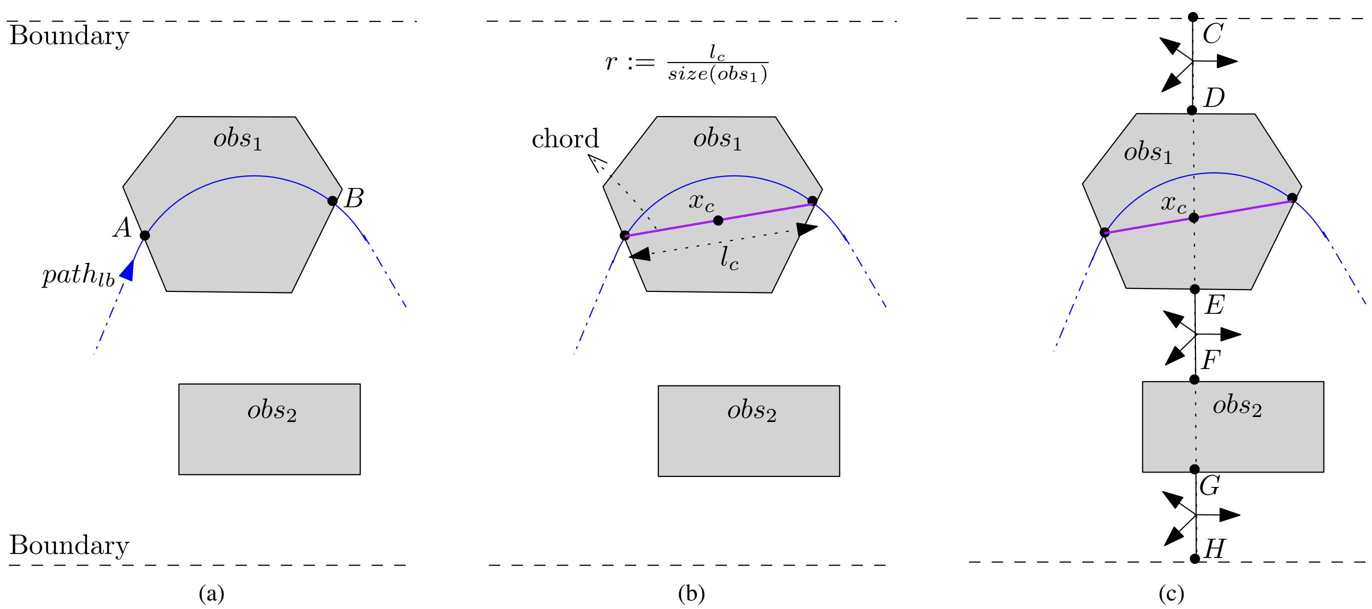


Fig. 8: (a) There are two obstacles, obs_1 and obs_2 in this illustration. $path_{lb}$ is intersecting obs_1 at points A and B. (b) The line segment that joins A and B is referred to as a chord. Here, the parameter r is a measure of the extent to which $path_{lb}$ intersects the obstacle obs_1. Here, the size of a polygonal obstacle is defined as the length of the longest edge of the obstacle. New gates are added by intersecting the vertical line segment passing through the center (x_c) of the chord with the free space. Each line segment in $\{\overline{CD}, \overline{EF}, \overline{GH}\}$ is initially associated with three sectors $[0, \frac{2\pi}{3}], [\frac{2\pi}{3}, \frac{4\pi}{3}]$ and $[\frac{4\pi}{3}, 2\pi]$. Therefore, there are three new gates created corresponding to each line segment in $\{\overline{CD}, \overline{EF}, \overline{GH}\}$.

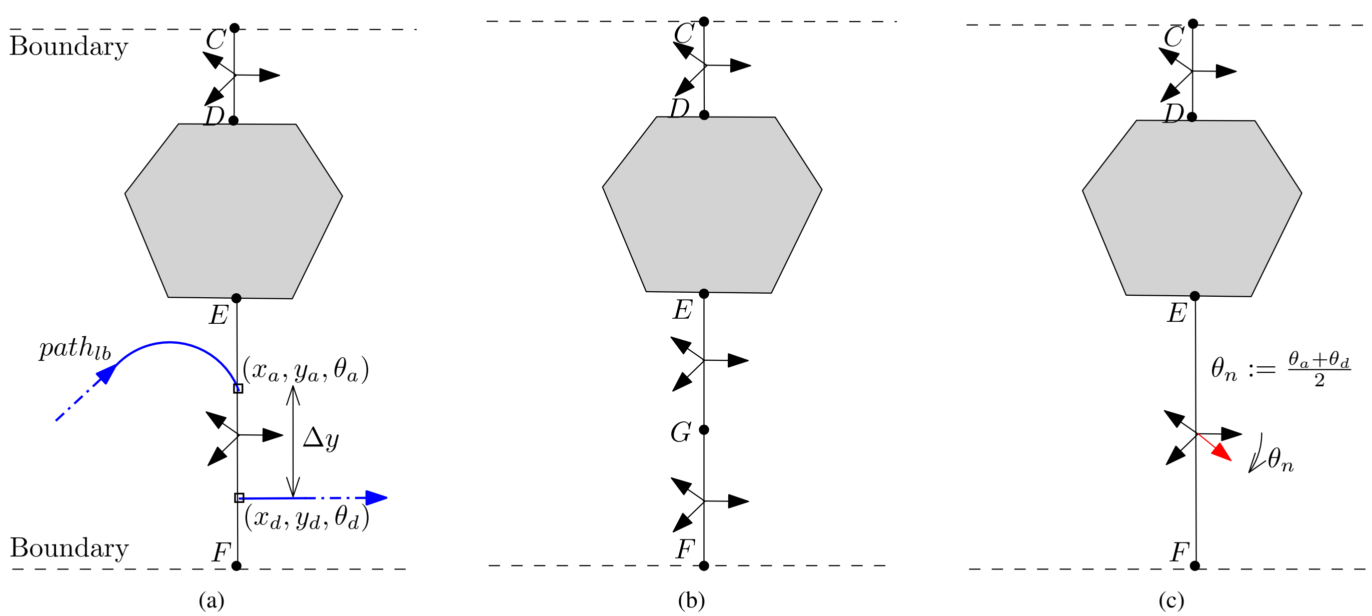


Fig. 9: (a) There are three gates associated with line segment EF, *i.e*, $G_{\overline{EF}}(0, \frac{2\pi}{3})$, $G_{\overline{EF}}(\frac{2\pi}{3}, \frac{4\pi}{3})$, and $G_{\overline{EF}}(\frac{4\pi}{3}, 2\pi)$. A lower bounding path, $path_{lb}$, is reaching gate $G_{\overline{EF}}(\frac{4\pi}{3}, 2\pi)$ at (x_a, y_a, θ_a) and departing $G_{\overline{EF}}(\frac{4\pi}{3}, 2\pi)$ at (x_d, y_d, θ_d). The discontinuity in position here is $\Delta y := y_a - y_d$ and the discontinuity in heading angle is $\Delta\theta := |\theta_a - \theta_d|$. (b) If $\Delta y > \tau_p$, then partition line segment $\overline{EF}$ into two equal segments $\overline{EG}$ and $\overline{GF}$. Delete the gates corresponding to $\overline{EF}$ and add three new gates corresponding to each line segment in $\{\overline{EG}, \overline{GF}\}$. (c) Similarly, if $\Delta\theta > \tau_\theta$, then partition the existing sector $[2\pi/3, 2\pi]$ into $[2\pi/3, \theta_n]$ and $[\theta_n, 2\pi]$ as shown in the figure. Also, delete the gate corresponding to $[2\pi/3, 2\pi]$ and add the new gates.

to the length of the shortest Dubins path from c_s to c_f (line 15 of Alg. 1). The cost of any path in $\mathcal{G}$ is defined as the sum of the cost of the edges in the path. A least-cost path from c_s to c_f found in $\mathcal{G}$ is denoted as $path^*$, and is updated during each iteration of the algorithm. Note that $path^*$ is a sequence of vertices in $\mathcal{G}$, and an edge joining any two adjacent vertices in $path^*$ corresponds to a Dubins path. Therefore, we keep track of $path^*$ as well as its corresponding collection of Dubins paths in $path_{lb}$ (lines 16-17 of Alg. 1). $path_{lb}$ may not be feasible when it crosses a gate; $i.e.$, the arrival configuration (x_a, y_a, θ_a) of the path at a gate may not be equal to the departure configuration (x_d, y_d, θ_d) at the gate (Fig. 9(a)). A *discontinuity* in a path is then defined as a tuple with its unequal arrival and departure configurations. We will later show that $path_{lb}$ found at the end of any iteration of G* is a lower bounding solution to the CSP. If $path_{lb}$ turns out to be feasible ($i.e.$,, it does not intersect the interior of any of the obstacles and does not contain any discontinuities), it must be an optimal solution to the CSP. However, this is generally not the case.

In each iteration of G*, if $path_{lb}$ is infeasible and the run time of G* has not exceeded the limit (T_m), we add new gates to $\mathcal{G}$ and update it based on the type of infeasibility in $path_{lb}$ as follows:

- **Infeasibility type:** $path_{lb}$ **passes through the interior of an obstacle (lines 20-29 of Alg. 1)**: Refer to Fig. 8. We add new gates (as vertices) to $\mathcal{G}$ based on a measure (r) that specifies the extent to which $path_{lb}$ intersects an obstacle. Refer to Fig. 8(b) on how we compute this measure. If this measure exceeds the given *obstacle intersection tolerance* (τ_i) and the center of the chord (x_c) strictly lies between 0 and x_f, we add a fixed number of gates[7] to $\mathcal{G}$ for each line segment as shown in Fig. 8(c).
  Updating $\mathcal{G}$: Consider the partition of V into subsets $V_1, V_2, \cdots, V_l$ (as described in section III) such that $\bar{x}(V_1) < \bar{x}(V_2) \le \bar{x}(V_3) \cdots \le \bar{x}(V_{l-1}) < \bar{x}(V_l)$. Let V_k for some $k \in \{2, \cdots, l-1\}$ be the set of new gates that has been added. To update the edges in $\mathcal{G}$, we (1) delete all the edges from any gate in V_{k-1} to any gate in V_{k+1}, (2) add edges from each gate in V_{k-1} to all the gates in V_k and (3) add edges from each gate in V_k to all the gates in V_{k+1}.

- **Infeasibility type:** $path_{lb}$ **has a path discontinuity in position (lines 30-38 of Alg. 1)**: Refer to Fig. 9(b). If the Euclidean distance between the arriving and departing configurations at a discontinuity is more than a given *position continuity tolerance* (say τ_p), we partition line segment $\overline{EF}$ into two equal segments $\overline{EG}$ and $\overline{GF}$, and add new gates corresponding to each line segment in $\{\overline{EG}, \overline{GF}\}$. We note here that the new gates will inherit the same level of angle discretizations as the gates corresponding to $\overline{EF}$. For example, the gates corresponding to $\overline{EF}$ is associated with three sectors $[0, \frac{2\pi}{3}], [\frac{2\pi}{3}, \frac{4\pi}{3}]$ and $[\frac{4\pi}{3}, 2\pi]$; the same sectors will also be inherited by all the new gates.
  Updating $\mathcal{G}$: Similar to the previous infeasibility type, consider the partition of V into subsets $V_1, V_2, \cdots, V_l$ as defined before. Let the new gates be added to V_k for some $k \in \{2, \cdots, l-1\}$. To update the edges in $\mathcal{G}$, (1) add edges from each gate in V_{k-1} to all the new gates in V_k and (2) add edges from each new gate in V_k to all the gates in V_{k+1}.

- **Infeasibility type:** $path_{lb}$ **has a path discontinuity in heading (lines 30-38 of Alg. 1)**: Refer to Fig. 9(c). If the difference between the arrival and departure headings is more than a given *angle continuity tolerance* (say τ_θ), we partition the gates associated with line segment EF as shown in Fig. 9(c).
  Updating $\mathcal{G}$: This step follows the same procedure as presented for the path discontinuity in position.

The cost of each new edge added can be obtained by solving the DGP (lines 39-41 of Alg. 1). At the end of each iteration of G*, a least-cost path ($path^*$) is computed in $\mathcal{G}$ using Dijkstra's shortest path algorithm [26] (line 42 of Alg. 1), and the iterations continue until the termination criteria are met.

A. Lower Bounding Proof

If we can solve the DGP to optimality (presented in the next section), the following theorem shows that sum of the edge costs in $path^*$ is a lower bound to the CSP problem.

Theorem 1. *Consider a CSP problem instance with a feasible solution. Let $path^*$ be a least-cost path in $\mathcal{G}$ at the end of any iteration of G* applied to the instance. Let l_{G*} denote the sum of the edge costs in $path^*$. Let l_{opt} denote the optimal length of the CSP. Then, $l_{G*} \le l_{opt}$.*

Proof: Consider the set of all the gates V in $\mathcal{G}$. Partition the gates into subsets $V_1, V_2, \cdots, V_l$ (as described in section III) such that $\bar{x}(V_1) \le \bar{x}(V_2) \cdots \le \bar{x}(V_l)$. No continuous path from c_s can reach c_f without passing through at least one of the gates in $V_i, \forall i = 1, \cdots, l$. This implies that any optimal path for the CSP problem must also pass through at least one of the gates in $V_i, \forall i = 1, \cdots, l$. Let a sequence of gates visited by an optimal path be $(g_1, g_2, \cdots, g_l)$ where $g_i \in V_i, i = 1, \cdots, l$. Since, for $i = 1, \cdots, l$, $cost(g_i, g_{i+1})$ denotes the length of the shortest Dubins path from any configuration in g_i to any configuration in g_{i+1}, we get $\sum_{i=1}^{l-1} cost(g_i, g_{i+1}) \le l_{opt}$. Note that $(g_1, g_2, \cdots, g_l)$ is a feasible path in $\mathcal{G}$. Therefore, we also have $l_{G*} \le \sum_{i=1}^{l-1} cost(g_i, g_{i+1})$. Putting these results together, we conclude that $l_{G*} \le l_{opt}$. ∎

V. Solving the Dubins Gate Problem

We use the same notations as the DGP stated in section III. Any position p_1 on line segment $\overline{AB}$ is represented as $p_1 = A + \lambda_1 v_1$ where $\lambda_1 \in [0, 1]$ and $v_1 := B - A$, a vector directed from A to B. Similarly, any position p_2 on line segment $\overline{CD}$ is represented as $p_2 = C + \lambda_2 v_2$ where $\lambda_2 \in [0, 1]$ and $v_2 := D - C$.

[7] In this paper, we initialize each line segment with three sectors.

TABLE I: List of candidate paths for the DGP

Path Mode	Candidate Paths
LSL	$LSL(\lambda_1^e, \theta_1^u, \lambda_2^e, \theta_2^l)$, $LSL(\lambda_1^e, \theta_1^u, \lambda_2^*, \theta_2^l)$, $LSL(\lambda_1^*, \theta_1^u, \lambda_2^e, \theta_2^l)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$
LSR	$LSR(\lambda_1^e, \theta_1^u, \lambda_2^e, \theta_2^u)$, $LSR(\lambda_1^e, \theta_1^u, \lambda_2^*, \theta_2^u)$, $LSR(\lambda_1^*, \theta_1^u, \lambda_2^e, \theta_2^u)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$
RSL	$RSL(\lambda_1^e, \theta_1^l, \lambda_2^e, \theta_2^l)$, $RSL(\lambda_1^e, \theta_1^l, \lambda_2^*, \theta_2^l)$, $RSL(\lambda_1^*, \theta_1^l, \lambda_2^e, \theta_2^l)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$
RSR	$RSR(\lambda_1^e, \theta_1^l, \lambda_2^e, \theta_2^u)$, $RSR(\lambda_1^e, \theta_1^l, \lambda_2^*, \theta_2^u)$, $RSR(\lambda_1^*, \theta_1^l, \lambda_2^e, \theta_2^u)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$
LRL	$LRL(\lambda_1^e, \theta_1^u, \lambda_2^e, \theta_2^l)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$
RLR	$RLR(\lambda_1^e, \theta_1^l, \lambda_2^e, \theta_2^u)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$
LS	$LS(\lambda_1^e, \theta_1^u, \lambda_2^e, \theta_2(\lambda_1^e, \lambda_2^e))$, $LS(\lambda_1^e, \theta_1^u, \lambda_2^*, \theta_2(\lambda_1^e, \lambda_2^*))$, $LS(\lambda_1^*, \theta_1^u, \lambda_2^e, \theta_2(\lambda_1^*, \lambda_2^e))$, $LS(\lambda_1^e, \theta_1^u, \lambda_2(\theta_2^e), \theta_2^e)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$, $\theta_2^e \in \{\theta_2^l, \theta_2^u\}$
RS	$RS(\lambda_1^e, \theta_1^l, \lambda_2^e, \theta_2(\lambda_1^e, \lambda_2^e))$, $RS(\lambda_1^e, \theta_1^l, \lambda_2^*, \theta_2(\lambda_1^e, \lambda_2^*))$, $RS(\lambda_1^*, \theta_1^l, \lambda_2^e, \theta_2(\lambda_1^*, \lambda_2^e))$, $RS(\lambda_1^e, \theta_1^l, \lambda_2(\theta_2^e), \theta_2^e)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$, $\theta_2^e \in \{\theta_2^l, \theta_2^u\}$
SL	$SL(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^e), \lambda_2^e, \theta_2^l)$, $SL(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^*), \lambda_2^*, \theta_2^l)$, $SL(\lambda_1^*, \theta_1(\lambda_1^*, \lambda_2^e), \lambda_2^e, \theta_2^l)$, $SL(\lambda_1(\theta_1^e), \theta_1^e, \lambda_2^e, \theta_2^l)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$, $\theta_1^e \in \{\theta_1^l, \theta_1^u\}$
SR	$SR(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^e), \lambda_2^e, \theta_2^u)$, $SR(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^*), \lambda_2^*, \theta_2^u)$, $SR(\lambda_1^*, \theta_1(\lambda_1^*, \lambda_2^e), \lambda_2^e, \theta_2^u)$, $SR(\lambda_1(\theta_1^e), \theta_1^e, \lambda_2^e, \theta_2^u)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$, $\theta_1^e \in \{\theta_1^l, \theta_1^u\}$
LR	$LR(\lambda_1^e, \theta_1^u, \lambda_2^e, \theta_2(\lambda_1^e, \lambda_2^e))$, $LR(\lambda_1^e, \theta_1^u, \lambda_2^*, \theta_2(\lambda_1^e, \lambda_2^*))$, $LR(\lambda_1^*, \theta_1^u, \lambda_2^e, \theta_2(\lambda_1^*, \lambda_2^e))$, $LR(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^e), \lambda_2^e, \theta_2^u)$, $LR(\lambda_1^*, \theta_1(\lambda_1^*, \lambda_2^e), \lambda_2^e, \theta_2^u)$, $LR(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^*), \lambda_2^*, \theta_2^u)$ $LR(\lambda_1^e, \theta_1^u, \lambda_2(\theta_2^e), \theta_2^e)$, $LR(\lambda_1(\theta_1^e), \theta_1^e, \lambda_2^e, \theta_2^u)$ for $\lambda_1^e, \lambda_2^e \in \{0,1\}$, $\theta_i^e, \in \{\theta_i^l, \theta_i^u\}$
RL	$RL(\lambda_1^e, \theta_1^l, \lambda_2^e, \theta_2(\lambda_1^e, \lambda_2^e))$, $RL(\lambda_1^e, \theta_1^l, \lambda_2^*, \theta_2(\lambda_1^e, \lambda_2^*))$, $RL(\lambda_1^*, \theta_1^l, \lambda_2^e, \theta_2(\lambda_1^*, \lambda_2^e))$, $RL(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^e), \lambda_2^e, \theta_2^l)$, $RL(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^*), \lambda_2^*, \theta_2^l\ )$, $RL(\lambda_1^*, \theta_1(\lambda_1^*, \lambda_2^e), \lambda_2^e, \theta_2^l\ )$ $RL(\lambda_1^e, \theta_1^l, \lambda_2(\theta_2^e), \theta_2^e)$, $RL(\lambda_1(\theta_1^e), \theta_1^e, \lambda_2^e, \theta_2^l)$ for $\lambda_1^e, \lambda_2^e \in \{0,1\}$, $\theta_i^e, \in \{\theta_i^l, \theta_i^u\}$
L or R	$\mathcal{P}(\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^e), \lambda_2^e, \theta_2(\lambda_1^e, \lambda_2^e))$, $\mathcal{P}(\lambda_1^e, \theta_1^e, \lambda_2(\lambda_1^e, \theta_1^e), \theta_2(\lambda_1^e, \theta_1^e))$, $\mathcal{P}(\lambda_1^e, \theta_1(\lambda_1^e, \theta_2^e), \lambda_2(\lambda_1^e, \theta_2^e), \theta_2^e)$, $\mathcal{P}(\lambda_1(\lambda_2^e, \theta_2^e), \theta_1(\lambda_2^e, \theta_2^e), \lambda_2^e, \theta_2^e)$, $\mathcal{P}(\lambda_1(\theta_1^e, \lambda_2^e), \theta_1^e, \lambda_2^e, \theta_2(\theta_1^e, \lambda_2^e))$, $\mathcal{P}(\lambda_1(\theta_1^e, \theta_2^e), \theta_1^e, \lambda_2(\theta_1^e, \theta_2^e), \theta_2^e)$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$, $\theta_i^e, \in \{\theta_i^l, \theta_i^u\}$
S	$S((\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^e), \lambda_2^e, \theta_2(\lambda_1^e, \lambda_2^e))$, $S((\lambda_1^e, \theta_1(\lambda_1^e, \lambda_2^*), \lambda_2^*, \theta_2(\lambda_1^e, \lambda_2^*))$, $S((\lambda_1^*, \theta_1(\lambda_1^*, \lambda_2^e), \lambda_2^e, \theta_2(\lambda_1^*, \lambda_2^e))$ $S((\lambda_1^e, \theta_1(\lambda_1^e, \theta_2^e), \lambda_2(\lambda_1^e, \theta_2^e), \theta_2^e)$, $S((\lambda_1(\theta_1^e, \lambda_2^e), \theta_1^e, \lambda_2^e, \theta_2(\theta_1^e, \lambda_2^e,))$, for $\lambda_1^e, \lambda_2^e \in \{0,1\}$, $\theta_i^e, \in \{\theta_i^l, \theta_i^u\}$

Let $\overline{\mathcal{P}} = \{LSL, RSR, RSL, LSR, LRL, RLR, LS, RS, SL, SR, LR, RL, L, R, S\}$ denote the set of all the Dubins path modes possible between two configurations. We denote the length of the path of a given Dubins type $\mathcal{P} \in \overline{\mathcal{P}}$ between an initial configuration, defined by (λ_1, θ_1), and a final configuration, defined by (λ_2, θ_2), as $l_{\mathcal{P}}(\lambda_1, \theta_1, \lambda_2, \theta_2)$. Let $l_{\mathcal{D}}(\lambda_1, \theta_1, \lambda_2, \theta_2) := \min_{\mathcal{P} \in \overline{\mathcal{P}}} l_{\mathcal{P}}(\lambda_1, \theta_1, \lambda_2, \theta_2)$. The Dubins Gate Problem (DGP) can be re-stated as follows:

$$\min_{\lambda_1, \lambda_2, \theta_1, \theta_2} l_{\mathcal{D}}(\lambda_1, \theta_1, \lambda_2, \theta_2),$$
$$\text{subject to } \lambda_1, \lambda_2 \in [0,1],\ \theta_1 \in [\theta_1^l, \theta_1^u],\ \theta_2 \in [\theta_2^l, \theta_2^u].$$

For a given λ_1 and λ_2, the DGP reduces to the Dubins Interval Problem (DIP), the solution of which must be one of the candidate paths presented in Section III. To solve DGP, we consider each candidate path for DIP and optimize over $\lambda_1 \in [0,1]$ and $\lambda_2 \in [0,1]$. We will now present the main results for each of these candidate paths; **the proofs of all the Lemmas are in the appendix**.

A. Three segment paths

Broadly all the three segment paths can be categorized as either a CSC or a CCC path where C stands for the circular arc turning left (L) or right (R).

1) CSC Path: Let λ_1^* correspond to $p_1^* \in \overline{AB}$ such that the S segment in the CSC path from (p_1^*, θ_1) to (p_2, θ_2) is perpendicular to $\overline{AB}$. The length of such path is denoted as $l_{CSC}(\lambda_1^*, \lambda_2)$[8]; if such a path doesn't exist, we set $l_{CSC}(\lambda_1^*, \lambda_2)$ to ∞. For this category, note that the headings θ_1 and θ_2 are given, and therefore, we do not state the length, l_{CSC}, as a function of the headings also. Similarly, let λ_2^* correspond to $p_2^* \in \overline{CD}$ such that the S segment in the CSC path from (p_1, θ_1) to (p_2^*, θ_2) is perpendicular to $\overline{CD}$.

Lemma 1.
$\min_{\lambda_1, \lambda_2 \in [0,1]} l_{CSC}(\lambda_1, \lambda_2) = \min\{l_{CSC}(\lambda_1^e, \lambda_2^e), l_{CSC}(\lambda_1^*, \lambda_2^e), l_{CSC}(\lambda_1^e, \lambda_2^*)\}, \lambda_1^e, \lambda_2^e \in \{0,1\}\}$.

2) CCC Path:

Lemma 2.
$\min_{\lambda_1, \lambda_2 \in [0,1]} l_{CCC}(\lambda_1, \lambda_2) = \min\{l_{CCC}(\lambda_1^e, \lambda_2^e), \lambda_1^e, \lambda_2^e \in \{0,1\}\}$.

B. Two Segment Paths

In this section, we analyze the two-segment paths CS, SC, and CC. For a given p_1 (or λ_1) and θ_1, the final heading of any two-segment path, θ_2, is a function of p_2 (or λ_2), and cannot be independently chosen. Similarly, for a given p_2 and θ_2, the initial heading θ_1 is a function of p_1 (or λ_1). Let p_i be the inflection point on the two-segment path.

1) CS or CC: We consider the CS or CC paths where θ_1 is given and θ_2 can lie in the interval $[\theta_2^l, \theta_2^u]$. For a given λ_1, let λ_2^* represent $p_2^* \in \overline{CD}$, that corresponds to a final position of a CS path, such that $\overline{p_i p_2^*}$ is perpendicular to $\overline{CD}$; let the length of such CS path be $l_{CS}(\lambda_1, \lambda_2^*)$. Similarly, for a given λ_2, $l_{CS}(\lambda_1^*, \lambda_2)$ is the length of a CS path, where $\overline{p_i p_2}$ is perpendicular to $\overline{AB}$. Let $l_{CS}(\lambda_1^*, \lambda_2^*)$ be the length of the CS path where $\overline{p_i, p_2^*}$ is perpendicular to both $\overline{AB}$ and $\overline{CD}$; such a path exists only when $\overline{AB}$ and $\overline{CD}$ are parallel. Moreover, the length, $l_{CS}(\lambda_1^*, \lambda_2^*)$, would be same as $l_{CS}(\lambda_1^*, \lambda_2)$ or $l_{CS}(\lambda_1, \lambda_2^*)$.

[8]For simplicity, l_{CSC} is not explicitly shown as a function of θ_1 and θ_2.

Let λ_2^l (or λ_2^u) correspond to the position $p_2^l \in \overline{CD}$ (or p_2^u), such that the final heading, $\theta_2(\lambda_2^l)$ (or $\theta_2(\lambda_2^u)$), is equal to θ_2^l (or θ_2^u). The definitions for the CC paths are similar to that of the CS paths.

Lemma 3. *For* $\mathcal{P} \in \{CS, CC\}$, $\min_{\lambda_1,\lambda_2\in[0,1]} l_{\mathcal{P}}(\lambda_1, \lambda_2) = \min\{l_{\mathcal{P}}(\lambda_1^e, \lambda_2^e), l_{\mathcal{P}}(\lambda_1^*, \lambda_2^e), l_{\mathcal{P}}(\lambda_1^e, \lambda_2^*), l_{\mathcal{P}}(\lambda_1^e, \lambda_2^l), l_{\mathcal{P}}(\lambda_1^e, \lambda_2^u),\ \lambda_1^e, \lambda_2^e \in \{0, 1\}\}$.

2) SC or CC: We consider the SC paths where θ_2 is given and θ_1 can lie in the interval $[\theta_1^l, \theta_1^u]$. The definition of the critical and boundary points is similar to that of the CS paths with few differences. For a given λ_2, $l_{SC}(\lambda_1^*, \lambda_2)$ is the length of a SC path, where $\overline{p_1 p_i}$ is perpendicular to $\overline{AB}$. For a given λ_1, $l_{SC}(\lambda_1, \lambda_2^*)$ is the length of a SC path, where $\overline{p_1 p_i}$ is perpendicular to $\overline{CD}$.

Let λ_1^l (or λ_1^u) correspond to the position $p_1^l \in \overline{AB}$ (or p_1^u), such that the initial heading, $\theta_1(\lambda_1^l)$ (or $\theta_1(\lambda_1^u)$), is equal to θ_1^l (or θ_1^u). The definitions for the CC paths are similar to that of the SC paths.

Lemma 4. *For* $\mathcal{P} \in \{SC, CC\}$, $\min_{\lambda_1,\lambda_2\in[0,1]} l_{\mathcal{P}}(\lambda_1, \lambda_2) = \min\{l_{\mathcal{P}}(\lambda_1^e, \lambda_2^e), l_{\mathcal{P}}(\lambda_1^*, \lambda_2^e), l_{\mathcal{P}}(\lambda_1^e, \lambda_2^*), l_{\mathcal{P}}(\lambda_1^l, \lambda_2^e), l_{\mathcal{P}}(\lambda_1^u, \lambda_2^e),\ \lambda_1^e, \lambda_2^e \in \{0, 1\}\}$.

C. One Segment Paths (C or S)

The one segment turns (L or R) are candidate solutions for DIP only when the turn angle is greater than π. We consider such paths here, and minimize over λ_1 and λ_2. The definitions of the boundary positions, λ_i^l, λ_i^u, $i = 1, 2$, are similar to the boundary positions defined for the CS or SC paths.

Lemma 5. *For* $\mathcal{P} \in \{L, R\}$, $\min_{\lambda_1,\lambda_2\in[0,1]} l_{\mathcal{P}}(\lambda_1, \lambda_2) = \min\{l_{\mathcal{P}}(\lambda_1^e, \lambda_2^e), l_{\mathcal{P}}(\lambda_1^e, \lambda_2^l), l_{\mathcal{P}}(\lambda_1^e, \lambda_2^u), l_{\mathcal{P}}(\lambda_1^l, \lambda_2^e), l_{\mathcal{P}}(\lambda_1^u, \lambda_2^e),\ \lambda_1^e, \lambda_2^e \in \{0, 1\}\}$.

Consider the paths that have just one straight line segment; for a given position $p_1(\lambda_1)$, let λ_2^* correspond to a position p_2, such that the straight line segment is perpendicular to $\overline{CD}$. For a given λ_2, λ_1^* is similarly defined.

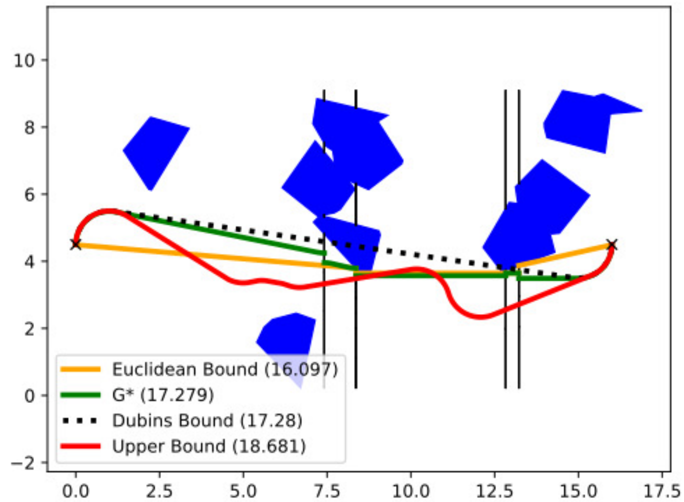
Lemma 6.
$\min_{\lambda_1,\lambda_2\in[0,1]} l_S(\lambda_1, \lambda_2) = \min\{l_S(\lambda_1^e, \lambda_2^e), l_S(\lambda_1^e, \lambda_2^*), l_S(\lambda_1^*, \lambda_2^e), l_S(\lambda_1^e, \lambda_2^l), l_S(\lambda_1^e, \lambda_2^u), l_S(\lambda_1^l, \lambda_2^e), l_S(\lambda_1^u, \lambda_2^e),\ \lambda_1^e, \lambda_2^e \in \{0, 1\}\}$.

D. Candidate Paths for the Dubins Gate Problem

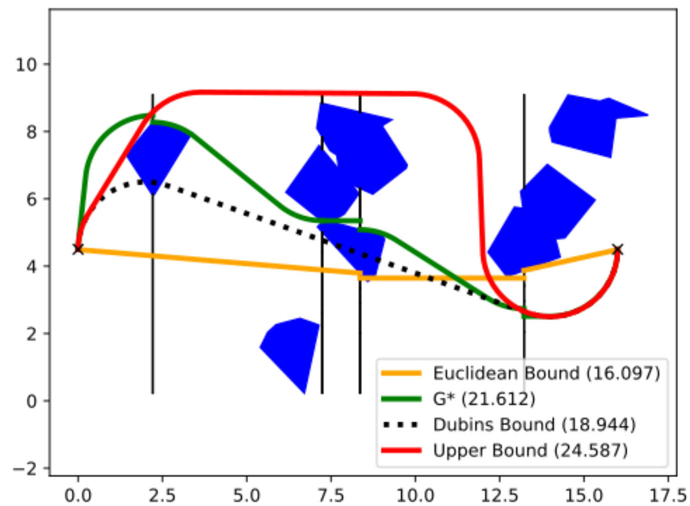
The candidate paths for finding the optimum of the Dubins Gate Problem are listed in the Table I.

VI. Numerical Results

We generated a set of thirty maps, ten each with 10, 15 and 20 obstacles. The obstacles are randomly generated convex polygons and discs in an area of dimensions 16×9 units of distance. We set the Euclidean distance between the initial and final configurations to be 16 units. Also, the heading angle at



(a) Turning radius (ρ) = 1



(b) Turning radius (ρ) = 2

Fig. 10: Comparison of the paths generated by the lower and upper bounding algorithms for an instance with polygonal obstacles.

the initial and final configurations were chosen[9] to be 90° for all the instances except for the ones where the heading angles are varied.

The upper bounds for the CSP were computed using the Open Motion Planning Library (OMPL) [27]. A Dubins State Space was defined, and the feasible solutions were generated using RRT* [18], BIT* [19], and FMT* [20] algorithms. A computational time limit of 10 minutes was set for all the algorithms. We used the best feasible solution generated using these algorithms and its length is set as the upper bound (l_{UB}) for the CSP problem. The best trivial lower bound (l_{LB}) was obtained by choosing the maximum of lengths of the two

[9]We note here that the initial (or the final) configuration can be set to any angle. We chose instances with initial and final configurations set to 90° as we found these instances to be harder to solve in our preliminary tests. G* works for any initial and final configuration, and these configurations do not have to equal.

TABLE II: Performance of G* with varying ρ

Obstacles	Radius (ρ)	Trivial LB (l_{LB})	% Improvement of G* Avg.	Max	Optimality Gap w.r.t. l_{LB} Avg.	Max.	Optimality Gap w.r.t. l_{G*} Avg.	Max.
10	1	18.456	0.753	8.349	2.947	22.937	1.935	13.402
	2	19.886	8.846	24.604	18.394	51.439	10.429	51.536
	3	22.386	23.264	57.395	46.278	62.287	14.698	65.213
15	1	18.456	2.660	19.405	8.264	24.890	2.825	7.443
	2	19.886	18.946	47.558	28.365	56.259	12.890	38.283
	3	22.386	38.826	58.297	54.294	69.698	13.457	44.936
20	1	18.456	8.425	18.917	12.450	28.258	4.583	9.789
	2	19.886	25.647	51.890	42.896	62.846	14.697	54.670
	3	22.386	44.637	58.294	62.485	69.256	16.738	48.286

paths obtained by 1) solving the CSP without the obstacles (provides the Dubins bound), and 2) solving the CSP ignoring the turning radius constraints as discussed in the introduction (provides the Euclidean bound).

G* was implemented in Python 3.6. Similar to the other algorithms, the computational time limit of G* was also set to 10 minutes. All computations were conducted on a computer with a 2.80 GHz Intel Core i7-7700HQ processor running Ubuntu 16.04. An illustration of the paths generated by the lower bounding algorithms, G*, and the best upper bounding solution (from RRT*,BIT*, FMT*) using one of the maps are shown in Fig. 10a and Fig. 10b. Here, the paths were computed for an instance with ten obstacles and with turning radius $\rho = 1$ and $\rho = 2$.

To evaluate the performance of G*, we vary the minimum turning radius of the robot ($\rho = 1, 2, 3$), the three tolerances (τ_i = 0.1, 0.2, 0.3, τ_p = 0.1, 0.2, 0.3, $\tau_\theta = 15°, 30°, 45°$) as well as the initial and final heading angles of the robots on all the 30 maps. Finally, we also present the performance of G* on the instances discussed in the introduction (Fig. 2).

A. *Impact of the minimum turning radius (ρ)*

For a given number of obstacles and ρ (referred to as case), we tested the algorithm on 10 maps. Each instance corresponds to one of the maps, and a value assigned to each of the tolerances. Since we have three different tolerances ($\tau_i, \tau_p, \tau_\theta$) and three values for each tolerance, for each case, the algorithms were tested on a total of 270 instances. For each case, the average and maximum of the bounds obtained are presented in Table II. Note that the trivial bound (l_{LB}) for each case is independent of the tolerances, and therefore there is only one value. As expected, as ρ increased, the average % improvement of G* bounds with respect to l_{LB} increased from 0.75% to 44.63%. A maximum improvement of 58.29% was observed for instances with 20 obstacles and $\rho = 3$. This improvement in the lower bounds has a direct impact on our understanding of the quality of the feasible solutions; specifically, in Table II, we can compare the optimality gaps with respect to (w.r.t.) l_{LB} *versus* the optimality gaps w.r.t. l_{G*}. For example, for maps with 15 obstacles and $\rho = 3$, the optimality gap w.r.t. l_{G*} improved to 13.45% on an average as compared to 54.29% w.r.t the l_{LB}.

Fig. 11 presents the reduction in the optimality gap due to G*. The case denoted as "$o10\_r1$" corresponds to a map with

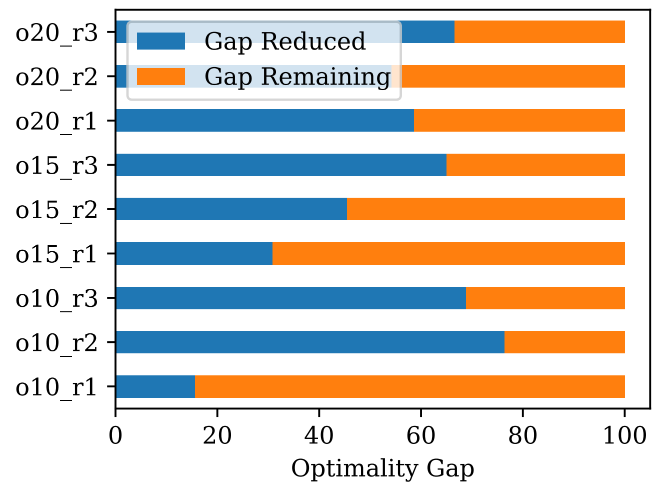


Fig. 11: Improvement of the optimality gap by the G* bounds.

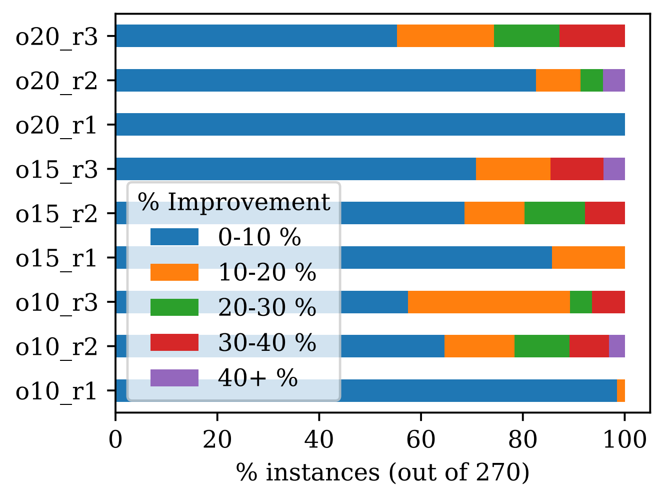


Fig. 12: A plot comparing the percentage split of instances (set of 270) based on their G* lower bound improvements.

10 obstacles and $\rho = 1$. This format of the case name applies to the other cases as well. The optimality gap with respect to the trivial lower bound is scaled to 100%, and the reduction in the gap due to the G* bounds and the remaining gap is shown in blue and orange respectively. The gap between the lower

TABLE III: Performance of G* with varying τ_i

Obstacles	**Intersection Tolerance**	**Trivial LB** (l_{LB})	**% Improvement of G*** Avg.	Max	**Optimality Gap w.r.t.** l_{LB} Avg.	Max.	**Optimality Gap w.r.t.** l_{G*} Avg.	Max.
10	τ_i = 0.1	18.296	15.978	54.235	26.468	72.349	9.350	45.274
	τ_i = 0.2	18.429	15.893	54.235	27.593	72.349	11.239	45.274
	τ_i = 0.3	18.739	14.847	54.235	28.266	72.349	12.348	45.274
15	τ_i = 0.1	18.197	16.115	58.927	28.561	79.766	11.958	48.594
	τ_i = 0.2	18.278	15.933	58.927	30.428	79.766	11.395	48.594
	τ_i = 0.3	18.982	14.629	58.927	31.521	79.766	12.349	48.594
20	τ_i = 0.1	19.043	13.385	61.589	38.653	78.350	13.395	56.467
	τ_i = 0.2	19.303	14.923	61.589	40.589	78.350	12.350	56.467
	τ_i = 0.3	19.184	14.573	61.589	41.842	78.350	14.234	56.467

TABLE IV: Performance of G* with varying τ_p

Obstacles	**Position Tolerance**	**Trivial LB** (l_{LB})	**% Improvement of G*** Avg.	Max	**Optimality Gap w.r.t.** l_{LB} Avg.	Max.	**Optimality Gap w.r.t.** l_{G*} Avg.	Max.
10	τ_p = 0.1	18.294	14.234	54.235	25.693	72.349	10.395	45.274
	τ_p = 0.2	18.429	15.893	54.235	27.593	72.349	11.239	45.274
	τ_p = 0.3	18.829	15.235	54.235	28.962	72.349	12.469	45.274
15	τ_p = 0.1	18.694	15.235	58.927	28.498	79.766	10.291	48.594
	τ_p = 0.2	18.278	15.933	58.927	30.428	79.766	11.395	48.594
	τ_p = 0.3	19.013	15.823	58.927	30.947	79.766	11.598	48.594
20	τ_p = 0.1	19.021	14.014	61.589	39.238	78.350	9.348	56.467
	τ_p = 0.2	19.303	14.923	61.589	40.589	78.350	12.350	56.467
	τ_p = 0.3	19.184	14.235	61.589	41.345	78.350	14.985	56.467

TABLE V: Performance of G* with varying τ_θ

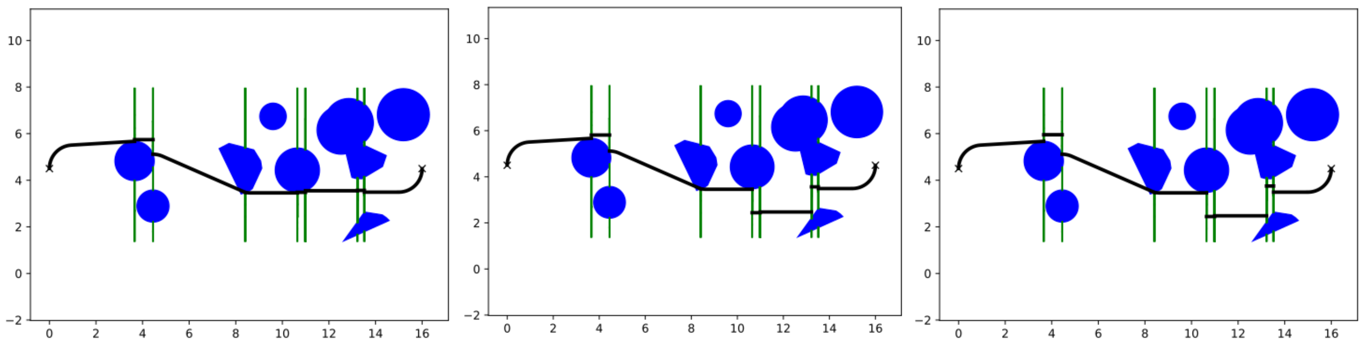
Obstacles	**Angle Tolerance**	**Trivial LB** (l_{LB})	**% Improvement of G*** Avg.	Max	**Optimality Gap w.r.t.** l_{LB} Avg.	Max.	**Optimality Gap w.r.t.** l_{G*} Avg.	Max.
10	$\tau_\theta = 15°$	18.429	15.893	54.235	27.593	72.349	11.239	45.274
	$\tau_\theta = 30°$	18.429	15.893	54.235	27.593	72.349	11.239	45.274
	$\tau_\theta = 45°$	18.429	15.893	54.235	27.593	72.349	11.239	45.274
15	$\tau_\theta = 15°$	18.278	15.933	58.927	30.428	79.766	11.395	48.594
	$\tau_\theta = 30°$	18.278	15.933	58.927	30.428	79.766	11.395	48.594
	$\tau_\theta = 45°$	18.278	15.933	58.927	30.428	79.766	11.395	48.594
20	$\tau_\theta = 15°$	19.303	14.923	61.589	40.589	78.350	12.350	56.467
	$\tau_\theta = 30°$	19.303	14.923	61.589	40.589	78.350	12.350	56.467
	$\tau_\theta = 45°$	19.303	14.923	61.589	40.589	78.350	12.350	56.467

TABLE VI: Performance of G* with varying initial/final heading angles

Heading (θ)	**Obstacles**	**Trivial LB** (l_{LB})	**% Improvement of G*** Avg.	Max	**Optimality Gap w.r.t.** l_{LB} Avg.	Max.	**Optimality Gap w.r.t.** l_{G*} Avg.	Max.
0	10	16.098	3.124	52.846	18.350	68.348	22.395	41.374
	15	16.106	3.259	54.388	26.457	68.239	24.587	64.234
	20	16.129	3.294	58.982	39.275	76.399	34.584	76.436
$\frac{\pi}{2}$	10	18.237	15.346	62.439	29.383	74.982	12.439	48.240
	15	18.497	15.683	61.237	32.349	76.498	12.985	43.249
	20	18.840	15.824	63.987	39.235	77.386	13.239	58.242
π	10	28.240	3.835	25.289	8.240	36.486	5.399	6.223
	15	28.458	3.392	25.399	12.346	37.985	8.499	16.244
	20	28.430	3.554	25.987	12.784	38.595	8.350	14.387
$\frac{3\pi}{2}$	10	18.937	24.239	84.235	38.395	88.346	7.346	28.364
	15	18.958	24.275	84.797	42.345	88.837	9.456	26.236
	20	19.064	24.336	84.679	41.785	89.397	9.973	34.235

and the upper bounds is reduced by 45-75% in most cases (except for the $o10\_r1$ and $o15\_r1$ cases). That is due to the fact that these cases consists of relatively *easier* instances, and the gap with respect to the trivial lower bound itself is quite low.

Fig. 12 captures the distribution of instances for different ranges of percent gap reduction. The optimality gap reduction by G* bounds for most instances of the case $o10\_r1$ were under 10%. However, for instances with a higher number of obstacles and larger turning radii, we observe a significantly higher reduction in the gap.



(a) $\tau_p = 0.1 (l_{G^*} = 17.6978)$ (b) $\tau_p = 0.2 (l_{G^*} = 17.6854)$ (c) $\tau_p = 0.3 (l_{G^*} = 17.6289)$

Fig. 13: Comparison of position tolerance (τ_p) on G* bounds on an instance with 10 obstacles for $\rho = 1$.

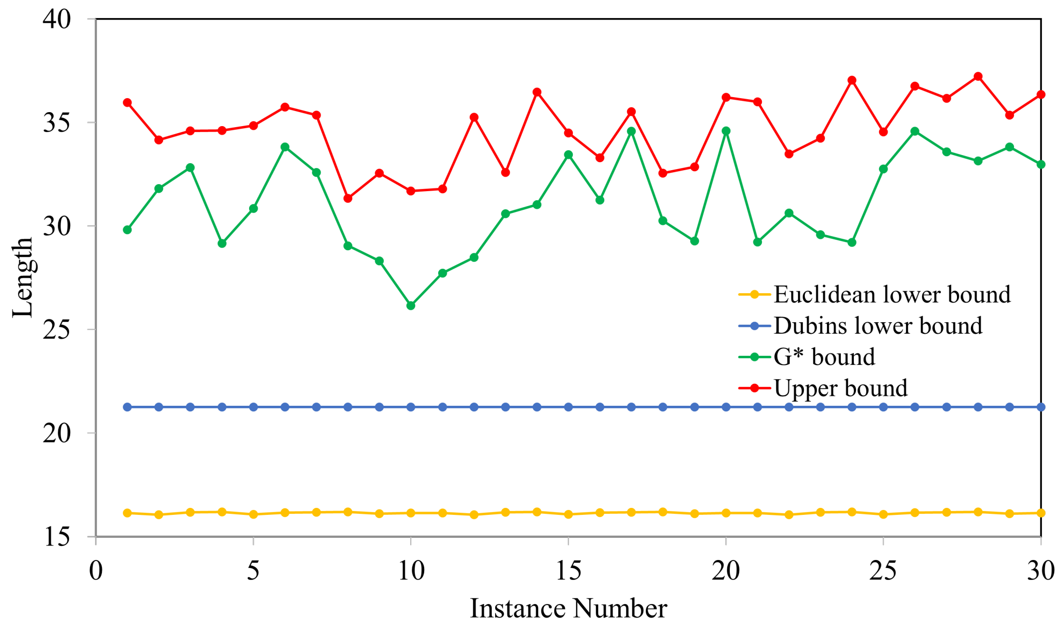


Fig. 14: A comparison of different bounds against G* bound for 30 instances. Each instance is generated on a map with dimensions 16 units $\times$ 9 units and has 10 or 15 or 20 obstacles. The minimum turning radius of the vehicle is set to 3 units.

B. Impact of the tolerances

The bounds obtained by varying the *obstacle intersection tolerance* (τ_i), the *position continuity tolerance* (τ_p), and the *angle continuity tolerance* (τ_θ) are presented in the Tables III, IV and V respectively. The values of τ_i, τ_p, and τ_θ are set to 0.2, 0.2 and $15°$ respectively, whenever that particular tolerance is not varied. In general, we can observe a trend that the gap between upper bound and the G* bounds are the lowest when the tolerances are the lowest. This is expected as G* bounds tend to get closer to the upper bound as $\tau_i, \tau_p, \tau_\theta$ gets smaller. An illustration of how τ_p affects the lower bounding paths found by G* is shown in Fig. 13.

C. Impact of the initial and final heading angles

We set the initial and final heading angle to be equal to θ, and chose four values $(0, \frac{\pi}{2}, \pi, \frac{3\pi}{2})$ for θ. Other parameters were chosen as follows: $\rho = 2$, $\tau_i = 0.2$, $\tau_p = 0.2$, $\tau_\theta = 15°$. The performance of G* for different values of θ are shown in Table VI. The reduction in the optimality gap was the highest when $\theta = \frac{3\pi}{2}$ and lowest when $\theta = 0$. These differences are likely a result of the distribution and the shapes of the obstacles with respect to the initial and final configurations.

D. G* Bounds for the instances presented in Fig. 2

For the instances in Fig. 2, we used the following parameters: $\rho = 3$, $\tau_i = 0.2$, $\tau_p = 0.2$, $\tau_\theta = 15°$. Bounds obtained using G* along with others are presented in Fig. 14. These results suggest G* can provide significant improvement over the existing lower bounding approaches. Also, in Fig. 15, we plot the convergence of the upper and lower bounds for a specific instance as a function of the running time of the algorithms. Clearly, the upper bounds (generated by asymptotically optimal algorithms) continue to decrease while the bounds generated by G* continue to increase with respect to the computational time.

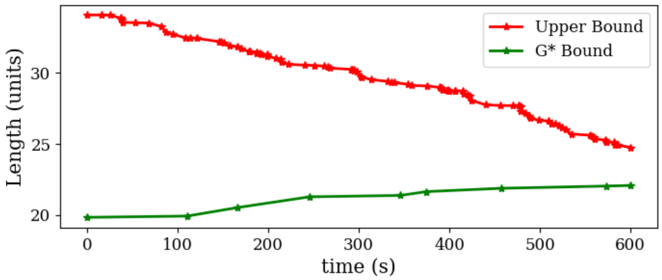


Fig. 15: A comparison of G* bound and Upper Bound convergence for a runtime of 10 minutes. The instance under consideration is generated on a map with dimensions 16 units $\times$ 9 units and has 15 obstacles. The minimum turning radius of the vehicle is set to 2 units.

VII. Conclusion

We presented G* that computes lower bounds to the CSP problem in the presence of a general class of obstacles. G* relies on optimally solving a new motion planning problem called the Dubins Gate problem (DGP). We find optimal solutions for the DGP and prove that the cost of the solution produced by G* is a lower bound to the CSP problem. Extensive numerical results were also presented to corroborate the performance of G*.

G* can be extended and generalized in several ways. If there is no computational time limit and the tolerances converge to zero, we would first like to show that the bounds produced by G* also converge to the optimum of the CSP problem. Another aspect is that the gates generated in this article correspond to vertical line segments, and only the forward connecting edges between the gates are added. This approach may not be suitable for maps such as mazes where the obstacles can intersect the boundaries of the map. This could be addressed by generalizing the gate generation process using cues from road-maps. Another future direction could be in constructing feasible paths for the CSP problem based on the lower bounding solutions, and showing approximation bounds.

Acknowledgment

This material is based upon work partially supported by the National Science Foundation under Grant No. 2120219. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

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

[2] J. T. SCHWARTZ and M. SHARIR, "CHAPTER 8 - Algorithmic Motion Planning in Robotics," in *Algorithms and Complexity* (J. VAN LEEUWEN, ed.), Handbook of Theoretical Computer Science, pp. 391–430, Amsterdam: Elsevier, 1990.

[3] D. Halperin, L. E. Kavraki, and J.-C. Latombe, "Robotics," in *Handbook of Discrete and Computational Geometry* (J. E. Goodman and J. O'Rourke, eds.), ch. 48, pp. 1065–1093, Boca Raton, FL: CRC Press LLC, 2004.

[4] J.-P. Laumond, "Finding collision-free smooth trajectories for a non-holonomic mobile robot," in *Proceedings of the 10th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'87, (San Francisco, CA, USA), p. 1120–1123, Morgan Kaufmann Publishers Inc., 1987.

[5] J.-P. Laumond, P. Jacobs, M. Taix, and R. Murray, "A motion planner for nonholonomic mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 5, pp. 577–593, 1994.

[6] G. Wilfong, "Shortest paths for autonomous vehicles," in *Proceedings, 1989 International Conference on Robotics and Automation*, pp. 15–20 vol.1, 1989.

[7] J.-D. Boissonnat, A. Cérézo, and J. Leblond, "Shortest paths of bounded curvature in the plane," *Journal of Intelligent and Robotic Systems*, vol. 11, no. 1-2, pp. 5–20, 1994.

[8] J. Sellen, "Planning paths of minimal curvature," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1976–1982 vol.2, 1995.

[9] J.-D. Boissonnat and S. Lazard, "A Polynomial-Time Algorithm for Computing a Shortest Path of Bounded Curvature amidst Moderate Obstacles (Extended Abstract)," in *Proceedings of the Twelfth Annual Symposium on Computational Geometry*, SCG '96, (New York, NY, USA), p. 242–251, Association for Computing Machinery, 1996.

[10] S. Fortune and G. Wilfong, "Planning Constrained Motion," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, (New York, NY, USA), p. 445–459, Association for Computing Machinery, 1988.

[11] J. Reif and H. Wang, "The Complexity of the Two Dimensional Curvature-Constrained Shortest-Path Problem," in *Robotics: The Algorithmic Perspective* (M. T. M. Pankaj K. Agarwal, Lydia E. Kavraki, ed.), Third International Workshop on Algorithmic Foundations of Robotics (WAFR98), pp. 49–57, Houston, Texas: A. K. Peters Ltd, 1998.

[12] J. Backer and D. Kirkpatrick, "A Complete Approximation Algorithm for Shortest Bounded-Curvature Paths," in *Algorithms and Computation* (S.-H. Hong, H. Nagamochi, and T. Fukunaga, eds.), (Berlin, Heidelberg), pp. 628–643, Springer Berlin Heidelberg, 2008.

[13] P. K. Agarwal, P. Raghavan, and H. Tamaki, "Motion planning for a steering-constrained robot through moderate obstacles," in *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '95, (New York, NY, USA), p. 343–352, Association for Computing Machinery, 1995.

[14] L. E. Dubins, "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents," *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[15] P. Jacobs and J. Canny, "Planning smooth paths for mobile robots," in *Proceedings, 1989 International Conference on Robotics and Automation*, pp. 2–7 vol.1, 1989.

[16] H. Wang and P. K. Agarwal, "Approximation Algorithms for Curvature-Constrained Shortest Paths," in *Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '96, (USA), p. 409–418, Society for Industrial and Applied Mathematics, 1996.

[17] P. K. Agarwal, T. Biedl, S. Lazard, S. Robbins, S. Suri, and S. Whitesides, "Curvature-Constrained Shortest Paths in a Convex Polygon (Extended Abstract)," in *Proceedings of the Fourteenth Annual Symposium on Computational Geometry*, SCG '98, (New York, NY, USA), p. 392–401, Association for Computing Machinery, 1998.

[18] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[19] J. D. Gammell, T. D. Barfoot, and S. S. Srinivasa, "Batch Informed Trees (BIT*): Informed asymptotically optimal anytime search," *The International Journal of Robotics Research*, vol. 39, no. 5, pp. 543–567, 2020.

[20] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *The International journal of robotics research*, vol. 34, no. 7, pp. 883–921, 2015.

[21] P. Maini and P. B. Sujit, "Path planning for a UAV with kinematic constraints in the presence of polygonal obstacles," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 62–67, 2016.

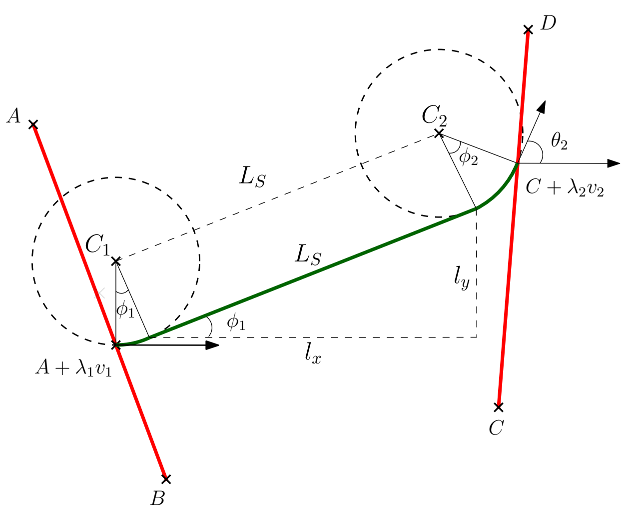
[22] M. Naazare, D. Ramos, J. Wildt, and D. Schulz, "Application of graph-based path planning for uavs to avoid restricted areas," in *2019 IEEE international symposium on safety, security, and rescue robotics (SSRR)*, pp. 139–144, IEEE, 2019.

[23] S. G. Manyam and S. Rathinam, "On tightly bounding the Dubins traveling salesman's optimum," *Journal of Dynamic Systems, Measurement and Control*, vol. 140, no. 7, p. 071013, 2018.
[24] P. Vána and J. Faigl, "Optimal Solution of the Generalized Dubins Interval Problem," in *Robotics: Science and Systems*, 2018.
[25] S. G. Manyam, S. Rathinam, D. Casbeer, and E. Garcia, "Tightly bounding the shortest Dubins paths through a sequence of points," *Journal of Intelligent & Robotic Systems*, vol. 88, no. 2, pp. 495–511, 2017.
[26] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.
[27] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, pp. 72–82, December 2012. https://ompl.kavrakilab.org.

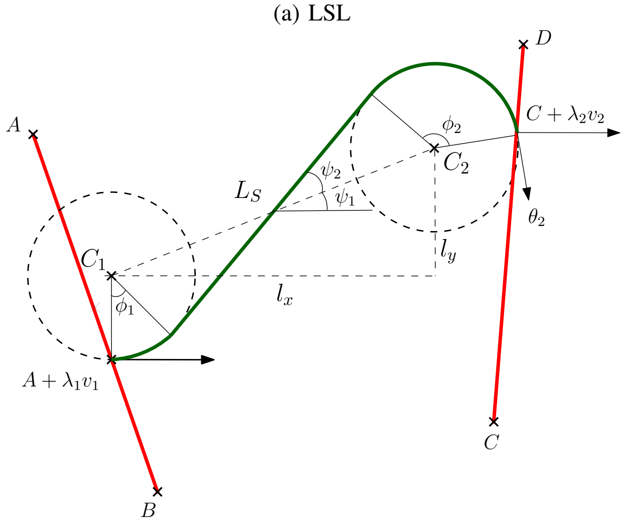
APPENDIX

Notation: $\mu(S) = 1$, if S is true, $\mu(S) = -1$, if S is false.

A. Proof of Lemma 1



(a) LSL



(b) LSR

Fig. 16: Dubins CSC paths

Proof: We only prove this lemma for the LSL and LSR paths. Due to the symmetry, the proofs for RSR and RSL paths follow similarly. Without loss of generality, we assume the initial heading is 0 with respect to $x-$axis. Let the vectors v_1 and v_2 be defined as the following (refer to Fig. 16): $v^1 := B - A$ and $v^2 := D - C$. Let C_1 and C_2 be the centers of the first and last segments in the CSC path.

Case LSL: For an LSL path, the centers are given as $C_1 = (A_x + \lambda_1 v_x^1, A_y + \lambda_1 v_y^1 + \rho)$ and $C_2 = (C_x + \lambda_2 v_x^2 - \rho \sin\theta_2,\ C_y + \lambda_2 v_y^2 + \rho\cos\theta_2)$. Let $l_x(\lambda)$ and $l_y(\lambda)$ denote the projections of the S segment in the LSL path along the x-axis and y-axis respectively (Fig. 16a). Note that $l_x(\lambda_1, \lambda_2) = A_x + \lambda_1 v_x^1 - C_x - \lambda_2 v_x^2 + \rho\sin\theta_2$ and $l_y(\lambda_1, \lambda_2) = A_y + \lambda_1 v_y^1 + \rho - C_y - \lambda_2 v_y^2 - \rho\cos\theta_2$[10]. The length of the S segment is given as $l_S(\lambda_1, \lambda_2) = \sqrt{l_x^2 + l_y^2}$.

Now, $l_{LSL}(\lambda_1, \lambda_2) = l_S(\lambda) + \rho(\phi_1(\lambda_1, \lambda_2) + \phi_2(\lambda_1, \lambda_2))$. Since $\phi_1(\lambda_1, \lambda_2) + \phi_2(\lambda_1, \lambda_2) = \theta_2$, $l_{LSL} = l_S(\lambda_1, \lambda_2) + \rho\theta_2$. Therefore, the minimum of l_{LSL} for $\lambda_1, \lambda_2 \in [0, 1]$ may occur at the boundary points or at a local minimum where $\frac{d}{d\lambda_i} l_{LSL}(\lambda_1, \lambda_2) = \frac{d}{d\lambda_i} l_S(\lambda_1, \lambda_2) = 0$. Differentiating l_S with respect to λ_i and simplifying the resulting expression, we get,

$$\frac{d}{d\lambda_i} l_s = \frac{1}{l_s}\left(v_i^x l_x + v_i^y l_y\right),$$

Therefore, $\frac{d}{d\lambda_i} l_S = 0$ implies that $v_i^x l_x + v_i^y l_y = 0$, *i.e.*, the straight line segment in the LSL path is perpendicular to $\overline{AB}$ for $i = 1$, or the straight line segment in the LSL path is perpendicular to $\overline{CD}$ for $i = 2$.

Case LSR: The centers corresponding to the L and the R segments of the LSR path are given as follows: $C_1 = (A_x + \lambda_1 v_x^1, A_y + \lambda_1 v_y^1 + \rho)$ and $C_2 = (C_x + \lambda_2 v_x^2 + \rho\sin\theta_2,\ C_y + \lambda_2 v_y^2 - \rho\cos\theta_2)$. The quantities l_x and l_y, shown in Fig. 16b, are defined as, $l_x := A_x - C_x + \lambda_1 v_x^1 - \lambda_2 v_x^2 - \rho\sin\theta_2$ and $l_y := A_y - C_y + \lambda_1 v_y^1 - \lambda_2 v_y^2 + \rho + \rho\cos\theta_2$. The length of the straight line segment, $l_S = \sqrt{l_x^2 + l_y^2 - 4\rho^2}$. The quantities ψ_1 and ψ_2, shown in in Fig. 16b, are given as following: $\psi_1 = \arctan(\frac{l_y}{l_x})$ and $\psi_2 = \arctan(\frac{2\rho}{l_S})$. Since $\phi_1 + \phi_2 = 2(\psi_1 + \psi_2) - \theta_2$, the derivative of the length of the path is given as below,

$$\frac{\partial}{\partial\lambda_i} l_{LSR} = \frac{\partial}{\partial\lambda_i} l_S + 2\rho\frac{\partial}{\partial\lambda_i}(\psi_1 + \psi_2).$$

Differentiating l_S, ψ_1 and ψ_2 with respect to λ_i, we get

$$\frac{\partial}{\partial\lambda_i} l_S = \mu(i = 1)(l_x v_x^i + l_y v_y^i), \tag{1}$$

$$\frac{\partial}{\partial\lambda_i} \psi_1 = \mu(i = 1)\frac{l_x v_y^i - l_y v_x^i}{l_x^2 + l_y^2}, \tag{2}$$

$$\frac{\partial}{\partial\lambda_i} \psi_2 = -\mu(i = 1)\frac{2\rho(l_x v_x^i + l_y v_y^i)}{(l_x^2 + l_y^2)l_S}. \tag{3}$$

This derivative of the length l_{LSR} is obtained as

$$\frac{\partial}{\partial\lambda_i} l_{LSR} = \mu(i = 1)\left[v_x^i\cos\phi_1 + v_y^i\sin\phi_1\right]. \tag{4}$$

[10]For simplicity, in some places, we write l_x, l_y instead of $l_x(\lambda_1, \lambda_2), l_y(\lambda_1, \lambda_2)$.

Clearly, $\frac{\partial}{\partial \lambda_i} l_{LSR} = 0$ when the straight line segment in the LSR path is perpendicular to $\overline{AB}$ for $i = 1$, or the straight line segment in the LSR path is perpendicular to $\overline{CD}$ for $i = 2$. ■

B. Proof of Lemma 2

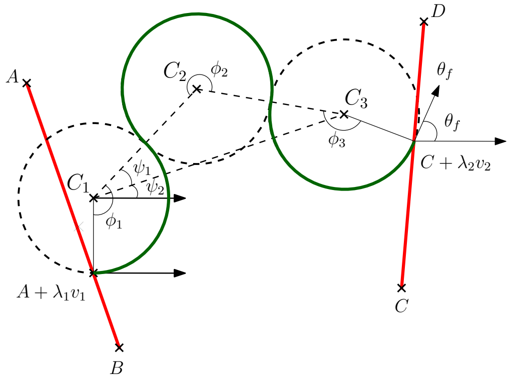


Fig. 17: Dubins LRL path

Proof: We prove this result for the LRL path, and the proof for the RLR path follows similarly, due to symmetry. The minimum of l_{LRL} with respect to λ_1 or λ_2 should occur at a local minima or at the boundary points. We show the local extrema is always a maximum. Without loss of generality, we assume the starting heading as 0. The centers C_1 and C_3 of the L segments in the LRL path (refer to Fig. 17) are $(A_x + \lambda_1 v_x^1, A_y + \lambda_1 v_y^1 + \rho)$ and $(C_x + \lambda_2 v_x^2 - \rho \sin\theta_2, C_y + \lambda_2 v_y^2 + \rho\cos\theta_2)$. Let l_x and l_y be the projections of $\overline{C_1 C_3}$ on $x-$axis and $y-$axis respectively, and are given as $l_x = A_x + \lambda_1 v_x^1 - C_x - \lambda_2 v_x^2 + \rho\sin\theta_2$ and $l_y = A_y + \lambda_1 v_y^1 + \rho - C_y - \lambda_2 v_y^2 - \rho\cos\theta_2$. The length of $\overline{C_1 C_3}$, $l_{cc} := \sqrt{l_x^2 + l_y^2}$. We know that $\phi_1 + \phi_2 + \phi_3 = \theta_2 - \theta_1 + 2\phi_2$, and $\phi_2 = 2\psi_1 + \pi$. The length of the path, $l_{LRL} = \rho(4\psi_1 + 2\pi + \theta_2)$, and its derivative, $\frac{\partial}{\partial\lambda_i} l_{LRL} = 4\rho \frac{\partial}{\partial\lambda_i}\psi_1$. The quantity ψ_1 is given by $\arccos(\frac{l_{cc}}{4\rho})$, where $l_{cc} = \sqrt{l_x^2 + l_y^2}$, and thus we get the derivatives of l_{LRL} as the following:

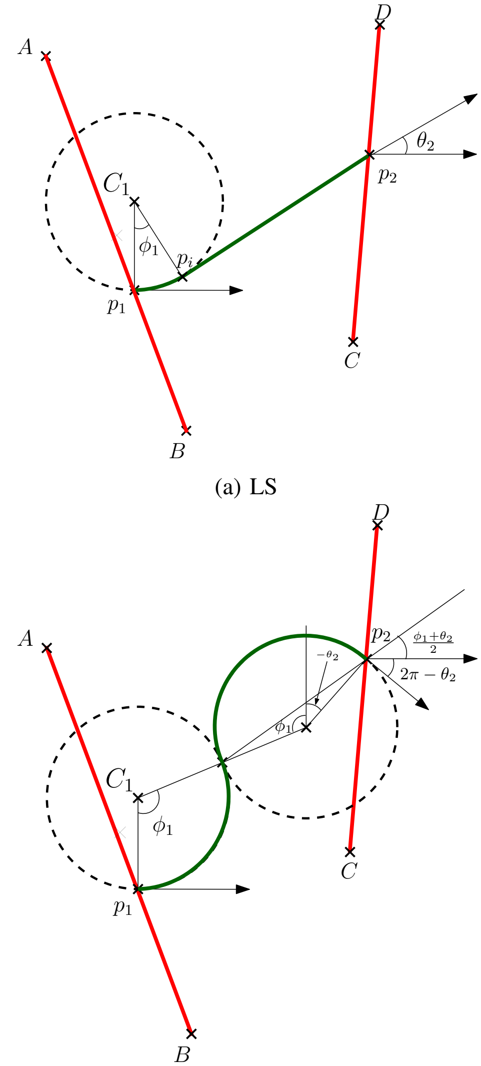
$$
\begin{aligned}
\frac{\partial}{\partial\lambda_i} l_{LRL} =& -\frac{4\rho}{\sqrt{16\rho^2 - l_{cc}^2}}\frac{1}{l_{cc}}(v_x^i l_x + v_y^i l_y),\\
\frac{\partial^2}{\partial\lambda_i^2} l_{LRL} =& \frac{\partial}{\partial\lambda_i}\left(-\frac{4\rho}{l_{cc}\sqrt{16\rho^2 - l_{cc}^2}}\right)(v_x^i l_x + v_y^i l_y)\\
& - \frac{4\rho}{l_{cc}\sqrt{16\rho^2 - l_{cc}^2}}({v_x^i}^2 + {v_y^i}^2).
\end{aligned}
$$

At the local extrema $v_x^i l_x + v_y^i l_y = 0$, and therefore $\frac{\partial^2}{\partial\lambda^2} l_{LRL} = -\frac{4\rho}{l_{cc}\sqrt{16\rho^2 - l_{cc}^2}}({v_x^i}^2 + {v_y^i}^2) < 0$; *i.e.*, the local extremum is always a maximum.

■

C. Proof of Lemma 3

We prove this lemma for the LS and LR paths, and the proofs for RS and RL paths follows similarly.



(a) LS

(b) LR

Fig. 18: Dubins two-segment paths with initial heading given, the final heading depends on the initial and final positions.

Case LS: Without loss of generality, we assume θ_1 is 0. The final heading is a function of λ_1 and λ_2, and it cannot be chosen independently. The start and end points of the LS path (refer to Fig. 18a) are $p_1 = A + \lambda_1 v^1$ and $p_2 = C + \lambda_2 v^2$, respectively. We get the following equations from the projections of $\overline{p_1 p_2}$ on x and y axes:

$$
\begin{aligned}
\rho\sin\phi_1 + l_S\cos\phi_1 &= C_x + \lambda_2 v_x^2 - A_x - \lambda_1 v_x^1,\\
\rho - \rho\cos\phi_1 + l_S\sin\phi_1 &= C_y + \lambda_2 v_y^2 - A_y - \lambda_1 v_y^1.
\end{aligned}
$$

Differentiating with respect to λ_i, we get,

$$
\begin{aligned}
\rho\cos\phi_1\frac{\partial\phi_1}{\partial\lambda_i} + \frac{\partial l_S}{\partial\lambda_i}\cos\phi_1 - l_S\sin\phi_1\frac{\partial\phi_1}{\partial\lambda_i} &= \mu(i=2)v_x^i,\\
\rho\sin\phi_1\frac{\partial\phi_1}{\partial\lambda_i} + \frac{\partial l_S}{\partial\lambda_i}\sin\phi_1 + l_S\cos\phi_1\frac{\partial\phi_1}{\partial\lambda_i} &= \mu(i=2)v_y^i.
\end{aligned}
$$

Using the above equations, we get $\frac{\partial l_{LS}}{\partial \lambda_i} = \rho \frac{\partial \phi_1}{\partial \lambda_i} + \frac{\partial l_S}{\partial \lambda_i} = v_x^i \cos\phi_1 + v_y^i \sin\phi_1$. At the local minima, $v_x^i \cos\phi_1 + v_y^i \sin\phi_1 = 0$, which implies the straight line segment in the LS path is perpendicular to $\overline{AB}$ for $i = 1$ or the straight line segment in the LS path is perpendicular to $\overline{CD}$ for $i = 2$.

Case LR: The initial and final points are defined similar to the LS path. We get the following equations from the projections of $\overline{p_1 p_2}$:

$$\begin{aligned} 2\rho\cos(\frac{\phi_1 - \pi}{2}) + \rho\cos(\frac{\pi}{2} + \theta_2) =& C_x + \lambda_2 v_x^2 - A_x - \lambda_1 v_x^1, \\ 2\rho\sin(\phi_1 - \frac{\pi}{2}) + \rho\sin(\frac{\pi}{2} + \theta_2) =& C_y + \lambda_2 v_y^2 - A_y - \lambda_1 v_y^1 \\ & - \rho. \end{aligned}$$

Differentiating the above with respect to λ_i, we get,

$$\begin{aligned} 2\rho\cos\phi_1 \frac{\partial \phi_1}{\partial \lambda_i} - \rho\cos\theta_2 \frac{\partial \theta_2}{\partial \lambda_i} = \mu(i=2) v_x^i, \\ 2\rho\sin\phi_1 \frac{\partial \phi_1}{\partial \lambda_i} - \rho\sin\theta_2 \frac{\partial \theta_2}{\partial \lambda_i} = \mu(i=2) v_y^i. \end{aligned}$$

The length of the LR paths is $l_{LR} = \rho(\phi_1 + \phi_2) = \rho(2\phi_1 - \theta_2)$. Using the above equations, we obtain the partial derivative of l_{LR} as given below,

$$\begin{aligned} \frac{\partial l_{LR}}{\partial \lambda_i} &= \mu(i=2) \frac{v_x^i \sin\theta_2 - v_y^i \cos\theta_2 - v_x^i \sin\phi_1 + v_y^i \cos\phi_1}{\sin(\theta_2 - \phi_1)}, \\ &= 2\mu(i=2) \frac{\sin(\frac{\theta_2 - \phi_1}{2}) \left[v_x^i \cos(\frac{\theta_2 + \phi_1}{2}) + v_y^i \sin(\frac{\theta_2 + \phi_1}{2}) \right]}{\sin(\theta_2 - \phi_1)}. \end{aligned}$$

At the extremum, $\frac{\partial l_{LR}}{\partial \lambda_i} = 0$. This could happen if $\theta_2 = \phi_1$, which essentially means the second arc in LR path vanishes, and therefore is a degenerate case. Therefore, $v_x^i \cos(\frac{\theta_2 + \phi_1}{2}) + v_y^i \sin(\frac{\theta_2 + \phi_1}{2}) = 0$, implying that $\overline{p_1 p_2}$ is perpendicular to $\overline{AB}$ for $i = 1$, or $\overline{p_1 p_2}$ is perpendicular to $\overline{CD}$ for $i = 2$.

D. Proof of Lemma 4

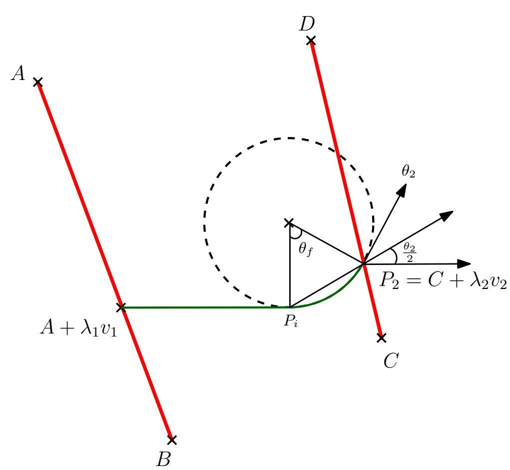


Fig. 19: Dubins two segment path SL.

The path SL with the final heading given is a reflection of the path LS with the initial heading given, and therefore the local extrema is similar to that of the LS path, which occurs when the straight line segment is perpendicular to $\overline{AB}$ or $\overline{CD}$.