Reparametrization of 3D CSC Dubins Paths Enabling 2D Search

Ling Xu[1], Yuliy Baryshnikov[2], and Cynthia Sung[1]

[1] General Robotics, Automation, Sensing & Perception Lab,
University of Pennsylvania, Philadelphia, PA, USA
{xu5,crsung}@seas.upenn.edu
[2] Department of Mathematics
University of Illinois Urbana-Champaign, Urbana, IL, USA
ymb@illinois.edu

Abstract. This paper addresses the Dubins path planning problem for vehicles in 3D space. In particular, we consider the problem of computing CSC paths – paths that consist of a circular arc (C) followed by a straight segment (S) followed by a circular arc (C). These paths are useful for vehicles such as fixed-wing aircraft and underwater submersibles that are subject to lower bounds on turn radius. We present a new parameterization that reduces the 3D CSC planning problem to a search over 2 variables, thus lowering search complexity, while also providing gradients that assist that search. We use these equations with a numerical solver to explore numbers and types of solutions computed for a variety of planar and 3D scenarios. Our method successfully computes CSC paths for the large majority of test cases, indicating that it could be useful for future generation of robust, efficient curvature-constrained trajectories.

Keywords: Dubins vehicle · 3D path planning · numerical solver

1 Introduction

Aircraft and underwater vehicles are often subject to control constraints, limiting their ability to make sharp turns. These vehicles require robust planning algorithms to successfully navigate their environments, avoid obstacles, and reach their goal destinations. The Dubins vehicle model (where vehicles have a lower bound on turn radius) is a well-studied model for car-like vehicles in 2D. A natural extension is to apply these models to 3D vehicles.Dubins paths enhance the efficiency and safety of robotic systems in areas including logistics [14], surveillance [9], search and rescue operations [16], and even in robot design [5, 10]. Thus, understanding and optimizing Dubins paths hold immense practical importance in modern robotics research and applications. However, it is challenging to accurately classify the solution space for the Dubins problem in 3D.

Background on the 3D Dubins Path Planning Problem In this paper, we investigate a solution method for computing 3D Dubins paths of the CSC

type. Consider a vehicle moving in 3D space. Let $\mathbf{x} \in \mathbb{R}^3$ be the position of the vehicle and $R \in SO(3)$ be its orientation. Let $\mathbf{v} = \dot{\mathbf{x}} \in \mathbb{R}^3$ and $\boldsymbol{\omega} \in \mathbb{R}^3$ denote the vehicle's linear velocity and angular velocity, respectively.

For a fixed-wing aircraft [1,7,13,15,18,24], underwater drone or submersible [7, 27] performing low-dynamic maneuvers and simple navigation and waypoint route-planning, a model of the vehicle's motion is the 3D Dubins model:

$$\begin{cases} \dot{\mathbf{v}} = \boldsymbol{\omega} \times \mathbf{v} \\ \boldsymbol{\omega} = \mathbf{u}, \|\mathbf{u}\| \leq u_{max}. \end{cases} \tag{1}$$

That is to say, the vehicle moves at constant forward speed, and the control input $\mathbf{u}$ to the system is the angular velocity only, which is subject to a upper bound in magnitude u_{max}. Both the system's forward speed $\|\mathbf{v}\|$ and u_{max} are dictated by the particular system, and, without loss of generality, it is often assumed that $\|\mathbf{v}\| = 1$. This is a full 3D path-planning problem, where other works [3, 6, 11, 17, 19, 23, 25, 26] address a different mathematical problem by modeling fixed-wing unmanned aerial systems through decoupling 3D path-planning into a 2D problem and elevation with a climb rate constraint. Note that for our simplified 3D model, the full orientation R of the vehicle is not important since the angular velocity $\boldsymbol{\omega}$ can be specified arbitrarily as long as it satisfies the upper bound constraint.[3] Thus, the system's current state can be fully characterized by $\mathbf{x}$ and $\hat{\mathbf{v}}$, where $\hat{\cdot}$ indicates the normalized vector.

Let $\mathbf{x}_i, \hat{\mathbf{v}}_i$ be the starting position and orientation of a 3D vehicle, and $\mathbf{x}_f, \hat{\mathbf{v}}_f$ its goal position and orientation. The *Dubins path planning problem* is:

Problem 1 (3D Dubins Planning). Given an initial configuration $(\mathbf{x}_i, \hat{\mathbf{v}}_i)$, a final configuration $(\mathbf{x}_f, \hat{\mathbf{v}}_f)$, and a control input upper bound u_{max}, find the shortest-length path between points $\mathbf{x}_i$ and $\mathbf{x}_f$ such that the initial and final directions are given by $\hat{\mathbf{v}}_i$ and $\hat{\mathbf{v}}_f$, respectively, subject to Eq. (1).

Since the vehicle velocity is constant, this problem can be equivalently defined as finding a shortest length path with a curvature constraint:

Problem 2. Given an initial configuration $(\mathbf{x}_i, \hat{\mathbf{v}}_i)$, a final configuration $(\mathbf{x}_f, \hat{\mathbf{v}}_f)$, and a control input upper bound u_{max}, find the shortest-length path between points $\mathbf{x}_i$ and $\mathbf{x}_f$ such that the initial and final directions are given by $\hat{\mathbf{v}}_i$ and $\hat{\mathbf{v}}_f$ and the path curvature does not exceed $1/u_{max}$.

This model is an extension to the 2D classical Dubins model first introduced in 1957 in [8] for a car-like vehicle that can only move forward in the plane at constant speed with bounded minimum turn radius, and various approaches exist to find optimal paths. One common method involves the application of

[3] In a real vehicle, the elements of $\boldsymbol{\omega}$ would be subject to constraints from the vehicle design (e.g, the propeller configuration of an underwater vehicle) and the full orientation R would be required to relate each control input to $\boldsymbol{\omega}$. We make the assumption that vehicle is equally able to turn in any direction for simplicity. Other works [6, 11, 17, 23, 25–27] have addressed models where directionality matters.

the Pontryagin's Maximum Principle, which provides necessary conditions for an optimal control problem [20]. This principle implies that the optimal trajectories solving this problem are of bang-bang type (arcs of the maximal possible curvature) connected, possibly, by sliding trajectories. A careful analysis reveals that the sliding trajectories need to be straight lines [4,21]. Further, the number of fragments is at most three. Dubins [8] proved that optimal trajectories must necessarily fall into one of two forms: CSC or CCC, where C denotes a circular arc of radius $1/u_{max}$ and S a straight line segment. For any initial and final configuration, the Dubins set is defined to be the six admissible paths LSL, RSR, RSL, LSR, RLR, and LRL, with L denoting a left turn and R a right turn.

When the problem is extended to 3D, trajectories must still be of bang-bang type, but the optimal trajectories now consist of helicoidal trajectories (spiraling around a cylinder with constant gain), including a special case when the gain is zero, i.e., the trajectory is planar (an arc). As in the planar case, sliding regimes are possible and, again, can be shown to consist of straight lines. One can ask whether the helicoidal trajectories are truly necessary. To this end, [22] showed that there exist pairs of initial and final configurations such that helicoidal trajectories are strictly shorter than any CSC or CCC paths, and [25] constructed these 3D helical paths. For many practical purposes, the CSC or CCC paths are far more desirable since they are easier to parameterize geometrically and easier to implement in practice, as each of the fragments is planar.

In this paper, we focus on an efficient algorithm for computing CSC trajectories connecting two 3D configurations and satisfying Eq. 1:

Problem 3 (3D CSC Paths). Given an initial configuration $(\mathbf{x}_i, \hat{\mathbf{v}}_i)$, a final configuration $(\mathbf{x}_f, \hat{\mathbf{v}}_f)$, and a minimum radius of curvature r, find valid CSC paths between points $\mathbf{x}_i$ and $\mathbf{x}_f$ such that the initial and final directions are given by $\hat{\mathbf{v}}_i$ and $\hat{\mathbf{v}}_f$ and the path curvature does not exceed $1/r$.

Existing approaches often face limitations that restrict their generality. Hota and Ghose [13] provide a geometric formulation for finding 3D CSC Dubins paths that requires numerically solving a nonlinear system of equations with five unknown variables, but the solution only applies when initial and final points are sufficiently far apart. Further, they did analyze how the number of solutions vary with the 3D configuration space. Some 3D path planning algorithms rely on optimal control methods [1,6,24]. While they are computationally tractable, they do not admit an analytical solution and are only resolution complete. Other numerical methods involving iterative procedures such as optimization for a nonlinear programming formulation [11] and RRT [17] can fail to converge to the optimal solution for certain 3D cases and require exhaustive search. Recent work in [2] computes up to 7 valid 3D CSC Dubins paths, but the formulation of the Dubins problem as a 6R manipulator and solving for the analytical solutions of the inverse-kinematics does not address applicability to real-time path-planning that gradient-based numerical solvers enable.

Contributions We present a parametrization of the 3D CSC Dubins problem that simplifies computation of an optimal path, reducing the problem to a nu-

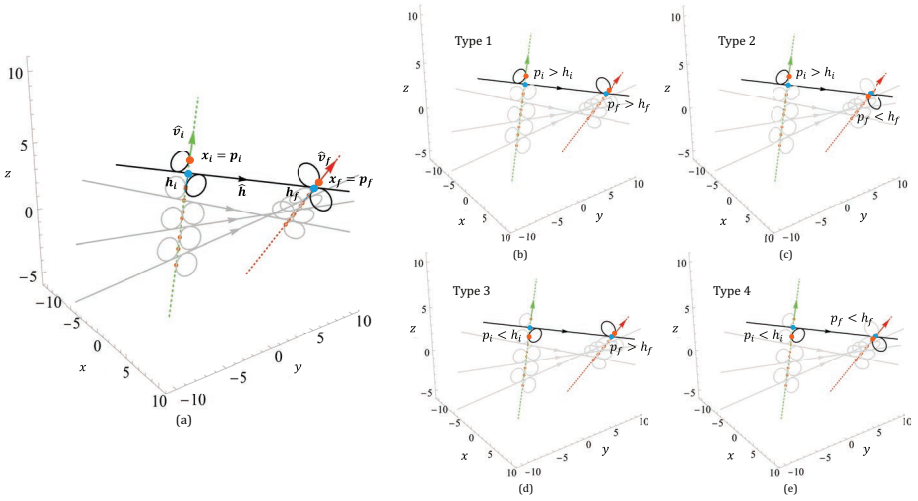

Fig. 1: **Parametrization of 3D CSC Paths** **(a)** A straight line in the $\hat{\mathbf{h}}$ direction intersects with lines through the starting and ending configurations, resulting in two possibilities for the starting point $\mathbf{p}_i$ and two possibilities for ending point $\mathbf{p}_f$. **(b-e)** Type 1-4 starting and ending circles.

merical search over two variables, h_i and h_f. We ignore for now the problem of CCC paths, leaving that for future investigation.

For CSC paths, we develop closed-form equations (Section 2) that describe the desired initial and final configurations as a function of h_i and h_f, thereby allowing one to take gradients of the equations with respect to these parameters. This gradient can assist a numerical solver when the paths are more geometrically complex. We characterize the solution space of 3D CSC Dubins paths in both planar and non-planar cases (Section 3), and we show how the new formulation improves the detection of 3D CSC Dubins path solutions compared to an existing work [12].

2 Parametrization of the 3D Dubins Planning Problem

To start, we observe that regardless the orientations of the initial and final configurations, the initial CS component of the path is planar, and so is the final SC component of the path. Further, if the line along which the S component lies is known, then the planes containing the two C components are completely defined. Let us introduce two new points

$$\mathbf{h}_i = \mathbf{x}_i + h_i \hat{\mathbf{v}}_i, \quad \mathbf{h}_f = \mathbf{x}_f + h_f \hat{\mathbf{v}}_f, \tag{2}$$

which lie on the lines defined by the initial and final configurations, respectively. Assume that the S component of the CSC path lies on the line from $\mathbf{h}_i$ to $\mathbf{h}_f$.

We define $\hat{\mathbf{h}}$ to be the normalized vector corresponding to $\mathbf{h}_f - \mathbf{h}_i$. The initial C component of the path must lie in the plane defined by $\mathbf{x}_i$ and the vectors $\hat{\mathbf{v}}_i$ and $\hat{\mathbf{h}}$, and similarly for the final C component. Figure 1 shows in gray lines how this plane and the resulting C components change with h_i and h_f.

Consider the initial C component. There are two circles of radius r that lie tangent both to the line defined by $\mathbf{x}_i$ and $\hat{\mathbf{v}}_i$ and to the line defined by $\mathbf{h}_i$ and $\mathbf{h}_f$ and correctly oriented for the vehicle to transition from an initial orientation of $\hat{\mathbf{v}}_i$ to a straight-line orientation of $\hat{\mathbf{h}}$. The centers of these circles are

$$\mathbf{c}_i = \mathbf{h}_i \pm \frac{r}{\left\|\hat{\mathbf{v}}_i \times \hat{\mathbf{h}}\right\|} \left(\hat{\mathbf{v}}_i - \hat{\mathbf{h}}\right) \tag{3}$$

We will use $\mathbf{c}_{i1}$ to indicate the $+$ solution and $\mathbf{c}_{i2}$ to indicate the $-$ solution. The point of tangency between the circle and the line defined by $(\mathbf{x}_i, \hat{\mathbf{v}}_i)$ can then be computed as the projection of $\mathbf{c}_i$ onto the line:

$$\mathbf{p}_i = \mathbf{x}_i + p_i \hat{\mathbf{v}}_i, \quad p_i = h_i \pm \frac{r}{\left\|\hat{\mathbf{v}}_i \times \hat{\mathbf{h}}\right\|} \left(1 - \hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_i\right) \tag{4}$$

Again, we will use p_{i1} (and equivalently, $\mathbf{p}_{i1}$) to indicate the $+$ solution and p_{i2} ($\mathbf{p}_{i2}$) to indicate the $-$ solution. The same analysis can be conducted for the final C component, where the center is defined as $\mathbf{c}_f$, resulting in:

$$\mathbf{p}_f = \mathbf{x}_f + p_f \hat{\mathbf{v}}_f, \quad p_f = h_f \pm \frac{r}{\left\|\hat{\mathbf{v}}_f \times \hat{\mathbf{h}}\right\|} \left(1 - \hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_f\right) \tag{5}$$

The points $\mathbf{h}_i$ and $\mathbf{h}_f$ correctly identify the desired CSC path if one of $\mathbf{p}_{i1}$ or $\mathbf{p}_{i2}$ is $\mathbf{x}_i$ and one of $\mathbf{p}_{f1}$ or $\mathbf{p}_{f2}$ is $\mathbf{x}_f$ such that the scalar quantities satisfy:

$$\begin{cases} p_{i1} = 0 & \text{or} \quad p_{i2} = 0 \\ p_{f1} = 0 & \text{or} \quad p_{f2} = 0 \end{cases} \tag{6}$$

We categorize these solution types in Table 1 with the geometries shown in Fig. 1.

Path Directionality While the given equations ensure that the directions of circular arc components are consistent with the directions of the given $\hat{\mathbf{h}}$ vector, it is possible for the intersection between the starting circle centered around $\mathbf{c}_i$ and the line from $\mathbf{h}_i$ to $\mathbf{h}_f$ to occur "before" the intersection between the ending circle centered around $\mathbf{c}_f$ and the line. For this situation, the vehicle would

Table 1: Solution Types

Regular

	$p_{i1} = 0$	$p_{i2} = 0$
$p_{f1} = 0$	Type 1	Type 3
$p_{f2} = 0$	Type 2	Type 4

Switched

	$p^*_{i1} = 0$	$p^*_{i2} = 0$
$p^*_{f1} = 0$	Type 5	Type 7
$p^*_{f2} = 0$	Type 6	Type 8

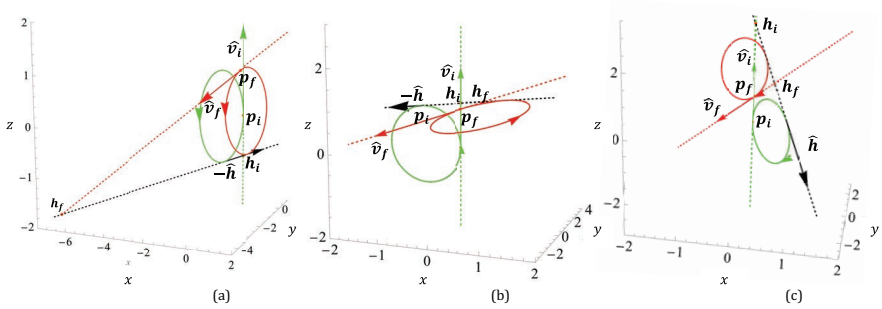

Fig. 2: **Path Directionality (a)** Example of a "switched" solution where the straight line path is traveling along the $-\hat{\mathbf{h}}$ direction. **(b)** Example of an invalid switched-type computed path in 3D where the $-\hat{\mathbf{h}}$ direction travels away from the ending point. **(c)** Example of a an invalid regular-type computed path in 3D where the $\hat{\mathbf{h}}$ direction travels away from the ending point.

have to travel in the reverse direction along the straight line component, from $\mathbf{h}_f$ to $\mathbf{h}_i$. We therefore consider four additional solution types, corresponding to "switched" situations in which the vehicle must travel along the straight line segment in the reverse direction. These paths can be computed using the same equations as for the regular case, except replacing $\hat{\mathbf{h}}$ with $-\hat{\mathbf{h}}$. That is,

$$p_i^* = h_i \pm \frac{r}{\left\|\hat{\mathbf{v}}_i \times -\hat{\mathbf{h}}\right\|}\left(1 + \hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_i\right), \qquad \mathbf{p}_i^* = \mathbf{x}_i + p_i \hat{\mathbf{v}}_i \tag{7}$$

$$p_f^* = h_f \pm \frac{r}{\left\|\hat{\mathbf{v}}_f \times -\hat{\mathbf{h}}\right\|}\left(1 + \hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_f\right), \qquad \mathbf{p}_f^* = \mathbf{x}_f + p_f \hat{\mathbf{v}}_f. \tag{8}$$

We still use p_{i1}^* (and equivalently, $\mathbf{p}_{i1}^*$) to indicate the $+$ solution and p_{i2}^* ($\mathbf{p}_{i2}^*$) to indicate the $-$ solution, and similarly for p_f^* and $\mathbf{p}_f^*$.

Similarly to the regular situation, when the scalar quantities satisfy

$$\begin{cases} p_{i1}^* = 0 & \text{or} \quad p_{i2}^* = 0 \\ p_{f1}^* = 0 & \text{or} \quad p_{f2}^* = 0 \end{cases} \tag{9}$$

we have the desired CSC path. We categorize these solution types in Table 1.

Full Solution Set Thus, for any given initial and final configuration, there are 8 possible solution types. It is not necessarily the case that every solution type yields a solution, and in some cases, there are multiple unique paths of one solution type.

Additionally, it is possible for the solution for h_i and h_f to be invalid even if the equations are satisfied. In particular, the direction of the straight line path, either $\hat{\mathbf{h}}$ or $-\hat{\mathbf{h}}$, may be in the opposite direction of the desired ending point. Invalid paths can be identified as follows:

1. Regular solutions invalid when $(\mathbf{c}_f - \mathbf{c}_i) \cdot \hat{\mathbf{h}} < 0$ as shown in Fig. 2(b).
2. Switched solutions invalid when $(\mathbf{c}_f - \mathbf{c}_i) \cdot \hat{\mathbf{h}} > 0$ as shown in Fig. 2(c).

2.1 Extracting the CSC Path

The components of the resulting CSC path can be extracted from h_i and h_f. For the initial C component, the plane on which the arc lies is defined by the vectors $\hat{\mathbf{v}}_i$ and $\hat{\mathbf{h}}$. The center of the circular arc can be computed according to Eq. 3. The arc length θ_i is given by

$$\theta_i = \begin{cases} \cos^{-1}\left(\hat{\mathbf{v}}_i \cdot \hat{\mathbf{h}}\right), & p_i < h_i \\ 2\pi - \cos^{-1}\left(\hat{\mathbf{v}}_i \cdot \hat{\mathbf{h}}\right), & p_i > h_i \end{cases} \tag{10}$$

These same equations also hold for the final circular arc. The S component consists of the line between $\mathbf{h}_i$ and $\mathbf{h}_f$ where it intersects with the initial and final arc. For the initial arc, the intersection $\mathbf{d}_i \in \mathbb{R}^3$ can be found by projecting the center of the circle onto the vector $\hat{\mathbf{h}}$:

$$\mathbf{d}_i = \left[(\mathbf{c}_i - \mathbf{h}_i) \cdot \hat{\mathbf{h}}\right] \hat{\mathbf{h}} + \mathbf{h}_i \tag{11}$$

and similarly for the final arc. Then the S component is the straight line path from $\mathbf{d}_i$ to $\mathbf{d}_f$. This segment is in the $\hat{\mathbf{h}}$ direction for regular type paths and in the $-\hat{\mathbf{h}}$ direction for switched type paths. The two arcs together with the straight line form the CSC path.

2.2 Gradients

The previous subsections have demonstrated that it is possible to solve for CSC Dubins paths as a system of two equations on two scalar variables h_i an h_f. These closed-form equations admit gradients, which allow us to solve them more efficiently using numerical solvers.

These derivatives take the following form:

$$\frac{\partial p_i}{\partial h_i} = \pm \frac{-r(1 - \hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_i)(\hat{\mathbf{v}}_i \times -\hat{\mathbf{h}}) \cdot (\hat{\mathbf{v}}_i \times -a_i))}{\|\hat{\mathbf{v}}_i \times -\hat{\mathbf{h}}\|^3} - \frac{r(a_i \cdot \hat{\mathbf{v}}_i)}{\|\hat{\mathbf{v}}_i \times -\hat{\mathbf{h}}\|} + 1 \tag{12}$$

$$\frac{\partial p_i}{\partial h_f} = \pm \frac{-r(1 - \hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_i)(\hat{\mathbf{v}}_i \times -\hat{\mathbf{h}}) \cdot (\hat{\mathbf{v}}_i \times -a_f)}{\|\hat{\mathbf{v}}_i \times -\hat{\mathbf{h}}\|^3} - \frac{r(a_f \cdot \hat{\mathbf{v}}_i)}{\|\hat{\mathbf{v}}_i \times -\hat{\mathbf{h}}\|} \tag{13}$$

$$\frac{\partial p_f}{\partial h_i} = \pm \frac{-r(1 - \hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_f)(\hat{\mathbf{v}}_f \times -\hat{\mathbf{h}}) \cdot (\hat{\mathbf{v}}_f \times -a_i)}{\|\hat{\mathbf{v}}_f \times -\hat{\mathbf{h}}\|^3} - \frac{r(a_i \cdot \hat{\mathbf{v}}_f)}{\|\hat{\mathbf{v}}_f \times -\hat{\mathbf{h}}\|} + 1 \tag{14}$$

$$\frac{\partial p_f}{\partial h_f} = \pm \frac{-r(1 - \hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_f)(\hat{\mathbf{v}}_f \times -\hat{\mathbf{h}}) \cdot (\hat{\mathbf{v}}_f \times -a_f))}{\|\hat{\mathbf{v}}_f \times -\hat{\mathbf{h}}\|^3} - \frac{r(a_f \cdot \hat{\mathbf{v}}_f)}{\|\hat{\mathbf{v}}_f \times -\hat{\mathbf{h}}\|} \tag{15}$$

where

$$a_i = \frac{d\hat{\mathbf{h}}}{dh_i} = \frac{(\hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_i)\hat{\mathbf{h}} - \hat{\mathbf{v}}_i}{\|\mathbf{h}_f - \mathbf{h}_i\|}, \quad a_f = \frac{d\hat{\mathbf{h}}}{dh_f} = \frac{-(\hat{\mathbf{h}} \cdot \hat{\mathbf{v}}_f)\hat{\mathbf{h}} + \hat{\mathbf{v}}_f}{\|\mathbf{h}_f - \mathbf{h}_i\|}. \tag{16}$$

The + in the equations corresponds to derivatives of p_{i1} or p_{f1} with respect to h_i and h_f, and the $-$ corresponds to derivatives of p_{i2} or p_{f2} with respect to h_i and h_f. Derivatives for switched equations p_i^* and p_f^* are identical except for replacing $\hat{\mathbf{h}}$ with $-\hat{\mathbf{h}}$.

3 Experimental Results

Given initial configuration $\mathbf{x}_i, \hat{\mathbf{v}}_i$ and final configuration $\mathbf{x}_f, \hat{\mathbf{v}}_f$, we set $\mathbf{x}_i = \mathbf{p}_i$ and $\mathbf{x}_f = \mathbf{p}_f$ and solve for h_i and h_f for each solution type. We implement a CSC path solver in Python using `scipy.optimize.fsolve` to solve the equations in Sec. 2. Tests were conducted on an 8-core AMD Ryzen 7 5825U processor with 16 GB RAM. Solutions for each pair of equations (Types 1-8) are computed over the variables h_i and h_f and using the gradients given in Sec. 2.2. We filter out invalid solutions that may stem from either failure of the numerical solver to converge (which may be detected by checking the values of p_i and p_f) or from directional inconsistencies as mentioned in Sec. 2. We test the effectiveness and versatility of these 8 equations in detecting possible CSC paths for both non-planar and planar cases. All of the tests are conducted with $r = 1$.

3.1 Example Solutions

We begin by testing specific planar and non-planar cases to explore the solver's ability to compute different solution types. For both planar and non-planar cases, the difficulty of the problem seems to depend on the proximity of the starting and ending positions.[4].

Planar Cases We begin by examining the planar case for which methods for computing solutions are well-established. When the initial and final points are sufficiently far apart, then all four types of regular solutions exist and the equations for p_i and p_f are fairly well-behaved. Figure 3 shows the solutions for $\mathbf{x}_i \to [0, 0, -\infty]$ and $\mathbf{x}_f = [0, 0, 0]$. Values for h_i and h_f corresponding to $p_i = 0$ are shown in red, and values corresponding to $p_f = 0$ are shown in blue. All of the curves are smooth and approach horizontal ($p_f = 0$) and vertical lines ($p_i = 0$) with straightforward intersections representing candidate solutions as $\mathbf{x}_i \to [0, 0, -\infty]$. This makes sense since as the starting and ending positions move further apart, changes in h_f will have less of an effect on the position of p_i, and similarly for the effect of h_i on p_f. We thus expect that solving for CSC paths when starting and ending positions are far to be straightforward for a standard numerical solver.For example, Figure 4 shows results for one test condition where the starting and ending points are "sufficiently far":

[4] Additional examples can be found in the supplementary materials included in the full paper, submitted on `https://arxiv.org/` titled *Reparametrization of 3D CSC Dubins Paths Enabling 2D Search*

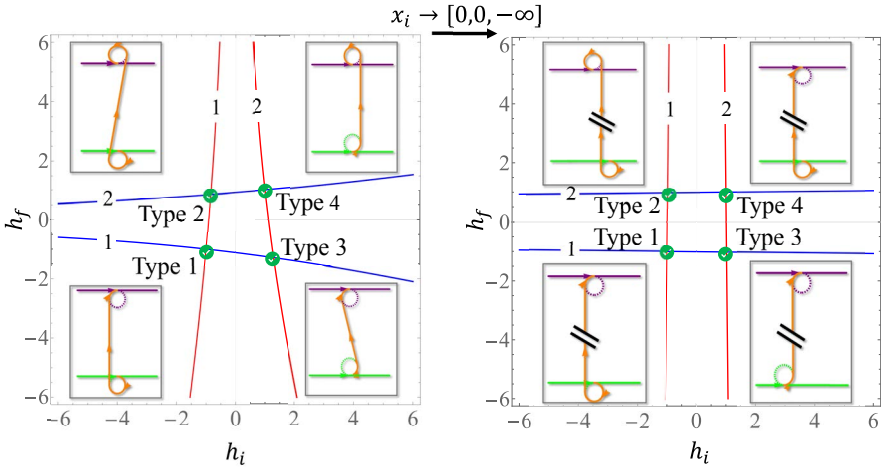

Fig. 3: **Solution plots**: $\mathbf{x}_f = [0,0,0], \hat{\mathbf{v}}_i = [1,0,0] = \hat{\mathbf{v}}_f$. $\mathbf{x}_i = [0,0,-10]$ (left) vs $\mathbf{x}_i = [0,0-100]$ (right). Values for h_i and h_f correspond to $p_i = 0$ (red) and $p_f = 0$ (blue). The (1) annotation indicates p_{i1} and p_{f1} curves and (2) marks p_{i2} and p_{f2} curves. Intersections marked by green checks are valid solutions. Insets in 4 corners show the solution types, with hashes indicating cutting out a portion of the path due to points being far apart. Green and purple lines indicate the starting and ending directions, respectively. The final CSC path is in orange.

$\mathbf{x}_i = [0,0,0], \mathbf{x}_f = [-1,0,3], \hat{\mathbf{v}}_i = [0,0,1], \hat{\mathbf{v}}_f = \frac{1}{\sqrt{2}}[1,0,1]$. Similarly to the previous example, this case produces 4 regular-type solutions, one of each type. The solutions were all found in 0.038 s with fewer than 10 solver iterations for each solution type.

More challenging cases occur when the starting and ending positions are close together. In these cases, not all regular-type solutions are valid, or even exist. Figure 5 shows the results for one such test condition: $\mathbf{x}_i = [0,0,0], \mathbf{x}_f = [0,1.01,1], \hat{\mathbf{v}}_i = [0,0,1], \hat{\mathbf{v}}_f = \frac{1}{\sqrt{17}}[0,1,4]$. The solver is able to successfully find all valid solutions for this case. We begin to see non-regular type solutions represented by intersections of dotted lines on the plot, and there are only 3 valid solutions since there is no straight line path between an initial right-turning arc and a final left-turning arc. One solution of Type 6 was found but was filtered as having an invalid straight segment direction. The solver took 0.146 s to find all of the solutions due to the need to filter out more solutions that either do not exist or are invalid.

Non-Planar Cases The same equations extend to 3D scenarios, showing similar trends for starting and ending positions that are close or far apart. Figure 6 shows the results for one 3D case where the starting and ending positions are sufficiently far apart in distance: $\mathbf{x}_i = [0,0,0], \mathbf{x}_f = [3,0,-1], \hat{\mathbf{v}}_i = [0,0,1], \hat{\mathbf{v}}_2 = \frac{1}{\sqrt{21}}[2,4,1]$. Similarly to the planar tests, this is fairly easy for the solver, pro-

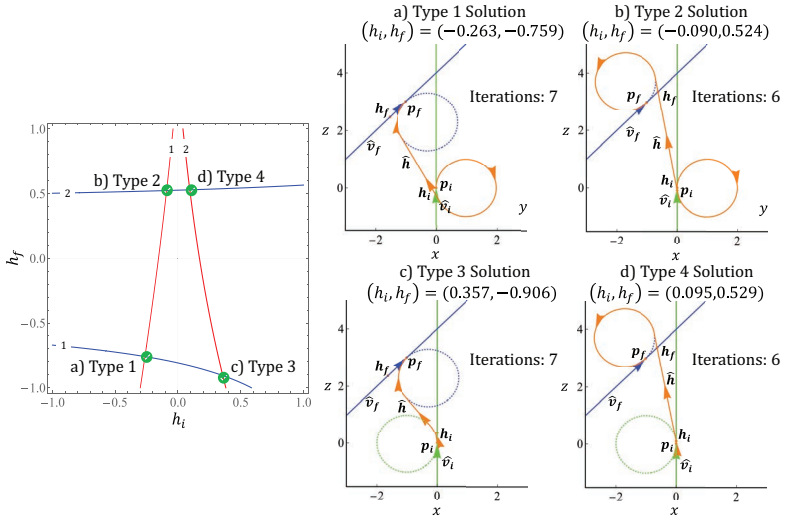

Fig. 4: **2D Problem Case 1 (Far)**: $\mathbf{x}_i = [0,0,0], \mathbf{x}_f = [-1,0,3], \hat{\mathbf{v}}_i = [0,0,1], \hat{\mathbf{v}}_f = \frac{1}{\sqrt{2}}[1,0,1]$. (Left) Solution plot for regular solutions of p_i (red) and p_f (blue). Intersections marked by green checks show the (h_i, h_f) solution pairs. (Right) CSC paths for each solution type 1-4 marked on the left.

ducing all four solutions (one of each regular type) in 0.031 s and requiring fewer than 10 solver iterations for each solution pair.

When starting and ending positions are close together in 3D, similarly to the planar case, there are not only solutions of the four regular types. Figure 7 shows the results for one such example: $\mathbf{x}_i = [0,0,0], \mathbf{x}_f = [-1,0,3], \hat{\mathbf{v}}_i = \frac{1}{\sqrt{3}}[1,1,1], \hat{\mathbf{v}}_f = [0,0,1]$ The solver found all intersection points in 0.061 s. The number of iterations required for the numerical solver to find solutions increases when the starting and ending positions are close together, and some equation pairs produce invalid $(\mathbf{h}_i, \mathbf{h}_f)$ solutions that need to be filtered out.

3.2 Slices of Planar and Non-Planar Solution Spaces

To explore the solution space and the difficulty of computing solution paths, we sweep a range of starting and ending configurations for both 2D and 3D cases.

Planar Solution Spaces Fixing the starting point at $\mathbf{x}_i = [0,0,0]$ and the starting direction at $\hat{\mathbf{v}}_i = [1,0,0]$, we investigate the solution space for planar cases by varying the end position $\mathbf{x}_f = [x,0,z]$ and direction $\hat{\mathbf{v}}_f = [-\sin(\theta), 0, \cos(\theta)]$ for values of $x, z \in [-6,6]$ and $\theta \in [0, 2\pi)$. The initial guess for the solver is set at $h_i = h_f = 0$. Figure 8 shows the results. Slices of the problem space are colored according to the number of solutions (regular and switched) found. The solver is able to find at least one solution for most inputs. Further, the solver produces

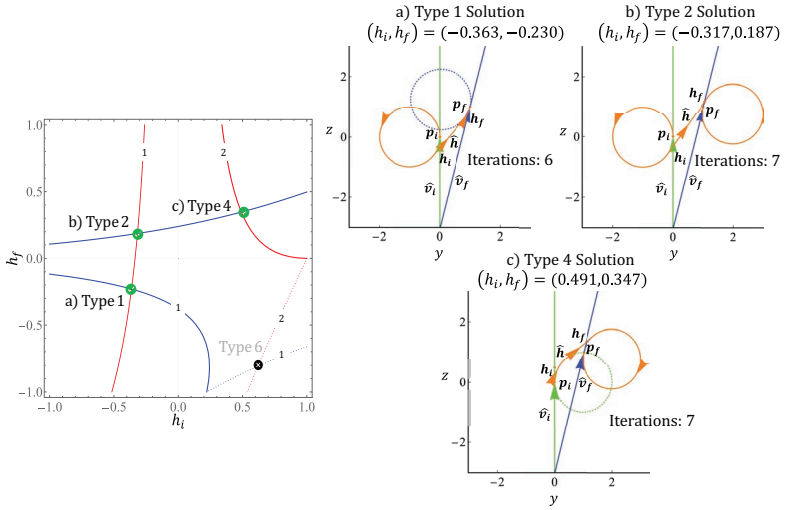

Fig. 5: **2D Problem Case 1 (Close)**: $\mathbf{x}_i = [0, 0, 0], \mathbf{x}_f = [0, 1.01, 1], \hat{\mathbf{v}}_i = [0, 0, 1], \hat{\mathbf{v}}_f = \frac{1}{\sqrt{17}}[0, 1, 4]$. (Left) Solution plot for regular solutions of p_i (solid red), regular solutions of p_f (solid blue), switched solutions of p_i^* (dotted red), and switched solutions of p_f^* (dotted blue). Intersections marked by green checks show the (h_i, h_f) valid solution pairs, and black xs show invalid solutions. (Right) CSC paths for each solution type. A Type 3 solution does not exist in this case.

expected symmetries. For example, as shown 8(a), when $\theta = 0$, the starting and ending directions are aligned, so we expect that the number of solutions should be symmetric around $x = 0$. If $x = 0$, the number of solutions should be symmetric about $\theta = \pi$, shown in Figure 8(b). Finally, when fixing the z coordinate in $\mathbf{x}_f = [x, 0, z]$, the number of solutions should be symmetric about $x = 0$ and $\theta = \pi$, as shown in the Figure 8(c).

Similarly to the qualitative observations on our individual test cases, the plots reveal that when the starting and ending positions are far apart, the solver is able to find 4 regular solutions much of the time. However, as the distance between starting and ending position decreases, the number of regular solutions also decreases, and switched solutions must be found. We did not compute the true number of solutions for each of the tested scenarios. It remains to be determined for some of the cases where fewer solutions were found whether it is because there are fewer solutions or because the solver did not find them all.

Non-Planar Solution Spaces Extending to non-planar problems, we consider the following situation. Fixing the starting point at $\mathbf{x}_i = [0, 0, 0]$ and the starting direction at $\hat{\mathbf{v}}_i = [0, 0, 1]$, we analyze the solution space of a simplified non-planar problem where we vary the ending point $\mathbf{x}_f = [x, 0, z]$ and the ending direction $\hat{\mathbf{v}}_f = [\sin(\phi), \cos(\phi), 0]$ for values of $x, z \in [-6, 6]$ and $\phi \in [0, 2\pi)$. The initial guess for the solver is set at $h_i = h_f = 0$. Figure 9 shows the results.

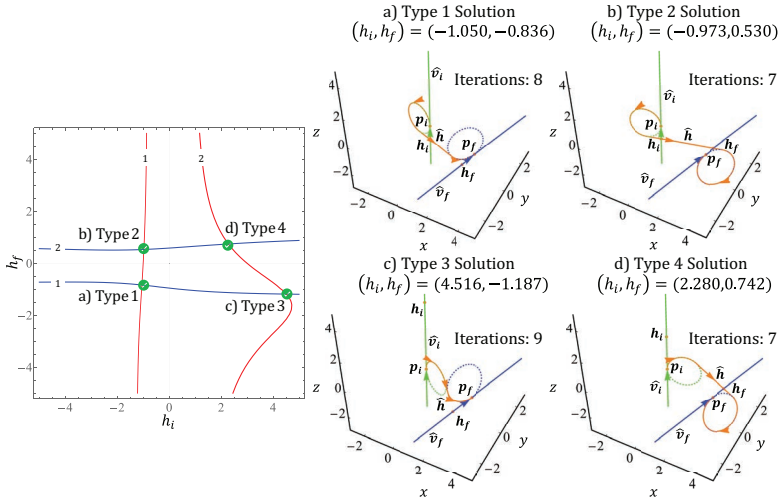

Fig. 6: **3D Problem Case 1 (Far)**: $\mathbf{x}_i = [0,0,0], \mathbf{x}_f = [3,0,-1], \hat{\mathbf{v}}_i = [0,0,1], \hat{\mathbf{v}}_2 = \frac{1}{\sqrt{21}}[2,4,1]$. (Left) Solution plot for regular solutions of p_i (red) and p_f (blue). Intersections marked by green checks show the (h_i, h_f) solution pairs. (Right) CSC paths for each solution type 1-4 marked on the left.

Again, the produced solution space shows expected symmetries. When fixing $x = 0$, the solution space has vertical "stripes" to indicate that changing the angle ϕ does not change the number of solutions (Figure 9(a)). Additionally, fixing some value of z, (in this case $z = -1$) makes the number of solutions symmetric about $\phi = \pi$ since the relative alignment between the starting and ending directions would be equivalent (Figure 9(b)). Finally, the number of solutions found is rotationally symmetric about the starting position. That is, if we fix $\phi = \phi_1$ for $\hat{\mathbf{v}}_i = [\sin(\phi_1), \cos(\phi_1), 0]$ and vary $\mathbf{x}_f = [x, 0, z]$, the landscape of the number of solutions for the slice where $\phi = \pi - \phi_1$ is a mirror image (Figure 9(c-d)).

Similar to the planar case, the outer edges of the plots, corresponding to the final configuration being farther away, produce in many cases the four regular solutions for h_i and h_f. We also observe that non-planar cases with a sufficiently small distance between initial and final configurations result in 5 solutions, which must contain a switched solution. This behavior cannot be achieved by planar cases, since it is limited to four possible combinations between the circles.

3.3 Varying Starting Seeds

In some cases, there are multiple solutions for a single solution type. Using a numerical solver, at most one of these solutions will be found, and the performance of the solver depends on the starting seed. We investigate this effect on a case where the starting and ending points are close enough that the valid h_i, h_f values are not the four regular solutions. When $\mathbf{x}_i = [0,0,0], \mathbf{x}_f =$

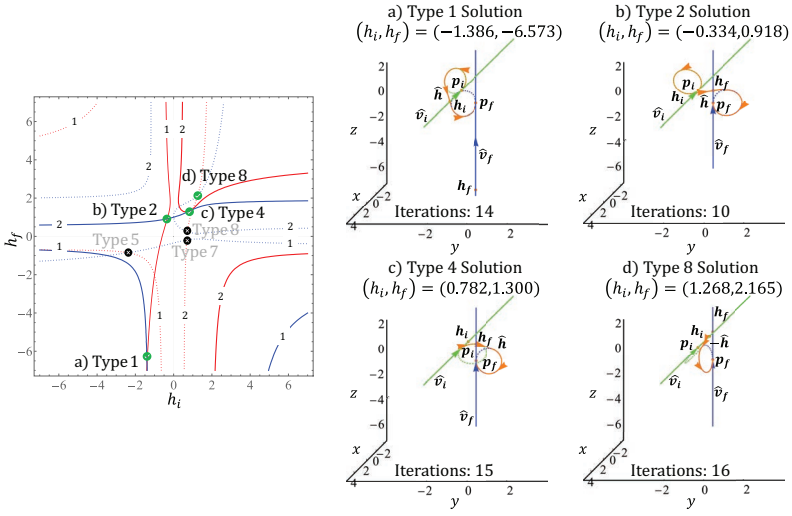

Fig. 7: **3D Problem Case 1 (Close)**: $\mathbf{x}_i = [0, 0, 0], \mathbf{x}_f = [-1, 0, 3], \hat{\mathbf{v}}_i = \frac{1}{3}[1, 1, 1], \hat{\mathbf{v}}_f = [0, 0, 1]$. (Left) Solution plot for regular solutions of p_i (solid red), regular solutions of p_f (solid blue), switched solutions of p_i^* (dotted red), and switched solutions of p_f^* (dotted blue). Intersections marked by green checks show the (h_i, h_f) valid solution pairs, and black xs show invalid solutions. There are 3 regular, 1 switched, and 3 invalid (Type 5, 7, 8) solutions. (Right) CSC paths for each of the solution types marked on the left.

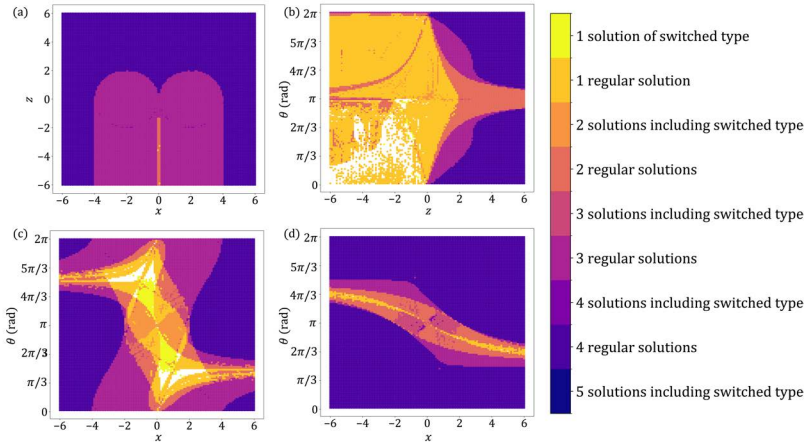

Fig. 8: **Planar Solution Spaces (a)** $x - z$ plane when $\theta = 0$. **(b)** $z - \theta$ plane when $x = 0$. **(c-d)** $x - \theta$ plane when $z = -1$ (left) and $z = 3$ (right).

$[1.059, 0, -4.588], \hat{\mathbf{v}}_i = [0, 0, 1], \hat{\mathbf{v}}_f = [-0.361, 0, 0.932]$, there are two switched type 6 solutions ($p_i^* > h_i, p_f^* < h_f$). Figure 10 shows which solutions are found

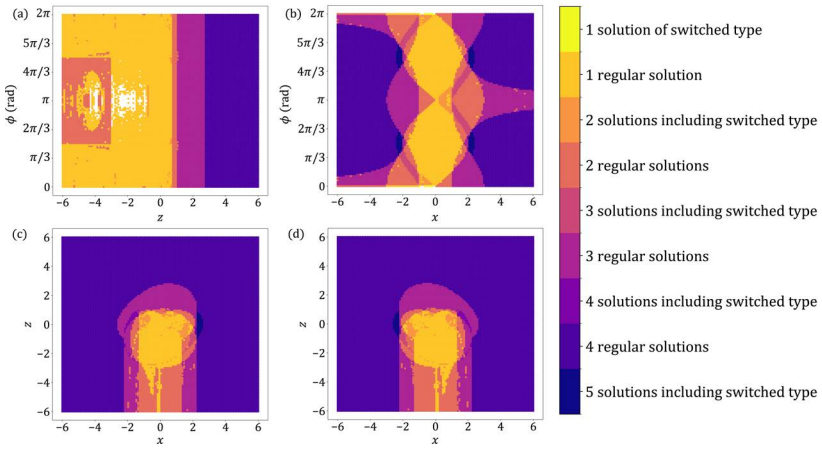

Fig. 9: **Non-planar Solution Spaces (a)** $z - \phi$ plane when $x = 0$. **(b)** $x - \phi$ plane when $z = -1$. **(c-d)** $x - z$ plane slice when $\phi = \pi - 2$ rad (left) and and $x - z$ plane slice when $\phi = 2$ rad (right).

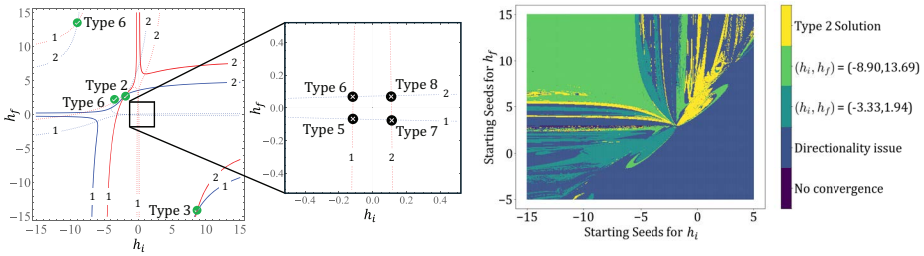

Fig. 10: **Effect of Initial Starting Seeds** (Left) Solution plot for $\mathbf{x}_i = [0, 0, 0], \mathbf{x}_f = [1.059, 0, -4.588], \hat{\mathbf{v}}_i = [0, 0, 1], \hat{\mathbf{v}}_f = [-0.361, 0, 0.932]$ with multiple type 6 solutions. (Right) Solver convergence behavior for the Type 6 solution, depending on initial h_i, h_f seeds. Yellow areas converge to the Type 2 solution $h_i = -1.804, h_f = 3.004$ that geometrically looks very close to a Type 6.

for different starting seed values. We find that when varying the initial guesses for h_i and h_f, the solver is able to converge to one of the intersection points in almost all cases. However, the solution that the solver converges to is incredibly sensitive to the starting seed, indicating that it may be difficult to force the solver to find a different solution, even if it is known that others exist.

3.4 Performance Comparison

Efficacy of Gradient Inclusion in Solving for Solutions Our equations admit gradients, which may provide the solver with some assistance in converging to a valid solution.

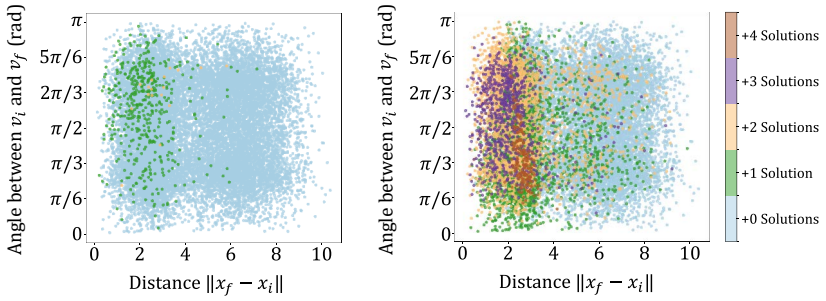

Fig. 11: **Effect of Gradients on Numerical Solver and Comparison to [13]** (Left) Number of additional solutions found when using `fsolve` with minus without the gradient for 18,000 random cases. (Right) Number of solutions found when using `fsolve` with our formulation minus [13] for 18,000 random cases.

We run a test with $\mathbf{x}_i = [0, 0, 0], \hat{\mathbf{v}}_i = [0, 0, 1]$ and 18,000 random final configurations and initial guesses for h_i and h_f. For each final configuration and initial guess combination, we compute the CSC path solutions with and without the gradients and compare the number of solutions found (Figure 11). In the majority (95%) of cases, the solver finds the same solutions with and without the gradient. However, when the initial and final configurations are close together with large orientation differences, the gradient makes the calculation easier, helping the solver to find up to 2 more solutions. This makes sense since the closer the initial and final positions are, the more influence the alignment of the directions will have over orientations of the arc and straight line components.

Comparison with Existing Method [13] Finally, we analyze how solving these equations compares to existing solution methods. We refer to [13], which is a different geometrical derivation for 3D CSC Dubins paths resulting in a nonlinear system of equations over five variables. We input the associated equations into `fsolve` and run the solver on the same 18,000 test cases as above and compare the number of solutions found (Figure 11).

In all cases, the number of solutions found using our simplified formulation is at least as many as that found using [13], and in the large majority of cases (95.7%), our formulation is able to produce more solution paths: 4.39% of the time we find the same number of solutions, 13.63% of the time we find one more solution, 21.31% of the time we find two more solutions, 59.65% of the time we find three more solutions, and 1.07% of the time we find four more solutions. We observe that our formulation produces a larger number of valid solutions when the distance between the initial and final configuration is small and, in some cases, the formulation in [13] is unable to produce any solutions at all.

We therefore conclude that our simplified formulation is able to streamline and robustify the computation of CSC paths.

Note that for this test, we did not vary the initial seeds, and the quality of the output may change if multiple solution attempts are made. The time taken to solve each system of equations, for our formulation (with and without gradients) or for [13], was approximately 0.01 s per test case.

4 Conclusion

We investigate the path planning problem for 3D Dubins vehicles, focusing particularly on CSC paths, and we present a parameterization of the problem for computing solution paths by solving a system of 2 nonlinear equations. We have shown that these equations can be written explicitly from the input positions and orientations, and that we can write analytical derivatives for these equations. We have used these equations together with the off-the-shelf Python `fsolve` solver and present results for multiple valid paths for a number of inputs. Further, we have demonstrated the ability to find more valid solutions than an existing geometric method [13], even at small distances between initial and final configurations, a limitation that other approaches have encountered. When the starting and ending positions are far apart, almost all solution types produce valid paths. However, when the starting and ending positions are close, some solution types do not produce valid paths, and others produce multiple valid paths. We have started to characterize empirically when these different situations occur. Future work includes a more rigorous study of the solution space for this problem.

We observe that our empirical results highly depend on the behavior of the numerical solver being used. In particular, for solution types that have multiple valid solutions, the generated output depends on the initial seed for the solver. In this paper, we used the `fsolve` solver, which may produce outputs that are not the closest solution (in terms of h_i, h_f distance) to the initial seed. However, at the same time, we were able to compute analytical derivatives for the equations that we used. We therefore expect that these gradients can be used in gradient-descent based algorithms to search for solutions that are close to an initial guess and reduce variability in the output. These approaches may be useful for future work in real-time planning for 3D vehicles, where small adjustments to the current path may be preferred over large changes. Future work includes further investigation into improving consistency of output solutions for these scenarios.

Acknowledgments. Support for this project has been provided in part by NSF Grant No. 2322898, by the Army Research Office under the SLICE Multidisciplinary University Research Initiatives Program award under Grant #W911NF1810327, and by the AFOSR Multidisciplinary University Research Initiatives Program HyDDRA. We also thank Wei-Hsi Chen and Daniel Feshbach for helpful discussions.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Ambrosino, G., Ariola, M., Ciniglio, U., Corraro, F., De Lellis, E., Pironti, A.: Path generation and tracking in 3-D for UAVs. IEEE Transactions on Control Systems Technology **17**(4), 980–988 (2009). https://doi.org/10.1109/TCST.2009.2014359
2. Baez, V.M., Navkar, N., Becker, A.T.: An analytic solution to the 3D CSC Dubins path problem. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 7157–7163 (2024). https://doi.org/10.1109/ICRA57147.2024.10611360
3. Blevins, A.T.: Real-time path optimization for 3D UAS line survey operations. In: AIAA SCITECH Forum. p. 0395 (2023). https://doi.org/10.2514/6.2023-0395
4. Boissonnat, J.D., Cerezo, A., Leblond, J.: Shortest paths of bounded curvature in the plane. In: IEEE International Conference on Robotics and Automation. pp. 2315–2320 (1992). https://doi.org/10.1109/ROBOT.1992.220117
5. Chen, W.H., Yang, W., Peach, L., Koditschek, D.E., Sung, C.R.: Kinegami: Algorithmic design of compliant kinematic chains from tubular origami. IEEE Transactions on Robotics **39**(2), 1260–1280 (2023). https://doi.org/10.1109/TRO.2022.3206711
6. Chitsaz, H., LaValle, S.M.: Time-optimal paths for a Dubins airplane. In: IEEE Conference on Decision and Control. pp. 2379–2384 (2007). https://doi.org/10.1109/CDC.2007.4434966
7. Cui, P., Yan, W., Wang, Y., et al.: Reactive path planning approach for docking robots in unknown environment. Journal of Advanced Transportation (2017). https://doi.org/10.1155/2017/6716820
8. Dubins, L.E.: On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. American Journal of Mathematics **79**(3), 497–516 (1957). https://doi.org/10.2307/2372560
9. Faigl, J., Vána, P.: Unsupervised learning for surveillance planning with team of aerial vehicles. In: IEEE International Joint Conference on Neural Networks (IJCNN). pp. 4340–4347 (2017). https://doi.org/10.1109/IJCNN.2017.7966405
10. Feshbach, D., Chen, W.H., Xu, L., Schaumburg, E., Huang, I., Sung, C.: Algorithmic design of kinematic trees based on CSC Dubins planning for link shapes. In: International Workshop on the Algorithmic Foundations of Robotics (WAFR) (2024)
11. Herynek, J., Váña, P., Faigl, J.: Finding 3D dubins paths with pitch angle constraint using non-linear optimization. In: European Conference on Mobile Robots (ECMR). pp. 1–6 (2021). https://doi.org/10.1109/ECMR50962.2021.9568787
12. Hota, S., Ghose, D.: Optimal geometrical path in 3D with curvature constraint. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 113–118 (2010). https://doi.org/10.1109/IROS.2010.5653663
13. Hota, S., Ghose, D.: Optimal path planning for an aerial vehicle in 3D space. In: IEEE Conference on Decision and Control (CDC). pp. 4902–4907 (2010). https://doi.org/10.1109/CDC.2010.5717246
14. Huang, X., Yan, C.B.: Dubins curve based continuous-curvature trajectory planning for autonomous mobile robots. arXiv preprint arXiv:2309.07565 (2023)
15. Indig, N., Ben-Asher, J.Z., Sigal, E.: Near-optimal minimum-time guidance under spatial angular constraint in atmospheric flight. Journal of Guidance, Control, and Dynamics **39**(7), 1563–1577 (2016). https://doi.org/10.2514/1.G006218

16. Ismail, A., Tuyishimire, E., Bagula, A.: Generating Dubins path for fixed wing uavs in search missions. In: International Symposium on Ubiquitous Networking. pp. 347–358 (2018). https://doi.org/10.1007/978-3-030-02849-7_31
17. Lin, Y., Saripalli, S.: Path planning using 3D dubins curve for unmanned aerial vehicles. In: International Conference on Unmanned Aircraft Systems (ICUAS). pp. 296–304 (2014). https://doi.org/10.1109/ICUAS.2014.6842268
18. Marino, H., Bonizzato, M., Bartalucci, R., Salaris, P., Pallottino, L.: Motion planning for two 3D-Dubins vehicles with distance constraint. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4702–4707 (2012). https://doi.org/10.1109/IROS.2012.6385866
19. Patsko, V., Fedotov, A.: Three-dimensional reachability set for a Dubins car: Reduction of the general case of rotation constraints to the canonical case. Journal of Computer and Systems Sciences International **62**, 445–468 (2023). https://doi.org/10.1134/S1064230723030115
20. Sachkov, Y.L.: Left-invariant optimal control problems on Lie groups: classification and problems integrable by elementary functions. Russian Mathematical Surveys **77**(1), 99 (2022). https://doi.org/10.1070/RM10019
21. Shkel, A.M., Lumelsky, V.: Classification of the Dubins set. Robotics and Autonomous Systems **34**(4), 179–202 (2001). https://doi.org/10.1016/S0921-8890(00)00127-5
22. Sussmann, H.J.: Shortest 3-dimensional paths with a prescribed curvature bound. In: IEEE Conference on Decision and Control. vol. 4, pp. 3306–3312 (1995). https://doi.org/10.1109/CDC.1995.478997
23. Váña, P., Alves Neto, A., Faigl, J., Macharet, D.G.: Minimal 3D dubins path with bounded curvature and pitch angle. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 8497–8503 (2020). https://doi.org/10.1109/ICRA40945.2020.9197084
24. Wang, W., Li, P.: Towards finding the shortest-paths for 3D rigid bodies. In: Robotics: Science and Systems (RSS) (2021)
25. Wang, Y., Wang, S., Tan, M., Zhou, C., Wei, Q.: Real-time dynamic Dubins-helix method for 3-D trajectory smoothing. IEEE Transactions on Control Systems Technology **23**(2), 730–736 (2015). https://doi.org/10.1109/TCST.2014.2325904
26. Wu, W., Xu, J., Gong, C., Cui, N.: Adaptive path following control for miniature unmanned aerial vehicle confined to three-dimensional Dubins path: From take-off to landing. ISA Transactions **143**, 156–167 (2023). https://doi.org/10.1016/j.isatra.2023.09.021
27. Yang, X., Zhu, X., Liu, W., Ye, H., Xue, W., Yan, C., Xu, W.: A hybrid autonomous recovery scheme for AUV based Dubins path and non-singular terminal sliding mode control method. IEEE Access **10**, 61265–61276 (2022). https://doi.org/10.1109/ACCESS.2022.3180836