To appear in the Proceedings of the 2024 IEEE/MTS Oceans Conference, Halifax, NS, Canada

3D Dubins Paths for Underwater Vehicles

Mark Moll
Metron, Inc.
Reston, VA, USA
mollm@metsci.com

***Abstract*—The dynamics of unmanned underwater vehicles (UUVs) impose significant constraints on the types of paths that can be followed. A number of motion models have been proposed that are inspired by the Dubins vehicle model. We have investigated two state-of-the-art models that ensure a bounded turning rate, bounded pitch angle, and (in one case) a bounded pitch change rate. We will show that these existing models can produce undesirable solutions (or even no solutions) between an initial and final pose of a UUV. Here, we present a new model that combines these two models, preserves the best features of both, and is empirically shown to connect any arbitrary start and goal pose. We also present results that show how the new model can be used as a local planner in sampling-based algorithms.**

***Index Terms*—Path planning, 3D Dubins, UUVs.**

I. Introduction

When planning paths for mobile robots, aerial vehicles, or underwater vehicles it is important to consider the motion constraints of the vehicle under consideration. The true system dynamics are often too complex to allow for planning over long time horizons, so simpler analytic models have been proposed that can be used to produce reference paths which can then be used to synthesize a trajectory and a controller. For navigating at constant depth, the Dubins model is often used [1]. The Dubins model describes motions of car-like vehicles that can only drive forward and have a bounded turning radius. The time-optimal trajectories between any two vehicle poses can be solved for analytically [1, 2]. Such trajectories consist of at most three segments corresponding to arcs of a circle and straight line segments.

The extension of the Dubins model to 3D is non-trivial. Several variants have been studied and different numerical and analytical methods have been proposed to compute shortest paths between given endpoints with specified tangents and curvature constraints (see, e.g., [3, 4, 5]). Many of these methods do not explicitly account for pitch angle constraints. To use such methods for underwater or aerial vehicles, one has to assume that endpoints are sufficiently far apart to ensure pitch is bounded. Chitsaz and LaValle [6] provide a characterization of a 3D Dubins model that *does* account for pitch angle constraints, but it does not provide a method to construct 3D Dubins curves that connect a start pose to a desired goal pose.

A practical approach to constructing Dubins curves in 3D with pitch constraints was presented by Owen et al. [7]. They classify the transitions between two vehicle poses into three classes with well-defined boundary conditions: low-, medium-, and high-depth changes. In the low-depth case, the path can be computed analytically. In the high-depth change case, a path is composed of a spiral depth change followed by a Dubins curve, all while changing the depth at the maximum pitch. For the medium depth change an extra turn is added to a Dubins path, while, again, changing depth at the maximum pitch. Both the medium and high depth paths require a one-dimensional numerical search. Within the Owen et al. model it is assumed that the pitch can be changed instantly (or that the path length needed to change pitch is negligibly small). This is not always the case. The more recent model proposed by Váňa et al. [8] addresses this limitation with an entirely different approach. The path between two poses is assumed to be curvature-constrained, which affects both turning rate and pitch rate equally. A path between two poses is computed in two stages. First, a Dubins path in the XY plane is computed. Next, a Dubins a path is computed in the SZ plane, where S is the arc length of the Dubins curve in the XY plane. The two 2D Dubins paths can be turned into one 3D Dubins path by taking the X and Y coordinates of the first curve and the Z coordinates along the second curve. The turning radii for the two 2D Dubins paths are chosen so that the curvature of the resulting 3D Dubins curve does not exceed the curvature limit. While this method ensures smooth pitch changes, it can be shown that for sufficiently large depth changes, this method produces undesirably long solutions or no solutions, as described below.

The rest of the paper is organized as follows. Section II gives a formal definition of the problem addressed by this paper. Section III presents our proposed algorithm for solving this problem. Section IV includes results from comparing the new algorithm against previous approaches. Section V describes how the new 3D Dubins model as well previous can be used as local planners within sampling-based planners, a class of motion planning algorithms. Section VI concludes the paper with discussion of the results and future work.

II. Problem Definition

The problem this paper seeks to address can be formally defined as follows (and follows the definition of [8]). Let $Q = \mathbb{R}^3 \times \mathrm{SO}(2) \times \mathrm{SO}(2)$ be the state space of a UUV. An element $\mathbf{q} = \langle x, y, z, \theta, \gamma \rangle \in Q$ describes the 3D position $\langle x, y, z \rangle$, the heading θ and pitch angle γ of the vehicle. Given a start configuration $\mathbf{q}_s$ and a goal configuration $\mathbf{q}_g$, we wish to find the shortest C^1 continuous path connecting $\mathbf{q}_s$ and $\mathbf{q}_g$ subject to the constraints described below. Let the path be described by $c(t) : [0, 1] \to \mathbb{R}^3$. The heading θ is defined in the XY projection of this path, while pitch is defined as the angle

between the curve tangent and the projection of that tangent onto the XY plane (diving corresponds to a negative pitch and surfacing to a positive pitch). The path is subject to the following constraints:

- The path starts at $\mathbf{q}_s$ and ends at $\mathbf{q}_g$: $c(0) = \mathbf{q}_s$ and $c(1) = \mathbf{q}_f$.
- The curvature $\kappa(t) = \|c'(t) \times c''(t)\|/\|c'(t)\|^3$ has to be less than or equal to $\rho_{\min}^{-1}$, where $\rho_{\min}$ is a minimum turning radius constraint. Note that this constraint affects both turns in the XY plane and the pitch rate.
- The pitch angle γ is bound to lie in the interval $\Gamma = [\gamma_{\min}, \gamma_{\max}]$.

III. Method

We will first present an informal high-level description of our approach. This is followed by a detailed description of the algorithm. In the Váňa et al. model significant depth changes are dealt with by increasing the turning radius, which results in longer paths. This, in turn, enables larger depth changes. If, however, the path in the XY plane is equivalent to a straight line, a solution will not be found. Furthermore, if the path in the XY plane is *close* to a straight line, the turning radius has to be made extremely large to make any necessary depth changes, resulting in very long paths. We observe that the model by Váňa et al. [8] essentially corresponds to low-depth transitions, in the terminology of Owen et al. The feasibility check in [8] defines the boundary between low-depth and medium-depth changes. We can define the boundary between medium-depth and high-depth changes as follows. For a given depth change, an initial pitch angle, and a final pitch angle, we can analytically compute the maximum depth change along a path in the XY plane (see Figure 1). This can be used to define a decision boundary between medium- and high-depth changes as described below. If the required depth change exceeds this boundary, we insert a spiral depth change. If the required depth change does not exceed this boundary, but also does not satisfy the feasibility check in [8], we append an extra turn in the XY plane to construct a medium-depth change path. In all cases the pitch and pitch rate satisfy their respective limits by construction. This approach results computes a locally optimal solution to the problem defined above by combining the two algorithms of Váňa et al. [8] and Owen et al. [7]. We will call this hybrid the Váňa–Owen algorithm.

At a high level, the Váňa–Owen algorithm retains the same structure as the Váňa algorithm. The main algorithm repeatedly calls a subroutine that for a given horizontal turning radius ρ_h ($\rho_h \geq \rho_{\min}$) computes the shortest 3D Dubins path. It searches over ρ_h to find the shortest 3D Dubins path over all valid values of ρ_h. This part is almost identical to the one in [8] and is shown for completeness in Algorithm 1. The main difference is that a 3D Dubins path is no longer completely defined by a pair $\langle D_h, D_v \rangle$ corresponding to 2D Dubins curves in the XY plane and SZ plane, respectively. Instead, a 3D Dubins curve is now defined by a 4-tuple $D = \langle D_h, D_v, k, \phi \rangle$, where k ($k \geq 0$) is a number of helical turns and ϕ is the arc length of an optional additional turn. Similar to [7], the additional

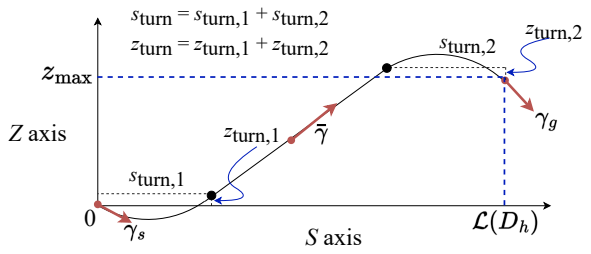


Fig. 1. **Illustration of maximum depth change.** Given an initial pitch γ_s, a final pitch γ_g, a maximum pitch of $\bar{\gamma}$, and a length of a XY path $\mathcal{L}(D_h)$, we can determine s_{turn}, the path length needed to achieve $\bar{\gamma}$ with initial and final turns, and z_{turn} the net change in depth due to these turns. From this figure, we see that the maximum change in depth is therefore $z_{\max} = (\mathcal{L}(D_h) - s_{\text{turn}})/\tan\bar{\gamma} + z_{\text{turn}}$.

Algorithm 1: VáňaOwenDubins3D($\mathbf{q}_s, \mathbf{q}_g$)

Input: $\mathbf{q}_s, \mathbf{q}_g$: start and goal configurations
Output: D: a 4-tuple that completely characterizes a 3D Dubins path

1 $\rho_h \leftarrow \rho_{min}$
2 **repeat** // find initial solution
3 $\quad \rho_h \leftarrow 2\rho_h$
4 **until** VáňaOwenDecoupled($\mathbf{q}_s, \mathbf{q}_g, \rho_h, D$)
5 $\Delta \leftarrow 0.1\rho_{\min}$
6 **repeat** // minimize length
7 $\quad \rho_h' \leftarrow \max(\rho_{\min}, \rho_h + \Delta)$
8 $\quad$ **if** VáňaOwenDecoupled($\mathbf{q}_s, \mathbf{q}_g, \rho_h, D'$) and $\mathcal{L}(D') < \mathcal{L}(D)$ **then**
9 $\quad\quad \rho_h \leftarrow \rho_h'$
10 $\quad\quad D \leftarrow D'$
11 $\quad\quad \Delta \leftarrow 2\Delta$
12 $\quad$ **else**
13 $\quad\quad \Delta \leftarrow -0.1\Delta$
14 **until** $|\Delta| \leq \Delta_{min}$
15 **return** D

two parameters are used to encode medium- and high-depth transitions as described below.

The length of a 3D Dubins curve D is denoted with $\mathcal{L}(D)$. The length of the corresponding 2D Dubins curves is denoted with $\mathcal{L}(D_h)$ and $\mathcal{L}(D_v)$. It can be shown that $\mathcal{L}(D) = \mathcal{L}(D_v)$. The minimum turning radius $\rho_{\min}$ and pitch range Γ are assumed to be global constants.

The main computation occurs in the VáňaOwenDecoupled subroutine shown in Algorithm 2. It uses a function called Dubins2D to analytically compute the shortest 2D Dubins path between two poses $\langle x_0, y_0, \theta_0 \rangle$ and $\langle x_1, y_1, \theta_1 \rangle$. Algorithm 2 first computes a path D_h in the XY plane. Given the horizontal turning ρ_h, it can be shown that setting the turning radius in the SZ plane equal to $(\rho_{\min}^{-2} - \rho_h^{-2})^{-\frac{1}{2}}$ ensures the curvature constraint is satisfied [8]. There are then four cases to consider:

a) Edge case (lines 5–10): If the specified horizontal turning radius ρ_h is equal to the minimum turning radius $\rho_{\min}$, then any turn in the SZ plane would cause the curvature to

Algorithm 2: VáňaOwenDecoupled($\mathbf{q}_s, \mathbf{q}_g, \rho_h, \langle D_h, D_v, k, \phi\rangle$)

Input: $\mathbf{q}_s, \mathbf{q}_g$: start and goal configurations
Input: ρ_h: horizontal turning radius
Output: D_h, D_v: hor. and vert. Dubins maneuver
Output: k: num. helical turns (high-depth change)
Output: ϕ: additional turn angle (med.-depth change)
Output: returns **true** iff a valid solution was found

1 $D_h \leftarrow \text{Dubins2D}(\langle x_s, y_s, \theta_s\rangle, \langle x_g, y_g, \theta_g\rangle, \rho_h)$
2 $k \leftarrow 0$
3 $\phi \leftarrow 0.0$
4 $\Delta_z \leftarrow z_g - z_s$
5 **if** $\rho = \rho_{min}$ **then**
  // a) edge case: path in XY-plane
6 **if** $\arctan\left(\frac{\Delta_z}{\mathcal{L}(D_h)}\right) = \gamma_s = \gamma_g$ **then**
7 $D_v \leftarrow$ "a straight line"
8 **return true**
9 **else**
10 **return false**
11 $\rho_v \leftarrow (\rho_{\text{min}}^{-2} - \rho_h^{-2})^{-\frac{1}{2}}$
12 $D_v \leftarrow \text{Dubins2D}(\langle 0, z_s, \gamma_s\rangle, \langle \mathcal{L}(D_h), z_g, \gamma_g\rangle, \rho_v)$
13 **if** IsValid(D_v) **then**
  // b) low-depth transition
14 **return true**
15 **if** $|\Delta_z| >$ depthThreshold($\mathbf{q}_s, \mathbf{q}_g, \mathcal{L}(D_h)$) **then**
  // c) high-depth transition
16 $k \leftarrow \lfloor(|\Delta_z - z_{\text{turn}}|/\tan\bar{\gamma} - \mathcal{L}(D_h) + s_{\text{turn}})/(2\pi\rho_h)\rfloor$
17 $\rho_h \leftarrow$ $\text{argmin}_\rho |(\mathcal{L}(D_h) - s_{\text{turn}} + 2\pi\rho)\tan\bar{\gamma} - (\Delta_z - z_{\text{turn}})|$
18 $D_h \leftarrow \text{Dubins2D}(\langle x_s, y_s, \theta_s\rangle, \langle x_g, y_g, \theta_g\rangle, \rho_h)$
19 $\rho_v \leftarrow (\rho_{\text{min}}^{-2} - \rho_h^{-2})^{-\frac{1}{2}}$
20 $D_v \leftarrow \text{Dubins2D}(\langle 0, z_s, \gamma_s\rangle, \langle \mathcal{L}(D_h), z_g, \gamma_g\rangle, \rho_v)$
21 **else** // d) medium-depth transition
22 $\phi \leftarrow \text{argmin}_\phi \left|\left(\phi\rho_h + \mathcal{L}\big(\text{Dubins2D}(\mathbf{q}_i(\phi), \mathbf{q}_g, \rho_h)\big)\right)\tan\bar{\gamma} - \Delta_z\right|$
23 $D_h \leftarrow \text{Dubins2D}(\mathbf{q}_i(\phi), \langle x_g, y_g, \theta_g\rangle, \rho_h)$
24 $D_v \leftarrow$ $\text{Dubins2D}(\langle 0, z_s, \gamma_s\rangle, \langle \mathcal{L}(D_h) + \phi\rho_h, z_g, \gamma_g\rangle, \rho_v)$
25 **return true**

be greater than ρ_{min}^{-1}. This means that if $\rho_h = \rho_{\text{min}}$, a solution exists if and only if the start and goal can be connected with a straight line. This is true if the start and goal pitch match the slope of the straight line connecting start and goal in the SZ plane.

b) Low-depth transition (lines 11–14): We can compute a Dubins path in the SZ plan with a turning radius ρ_v set to the lower bound. This solution is valid if and only if the pitch angles along this path are within the range Γ and the middle segment of D_v is a straight-line segment [8].

c) High-depth transition (lines 15–20): Let $\bar{\gamma}$ be γ_{min} if $\Delta_z < 0$ and γ_{max} otherwise. For a given start state $\mathbf{q}_s$ and goal state $\mathbf{q}_g$, we can compute the absolute maximum possible height change. This height change is achieved by turning from γ_s achieve pitch $\bar{\gamma}$, go straight at a constant pitch $\bar{\gamma}$, and, finally, turn again from $\bar{\gamma}$ to γ_g. The turns correspond to arcs of a circle with radius ρ_v. Figure 1 shows an illustration of such a path. The displacement along the S axis due the turns is denoted with s_{turn}, while the displacement along the Z axis is denoted with z_{turn}. From these displacements we can derive the maximum possible depth change.

Following [7], if $|z_{\text{max}}| + 2\pi\rho_h \leq |\Delta_z|$ (line 15 in Algorithm 2), then a helical turn is required. The number of turns of this helix is equal to

$$k = \left\lfloor \frac{|(\Delta_z - z_{\text{turn}})/\tan\bar{\gamma}| - \mathcal{L}(D_h) + s_{\text{turn}}}{2\pi\rho_h} \right\rfloor.$$

Using a numerical search, we then find ρ_h^* such that

$$(\mathcal{L}(D_h) + 2\pi k\rho_h^* - s_{\text{turn}})\tan\bar{\gamma} + z_{\text{turn}} = \Delta_z.$$

Note that almost all the turns on the left-hand side of this equation implicitly depend on ρ_h^*.

d) Medium-depth transition (lines 21–24): If less than one turn is needed to achieve the desired depth and a low-depth transition is not feasible, then we simply lengthen the XY path by prepending an extra turn before a 2D Dubins path. A numerical search is used to add just enough length to achieve the desired depth change. Let the initial turn be an arc of a circle with radius ρ_h parametrized by ϕ and let the intermediate pose achieved after this turn be denoted with $\mathbf{q}_i = \langle x_i, y_i, \theta_s + \phi\rangle$. The length of the path in the XY plane as a function of ϕ is thus

$$\mathcal{L}(\phi) = \phi\rho_{\text{h}} + \mathcal{L}(\text{Dubins2D}(\mathbf{q}_i, \langle x_g, y_g, \theta_g\rangle, \rho_h)).$$

A bisection search is used to find the angle ϕ^* such that $\mathcal{L}(\phi^*)\tan\bar{\gamma} = \Delta_z$.

Given a 3D Dubins path parametrized by $D = \langle D_h, D_v, k, \phi\rangle$, we can interpolate along this path as follows. First, note that we can determine the type of path from the values of k and ϕ. If both are 0, then D is a low-depth path. If $k > 0$, then D is a high-depth path. If $\phi \neq 0$, then D is a medium-depth path. By construction, k and ϕ cannot both be non-zero. For a low-depth path we compute interpolated points $\langle x, y, \theta\rangle$ and $\langle s, z, \gamma\rangle$ along D_h and D_v and construct the corresponding 3D Dubins state $\mathbf{q} = \langle x, y, z, \theta, \gamma\rangle$. For the medium- and high-depth path, the calculation of z and γ remains the same. For medium-depth paths, we need to determine whether the interpolated point lies on the extra turn or on D_h. Similarly, for high-depth paths, we need to determine whether the interpolated point lies on the helical part or the D_h part of the path. Note that the helical part is not *exactly* a helix: the pitch angle is not achieved instantaneously if $\gamma_s \neq \bar{\gamma}$. However, this does not affect the calculation of x, y, and θ.

IV. Results

Both the medium- and high-depth change paths rely on a numerical one-dimensional search. We used the classic TOMS-748 algorithm [9], which is designed to find the root of a continuous function. However, the functions for which we are trying to find a root are only *piecewise* continuous. This

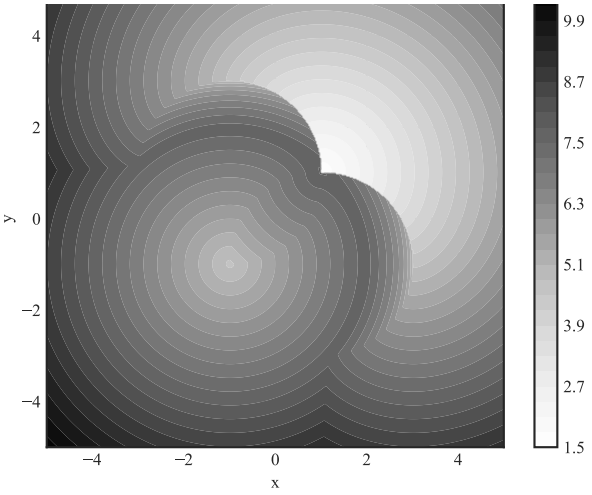


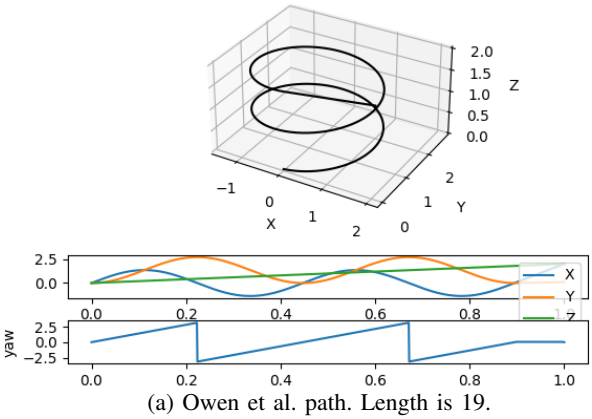
Fig. 2. Contour plot of the length of a 2D Dubins curve connecting $\langle 0, 0, 0\rangle$ and $\langle x, y, \pi/2\rangle$ for all $-5 \leq x, y \leq 5$. Note the discontinuities in the top-right quadrant: the abrupt change in shade of gray indicates that the Dubins path type changed which resulted in a discontinuity in the path length.

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT 3D DUBINS PATH ALGORITHMS

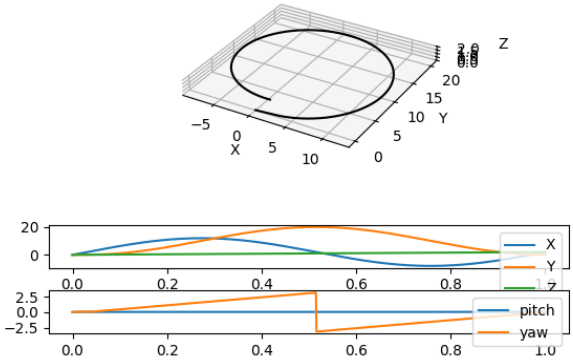
Algorithm	**Average time (μs)**	**Success rate**
Owen et al. [7]	2.9	98.6%
Váňa et al. [8]	40	100%
Hybrid Váňa–Owen (this paper)	195	100%

is due to the fact that the 2D Dubins distance function is only piecewise continuous. Small changes in either the turning radius or the relative pose between start and goal can result in large changes in distance. Figure 2 illustrates this point. It shows a contour plot of Dubins distance between $\langle 0, 0, 0\rangle$ and $\langle x, y, \theta\rangle$, where $\theta = \pi/2$. In rare cases, this can cause the numerical search to fail to find a solution. This affects both the original Owen algorithm as well as the Váňa–Owen algorithm. However, the Váňa–Owen algorithm guards against these rare failures by performing a numerical search over ρ_h in the outer loop. If Algorithm 2 fails to find a solution due to a root-finding failure, a slight change in ρ_h in Algorithm 1 is sufficient to avoid such a failure. In extensive testing we have not seen Váňa–Owen algorithm fail, while in our implementation the Owen algorithm fails to find a solution 1.4% of the time. The robustness of the Váňa–Owen comes at a computational price as shown in Table I. This table shows the average time it takes to compute a 3D Dubins path between $\mathbf{q}_s = \langle 0, 0, 0, 0, 0\rangle$ and a randomly sampled $\mathbf{q}_g$ such that $-10 \leq x, y, z \leq 10$. The turning radius and pitch range were set to $\rho_{\min}$ and $\Gamma = [-0.1 \text{ rad}, 0.1 \text{ rad}]$. Note that [7] does not respect pitch constraints and is, therefore, solving a simpler problem. There are likely further improvements possible in our implementation of all three algorithms by tuning the bracketing of the root finding algorithms and algorithmic parameters to determine convergence to a locally optimal solution path.

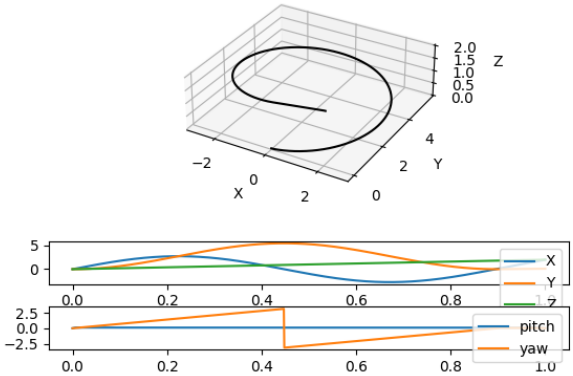
The 3D Dubins models described above have all been implemented using the Open Motion Planning Library (OMPL) [10], a library containing implementations of many sampling-based algorithms[1]. It already contains an efficient implementation of 2D Dubins curves. The 3D Dubins motion models are used to define a new state space, where distance between states is defined by the length of the corresponding 3D Dubins path and where interpolation is performed along such paths. (Note that the 3D Dubins path length is not a proper metric, but many



(a) Owen et al. path. Length is 19.



(b) Váňa et al. path. Length is 65.



(c) Path using the new hybrid model. Length is 19.

Fig. 3. 3D Dubins paths connecting the same start and goal poses. Note the different ranges on the X- and Y-axes. The start position is at $(0, 0, 0)$, while the goal is at $(2, 0.1, 2)$. The yaw and pitch are both 0 at the start and goal. The maximum pitch is set to 0.1 radians ($\approx 6°$).

[1]Code is available at https://github.com/ompl/ompl.

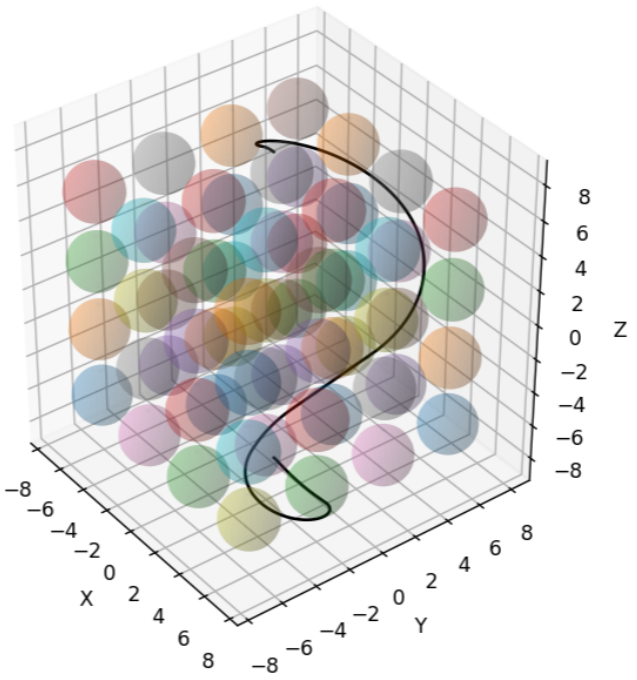


Fig. 4. Using a 3D Dubins paths with a sampling-based planner.

of OMPL's planners do not require one.) Figure 3 shows an example of the path differences. The new 3D Dubins model usually finds paths that are similar in length to the Owen et al. paths, but ensures smooth changes in pitch. It also finds shorter paths than the Váňa et al. model when depth changes are large. Note the different scales on the X and Y axes.

V. Integration with Sampling-Based Planners

The 3D Dubins algorithms can be thought of as *local* planners that can be used as primitives within a *global* planner that also considers collisions and other hard constraints. Figure 4 shows an example of a collision-free path composed of a series of waypoints connected by 3D Dubins paths as computed by the Váňa–Owen algorithm. Here, $\mathbf{q}_s = \langle 0, 0, -9, 0, 0\rangle$, $\mathbf{q}_g = \langle 0, 0, 9, 0, 0\rangle$, $\rho_{\min} = 2$, and $\Gamma = [-0.5 \text{ rad}, 0.5 \text{ rad}]$. The obstacles in this scene are spheres with a radius of $\frac{3}{4}\rho_{\min}$ placed at coordinates $\langle i\rho_{\min}, j\rho_{\min}, k\rho_{\min}\rangle$ for $i, j, k = \pm 1, \pm 3, \pm 5, \ldots$

The 3D Dubins algorithms can be used by many of the sampling-based algorithms in OMPL. The path shown in Figure 4 was computed with Anytime Path Shortening [11], a meta-planner that was configured to use four instances of the EST algorithm [12] and four instances of the KPIECE algorithm [13]. Anytime Path Shortening repeatedly runs these sampling-based planners in separate threads, collects the solutions paths, applies path short-cutting [14], and returns the best solution after a stopping criterion is met (a time limit in this case). The resulting path converges to the shortest path as a function of planning time. For illustrative purposes we opted for planning algorithms that do no rely heavily on a distance measure (since the 3D Dubins distance is not a metric). Other sampling-based planners like RRT [15] can be used as well, however.

OMPL has a concept of a motion validity checker, which is used to verify whether a path between two states (computed by a local planner) satisfies all hard constraints. Typically, a motion validity checker performs collision checking on a path and returns true if and only if the path is collision-free. In our case the local planner can (potentially) fail due to failures in the bisection search, as was observed in the Owen algorithm. We have augmented the motion validation step to catch such failures. This same mechanism can also be used to incorporate (analytic) Dubins 3D methods that do not handle pitch constraints and simply check in the motion validation step whether two waypoints are far enough apart in the XY plane that a resulting path is feasible. However, a sampling-based planner might end up spending significant time trying to connect states that are too close in the XY plane. It may be possible to address this with custom state sampling techniques, but this can weaken the theoretical guarantees on completeness and optimality of a planning algorithm.

VI. Discussion

We presented a practical approach to computing 3D paths for underwater vehicles (and aerial vehicles) such that the curvature and pitch angle along the path are bounded. These paths have a simple parametrization that facilitates the design of a tracking controller. The planner proposed here builds on two prior methods for computing 3D Dubins paths and combines the best properties of both by producing short paths that satisfy curvature and pitch rate constraints.

Initial results showed it is possible to use the 3D Dubins path planner as a local planner for a sampling-based planner. However, we have not performed a careful analysis on the requirements this local planner imposes on a global sampling-based path planner to ensure probabilistic completeness or convergence to optimality. It is still poorly understood how a distance measure that is not a metric impacts previously proposed sampling-based planners.

Acknowledgment

We would like to thank Artur Wolek and Collin Hague at the University of North Carolina at Charlotte for initial discussions of existing 3D Dubins models that provided the inspiration for this work.

References

[1] 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.

[2] A. M. Shkel and V. Lumelsky, "Classification of the Dubins set," *Robotics and Autonomous Systems*, vol. 34, no. 4, pp. 179–202, 2001. [Online]. Available: http://dx.doi.org/10.1016/S0921-8890(00)00127-5

[3] H. Sussmann, "Shortest 3-dimensional paths with a prescribed curvature bound," in *Proceedings of 1995 34th*

IEEE Conference on Decision and Control, vol. 4, Dec. 1995, pp. 3306–3312 vol.4.

[4] S. Hota and D. Ghose, "Optimal path planning for an aerial vehicle in 3D space," in *49th IEEE Conference on Decision and Control (CDC)*, Dec. 2010, pp. 4902–4907.

[5] V. M. Baez, N. Navkar, and A. T. Becker, "An Analytic Solution to the 3D CSC Dubins Path Problem," in *IEEE Intl. Conf. on Robotics and Automation*, Yokohama, Japan, 2024.

[6] H. Chitsaz and S. M. LaValle, "Time-optimal paths for a Dubins airplane," in *46th IEEE Conference on Decision and Control*, Dec. 2007, pp. 2379–2384.

[7] M. Owen, R. W. Beard, and T. W. McLain, "Implementing Dubins airplane paths on fixed-wing UAVs," in *Handbook of Unmanned Aerial Vehicles*. Springer, 2014, pp. 1677–1701.

[8] P. Váňa, A. Alves Neto, J. Faigl, and D. G. Macharet, "Minimal 3D Dubins Path with Bounded Curvature and Pitch Angle," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 8497–8503.

[9] G. E. Alefeld, F. A. Potra, and Y. Shi, "Algorithm 748: Enclosing zeros of continuous functions," *ACM Transactions on Mathematical Software (TOMS)*, vol. 21, no. 3, pp. 327–344, 1995.

[10] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, Dec. 2012. [Online]. Available: http://dx.doi.org/10.1109/MRA.2012.2205651

[11] R. Luna, I. A. Şucan, M. Moll, and L. E. Kavraki, "Anytime solution optimization for sampling-based motion planning," in *IEEE Intl. Conf. on Robotics and Automation*, 2013, pp. 5053–5059. [Online]. Available: http://dx.doi.org/10.1109/ICRA.2013.6631301

[12] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," *Intl. J. of Computational Geometry and Applications*, vol. 9, no. 4-5, pp. 495–512, 1999.

[13] I. A. Şucan and L. E. Kavraki, "Kinodynamic motion planning by interior-exterior cell exploration," in *Workshop on the Algorithmic Foundations of Robotics*, Dec. 2008.

[14] R. Geraerts and M. Overmars, "Creating high-quality paths for motion planning," *Intl. J. of Robotics Research*, vol. 26, no. 8, pp. 845–863, 2007. [Online]. Available: http://dx.doi.org/10.1177/0278364907079280

[15] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Intl. J. of Robotics Research*, vol. 20, no. 5, pp. 378–400, May 2001. [Online]. Available: http://dx.doi.org/10.1177/02783640122067453