Time-optimal motion of spatial Dubins systems

Weifu Wang[1] and Devin Balkcom[2]

[1] University at Albany; [2]Dartmouth College

Abstract. This paper presents a generic numerical approach to find the kinematic time-optimal trajectories for 3D rigid bodies with a finite set of translation, rotation, and screw controls, in an obstacle-free space. First, geometric necessary conditions for time-optimality are derived. Second, a method is presented for sampling trajectories satisfying the necessary conditions sufficiently densely to guarantee that for any start configuration and goal location, a trajectory can be found that provably approximately reaches the goal, approximately optimally.

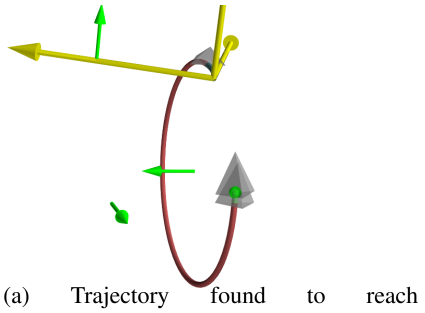
1 Introduction

This paper studies time-optimal motion of 3D rigid bodies that are permitted actions from a finite set of constant-velocity rotations, translations, and screws; we call these rigid bodies *spatial Dubins* systems, as the known optimal motion of the planar Dubins car follows from the 3D geometry derived. We admit that spatial Dubins systems are not quite a direct generalization, since the set of controls is finite; however, for the planar Dubins car, a finite number of controls does turn out to be sufficient for optimality.

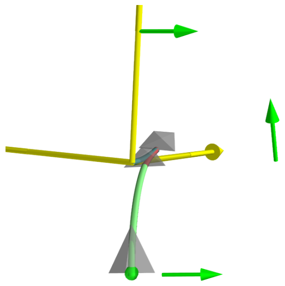
We have recently presented the derivation of basic necessary conditions in [3]; the current paper summarizes these results briefly. The paper then presents the new analysis of the geometry of these conditions applied to spatial rigid bodies. This geometry is used to further analyze a few interesting systems. The paper also presents an algorithm that searches for approximately optimal trajectories across all points in a bounded region of the space. Figure 1 shows a few trajectories for each system found using this algorithm.

Optimal trajectories may be used directly for robot control or as components for planning in more complex environments, such as the steering methods and local planners used in sampling-based planning algorithms [20, 19, 4]. Geometric understanding of the optimal trajectories may also influence the design of mechanical systems or planning algorithms, such as work on steerable medical needles [2] that uses planar Dubins curves as motion primitives.

Trajectory time is an obvious quantity to minimize, and for a bounded-velocity system, the time-optimal trajectories are a simple generalization of shortest paths for points on the robot. A typical approach to discovering the structure of time-optimal trajectories for a system or class of systems is the use of Pontryagin's Maximum Principle. System equations are manipulated to write certain *adjoint* differential equations; if those equations may be integrated, they yield geometric conditions that optimal trajectories must satisfy. Further work with the geometry may eliminate some trajectory structures, or allow a mapping from the configuration space to trajectory structure – the optimal control synthesis.



(a) Trajectory found to reach $(-0.4,-0.3,-0.7)$ using Dubins airplane following controls yaw(left)-pitch(down)-yaw(left). Euclidean distance 0.86, and the duration of the trajectory is 5.47.



(b) Trajectory found to reach $(-0.4,-0.3,-0.7)$ using 3D Reeds-Shepp system following controls pitch(forward-up)-yaw(back-right)-pitch(back-down). Euclidean distance 0.86, and the duration of the trajectory is 1.64.

Fig. 1: Sample trajectories found by the proposed algorithm, one for Dubins airplane, and one for 3D Reeds-Shepp model.

Generalizing time-optimal motion beyond the plane has proven challenging. Our own prior approaches to discovering optimal trajectories for 3D rigid bodies were halted by our inability to integrate the adjoint equations; representations of rotations in 3D as matrices, quaternions, or Euler angles each caused some difficulty. Work by Chitsaz *et al.* [9] presents some partial results and necessary conditions for a 3D Dubins airplane that includes bounded velocity altitude changes; optimal motion primitives are derived, but the configuration space is not that of a true 6DOF rigid body, as the plane effectively remains flat, sidestepping the issue of rotation representation.

We avoid the issue of finding analytical solutions to complicated differential equations in a different manner in this paper, by choosing a finite set of control actions for the 3D rigid body. These actions are taken to be rigidly attached to the local frame of the robot. For example, for an airplane, one might take an action to be a *pitch up* corresponding to rotating the airplane about a rotation axis directly above the robot and perpendicular the direction of travel. We further assume that the optimal trajectory will be a finite sequence of such actions – potentially, a very problematic assumption, which we will discuss shortly.

Restricting the search to finding durations and a sequence of actions from a finite set allows the problem to be phrased as a constrained optimization problem. Karush-Kuhn-Tucker (KKT) conditions allow simple necessary conditions to be written directly, without the need for integration of the adjoint. Analysis in [3] reveals an interesting result quite parallel to those known for the planar systems: time optimal trajectories can be viewed as virtual robot arms in special configurations that balance some external force described by a vector λ; revolute arm joints correspond to rotation actions, and prismatic joints correspond to translation actions. Choosing different λ vectors gives different trajectory structures, including different sequences of actions and action durations.

Unfortunately, the KKT-derived conditions do not provide some of the information about trajectory structure (in particular, the sequence of actions) that the PMP conditions do for the planar case, necessitating the use of additional discrete search in the described algorithm. The trajectory search algorithm we present explores the trajectory space in an in a branch-and-bound fashion, using constraints provided by λ to prune the search. As the algorithm runs, a numerical representation of the synthesis of approximately optimal trajectories is discovered. This algorithm is an extension to 3D of an algorithm for approximating planar optimal trajectories presented in [30]. In addition to the extension to 3D, the current algorithm works with the weaker conditions derived using KKT.

We hope that the numerical techniques and geometric insights presented in this paper prove useful, but must point out several weaknesses in the approach. Chief among the weaknesses is the possibility that for any given system, optimal trajectories do not exist. Typically, for each new system, existence must be considered carefully. One potential issue is that for certain goal configurations, for any given trajectory, there exists another trajectory that is faster, but which requires a greater number of switches between controls: the phenomenon of chattering. While it is common to discretize the action space before using a planner such as RRT* [18], doing so may well introduce chattering. For example, choosing only the left and right turning actions for a Dubins car as discrete actions (and not including pure forward translations) leads to a system that moves forwards most quickly by rapidly alternating between the two rotation actions.

If there is some fixed cost to switching between actions, then optimal trajectories do exist for any reachable configuration [22, 21]. We consider the assignment of some small switching cost to be at least as reasonable a model as a cost of zero for switches; effectively, a zero-cost switch requires infinite accelerations of the physical system. The algorithm in the present paper works without modification for this model of costly switches. As long as the described algorithm is applied to a system with (possibly small) switching costs, we can expect that the trajectory found is close to optimal, subject to the resolution of the search.

Further weaknesses include the fact that although the rigid bodies we consider have orientation, we do not restrict the orientation at the goal. We also point out that the kinematic model of rigid bodies is not a good model for airplanes, which have dynamics and thrust. Nonetheless, we hope that short or fast trajectories for this model are useful for the basic geometry they provide, as they have proven to be for the planar Dubins particle. Finally, this paper does not provide any particular insight about how to select a finite control set; for the systems studied, we choose simple extreme controls that are analogous to those that have turned out to be sufficient for planar systems.

2 Related work

Analytical time-optimal trajectories are known for several planar mobile robots, including the Dubins car [14], the Reeds-Shepp car [23, 8, 27, 24], bounded-velocity differential-drive robots [6, 10], and omnidirectional robots [5, 29], all in the unobstructed plane. These systems are all planar rigid bodies with different velocity constraints; Furtuna [16, 15] showed that this similarity allows derivation of necessary conditions on the time-optimal trajectories that apply to all of these models; a unified theory of time-optimal motion for simple models of planar mobile robots.

Few analytical solutions have been derived for dynamic models of systems. In fact, some systems with bounded accelerations do not even have time-optimal trajectories [25, 26]. Beyond land and air aerial vehicles, time-optimal trajectories for ships under constant current [13] and underwater vehicles has also been studied [11]. To simulate the effect of acceleration, a cost can be introduced between each switch of control. Lyu used Blatt's Indifference Principle (BIP) [7], Lyu *et. al.* [22, 21] to study and find approximation algorithms for optimal trajectories for a costly switch model for planar Dubins-like systems.

Most of the described work on time-optimal strategies does not consider the existence of obstacles; notable exceptions include work by Vendittelli *et. al.* on how to measure the distance between a car-like robot and obstacles [28, 17]. Planning among simple obstacles has also been studied for simple car-like systems [1, 12].

3 Model of constant-control rigid bodies

A Dubins car in the plane with a maximum turning radius of 1 may be considered to have forward velocity v of 1 and an angular velocity ω in the range $[-1,1]$. As Dubins proved, only three discrete controls out of this continuum are required for optimality: $\omega \in \{-1,0,1\}$. Six discrete controls are required for the Reeds-Shepp car, with $v \in \{-1,1\}$. We may view controls for the planar systems as actions attached to the local frame of the robot. For example, the left-turn action for a Dubins car has associated with it a rotation center to the left of the robot; as the robot moves, the rotation center moves with the local frame. Furtuna's work [15] showed that many other mobile robot models also are equivalent to rigid bodies in the plane with body-fixed rotation and translation controls, only a finite subset of which are required to find an optimal trajectory to each reachable configuration.

We extend this idea to spatial rigid bodies. We attach a small number of constant controls to the local frame of the robot. Each control is either a translation, a rotation about an axis fixed to the local frame, or a screw of constant pitch around an axis fixed to the local frame. We call such a system with constant controls attached to the local frame a spatial Dubins system.

We consider an example spatial Dubins system that might be called a *Dubins airplane* or submarine. We attach the following frame to the robot: x-axis is along the *nose* of the vehicle and the y-axis is along the left wing of the vehicle and is perpendicular to x-axis; the z-axis is perpendicular to both x and y axis and follows the right-hand rule. The vehicle can translate forward along the x axis of the robot frame at speed 1. In addition to translation, the vehicle can perform the following six controls:

- **pitch up or down**: unit-speed rotation around one of two axes, each parallel to y-axis and offset by 1 along the z axis;
- **yaw left or right**: unit-speed rotation around one of two axes, each parallel to the z-axis and offset by 1 along the y axis;
- **roll left or right**: unit-speed rotation around the x axis of the robot frame while translating along the x-axis at speed 1 (a screw).

One might question whether this model is reasonable for a jet airplane. It is not. Nonetheless, constant actions selected from the tangent space to $SE(3)$ may generate geometrically interesting paths that are in some sense short and may be useful for systems such as steerable needles or submarines.

4 Necessary conditions for time-optimal trajectories

In [3], we derived the necessary conditions for this model of controls for a spatial rigid body. The model may be termed a spatial Dubins model, but is also quite similar to the kinematic model of a robot arm. Specifically, an arm is a series of joints offset by links; the joints typically provide translations along or rotations about axes fixed to the prior link. The primary difference for the current system is that unlike the arm, the particular sequence of joints must be selected as well as the duration.

4.1 The constraint Jacobian

We require that the robot reach a specified goal location in minimum time T, such that at $t = T$, we have three constraint equations: $x(t) = x_g$, $y(t) = y_g$, and $z(t) = z_g$. Let the duration of the ith control applied in the trajectory be t_i. Then, for any given sequence of controls, $x(t)$, etc. are functions of $t_1 \dots t_k$. As shown in [3], this problem is thus a constrained minimization problem, for which necessary conditions may be found using Lagrange multipliers. For any time-optimal trajectory, there exists constants λ_1, λ_2, and λ_3 such that:

$$\begin{pmatrix} \partial x/\partial t_1 & \partial y/\partial t_1 & \partial z/\partial t_1 \\ \partial x/\partial t_2 & \partial y/\partial t_2 & \partial z/\partial t_2 \\ & \vdots & \\ \partial x/\partial t_n & \partial y/\partial t_n & \partial z/\partial t_n \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}. \tag{1}$$

4.2 Geometric interpretation of necessary conditions

Since for a given sequence of control actions, $x(t)$, $y(t)$, and $z(t)$ are equivalent to kinematics equations for an arm of a particular design, the Jacobian in equation 1 is equivalent to the Jacobian for the same arm. In fact, as pointed out by [3], equation 1 is equivalent to a force-torque balance equation for the arm.

The similarity of the trajectory to the kinematics of an arm motivates an observation: for each rotation action, the corresponding column of the Jacobian for an arm, or row of the matrix in equation 1, may be found by taking the cross product of a vector pointing along the rotation axis with the vector from the current joint to the end effector. If the ith control is a rotation, let the axis be ω_i, and let the vector pointing from the axis ω_i to the end-effector g be $\overrightarrow{o_i g}$. Then:

$$\begin{pmatrix} \partial x/\partial t_i \\ \partial y/\partial t_i \\ \partial z/\partial t_i \end{pmatrix} = \omega_i \times \overrightarrow{o_i g} \tag{2}$$

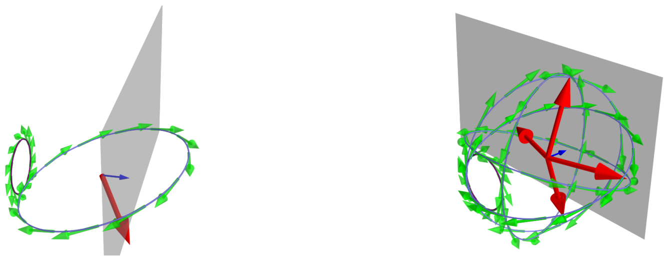
at any instant during control i. For translations, denote the velocity vector of the ith control as $\overrightarrow{v_i}$. Then $v_i \cdot \lambda = 1$. We will use this geometry to place constraints on the locations of rotation axes or lines of translation for controls; time-optimal trajectories must satisfy these constraints.

4.3 Geometric relationship between λ and trajectory structure

The main algorithm in this paper will iterate over durations and structures of the first few controls in the trajectory. Each iteration generates a particular geometry for the first few segments of the trajectory, with which only a single value for λ is consistent. That value is computed and then used to constrain the structure of the rest of the trajectory. Searching over consistent trajectory structures provides candidate optimal trajectories to various configurations. In the two steps of this process, the relationship between the potential λ values and the duration and structure of the trajectory will be critical; we now discuss this relationship.

Knowing λ informs us about potential trajectory geometries. Since the differential motion of the endpoint due to varying a control duration must make a unit dot product with λ, that motion vector must lie on a plane perpendicular to λ. For a translation control, this indicates that the translational velocity must lie on this plane. For a rotation control, the differential motion of the endpoint is generated by a cross product; specifying λ constrains orientation and distance to the goal of the rotation axis. Figure 2 illustrates some possible rotation axes (green arrows) for a given example λ (blue vector).

On the other hand, the geometry of a trajectory constraint λ values. Knowing the location and the orientation of one control limits λ to a plane, knowing two controls limits λ to a line, and a third control limits λ to a single vector value.



(a) A few rotation control axes (green arrows) that satisfy the necessary condition for a given λ (blue vector) and a fixed distance between the rotation axis and the goal.

(b) Several rotation control axes (green arrows) that can satisfy the necessary condition for a given λ and a fixed distance between the rotation axis and the goal.

Fig. 2: Relations between rotation controls and λ. λ is in blue, and a red vector is the cross product vector that satisfies the necessary condition; rotation axes are drawn in green.

Figure 2 shows a few rotation axes consistent with a particular λ vector; to simplify the figure, all of these axes are the same distance from the goal, yielding a sphere such that rotation axes are tangent the sphere. As the distance to the goal varies, the radius of the sphere changes; we omit some details. There are some further constraints on the placement of rotation axes. Specifically, there is a cylinder with the axis parallel to λ within which no rotation axis will occur. This occurs because any particular control switch that is followed by a fixed trajectory geometry cannot reach certain goals that are too close to that switch.

5 Finding time-optimal trajectories

The basic strategy we will pursue to find a near-optimal trajectory between a given start and goal is the following:

1. Loop over all possible triplets of actions for the first three controls in the trajectory.
2. For each triplet of actions, iterate over densely sampled durations for the first two actions.
3. Compute the unique λ vector consistent with the current trajectory structure and first two durations.
4. For the current λ vector, compute the fastest trajectory that reaches the goal.
5. Choose the fastest trajectory among all computed λ vectors.

In this section, we will discuss the details for each step. Why iterate over sampled trajectory durations and partial trajectory structures in steps 1 and 2? Why not just densely sample λ and search over λ-space? As we will discuss below, λ is not bounded. Further, it was shown in [30] that by carefully selecting the sampling rate for durations of a few of the controls, any final goal can be approximately reached with approximately optimal cost.

5.1 Finding λ given partial trajectory structure and durations

Since λ has three elements, we need three constraint equations to compute a λ vector, and thus we need to know three controls. Let the portion of the trajectory corresponding to the ith control be its ith *segment*. Given a endpoint g, define the *tangential velocity* of a given control as the sum of the translation velocity and the cross product between the rotation axis and the vector pointing from the rotation axis to g. Depending on the control type, a tangential velocity may have zero rotational or translational component. Two controls are considered *co-linear* if they have the same tangential velocity.

Lemma 1. *Given a endpoint g, three trajectory segments and their configurations in space, if the three corresponding controls are not co-linear, then either these three segments cannot be on the same time-optimal trajectory, or a λ vector can be uniquely computed.*

Proof. Given the ith control, the tangential velocity must have dot product 1 with λ vector in order to satisfy the necessary condition. Let there be a 3×3 matrix A with three different rows where each row is the tangential velocity for one of the controls corresponding to the three segments. If the controls are not co-linear, then matrix A is of full rank. Therefore, the equation $A \cdot \lambda = b$ where $b = [1,1,1]^T$ has one or zero solutions. □

Since three arbitrary segments can be used to compute λ vector as long as they are not co-linear, let us consider the first three segments for simplicity. The reason to choose the first three segments is that only two durations need to be known to generate the configurations of the three segments. When some of the first three controls are co-linear, we need to consider later segments.

For a rotation control, the duration is bounded in the range $[0, 2\pi)$, but no such natural bound exists for translation. However, if some (non-optimal) method exists to compute some trajectory to any goal, then the cost of reaching that goal can be used as an upper bound for the duration of a single translation section, yielding the following lemma:

Lemma 2. *If there exists an arbitrary trajectory that can reach a given endpoint g, then it is sufficient to search a bounded set of durations to compute the unique λ vector corresponding to any given trajectory structure of length no less than three.*

Given Lemma 2, the goal is to be able to iterate over the first few trajectory segments to compute the corresponding λ. Therefore, the key is to find a trajectory to serve as the upper bound. However, there do exist systems so that it is not easy to find any trajectory to reach a given endpoint. We will not focus on such systems.

5.2 Screw actions

A screw action for Dubins airplane (roll) is a combination of rotation and translation. In order for such a screw action to satisfy the necessary condition, the following equation must hold, given the goal g, the translation velocity v, the rotational component of the tangential velocity $\omega \times \overrightarrow{og}$.

$$(\omega \times \overrightarrow{og} + \overrightarrow{v}) \cdot \lambda = 1 \tag{3}$$

Along the trajectory, λ will not change, and v will not change. Therefore, in order to have a constant dot product with λ, the rotation axis of a roll must go through the goal. Because the translation direction of the roll shares the same direction vector with ω, this means that the translation velocity also goes through the goal.

For the spiral type of screw actions, because there is just a single rotation axis and no translation velocity, the necessary condition can be satisfied if the tangential velocity remains constant. Therefore, such screw actions are almost like rotations, except the duration is not bounded by 2π.

5.3 Fastest trajectory for a given λ

Given each feasible λ, we need to find the fastest trajectory that satisfies the necessary conditions. At each switch of controls, there may exist multiple controls that can satisfy the necessary condition. Then, all the possible trajectory structures corresponding to a specified λ forms a tree with unknown depth and potential large branching factor. However, one interesting observation is that many of the trajectory structures cannot lead the robot to the endpoint. Therefore, only a portion of the tree needs to be searched to find the fastest trajectory for a given λ. What is more, if we have already found a trajectory with duration T that can reach the goal, the candidate fastest trajectories cannot be longer than T, which further reduce the portion of the tree that needs to be searched.

For a Dubins airplane, it is easy to find a control sequence that can reach the goal. Let us consider a *drive-turn-drive* sequence. We first execute a roll control so that g is either on x-z plane or x-y plane, then find a trajectory within that plane that will lead the robot to the goal following a *drive-turn-drive* strategy. The same strategy can also be used to find a trajectory to reach any goal in 3D for the Reeds-Shepp system. The resulting trajectory with duration T will serve as the upper bound.

We, therefore, can use a Depth First Search (DFS) to find the fast trajectory for a given λ and terminate if any trajectory exceeds duration T. Whenever a new trajectory is found that can reach the goal and is shorter than the current solution, the upper bound will be updated, so that fewer and fewer branches need to be explored. What is more, this upper bound also reduces the region of durations we need to explore to find candidate λ vectors.

5.4 Complete procedure

Below is the complete procedure that we use to find the time-optimal trajectory for a given pair of start and goal.

Algorithm 1: Find time-optimal trajectory given a goal position

```
Input: g(oal) $T \leftarrow$ Find a trajectory that can reach $g$;
for All possible one segment or two segment trajectories that can reach $g$ do
    Update the upper bound $T$ to the fastest trajectory;
$k \leftarrow 3$;
for All possible sequence of first $k$ controls do
    for All possible durations for first $k-1$ controls that do not exceed $T$ do
        if The some of the first $k$ controls are co-linear then
            $k \leftarrow k+1$;
            continue;
        Compute corresponding $\lambda$;
        Starting from the third control, compute next possible control;
        Run DFS until reach goal, or exceeds upper bound;
        if Found trajectory can reach goal shorter than $T$ then
            Update the upper bound $T$;
return the upper bound $T$ to be the time-optimal trajectory;
```

One step we have not yet discussed in Algorithm 1 is the need to loop over all possible one or two segment trajectories. This is because in the main search, we need three controls to compute a corresponding λ, but we cannot rule out the situations where one or two controls can reach the goal. This procedure can be computed easily in closed form. In fact, we can compute in closed form whether any trajectory containing no more than three segments can reach the goal.

Algorithm 1 involves looping over the durations for the first $k-1$ linear independent controls. Given an arbitrary resolution, the returned trajectory may not even share the same structure with the time-optimal trajectory. So, what is the sufficient resolution? In the next section, we will show that given arbitrary tolerance ε, there exist a sampling resolution so that the resulting trajectory will reach a location g' where $|g'-g| < \varepsilon$.

5.5 Approximation theorems

We will show that given any two trajectories of the same structure, i.e. control sequence, if the duration of each segment is similar, then at any time the two trajectories will reach locations that are close (measured by Euclidean distance). We extend Lemmas and Theorems from [30] to 3D.

Lemma 3. *Consider two spatial trajectories Y and Y' with identical structure, equal duration for all segments but one translation segment k, and a small distance κ. The corresponding end points are G and G'. For any translation control k in Y with duration t_k, and the same control k in Y' with duration t'_k, if $|t_k - t'_k| < \frac{\kappa}{v_k}$, $\|G - G'\| < \kappa$.*

Proof. Translation is commutative. □

Lemma 4. *Consider a spatial trajectory Y with end point G, a small angle σ, and two pints P and Q on the trajectory, with $\|P - G\| > \|Q - G\|$. Form a new trajectory Y_1 with endpoint G_1 by rotating the trajectory from P to G around P by δ, and form another trajectory Y_2 with endpoint G_2 by rotating from Q to G by σ. Then, $\|G_2 - G\| < \|G_1 - G\|$. What is more, for any trajectory Y_k with end point G_k achieved by rotating the trajectory Y around a series of points along the trajectory with σ angle in total, $\|G_k - G\|$ is upper bounded by only rotating σ around the furthest point from G on Y.*

Proof. If the trajectory only moves in a plane, the proof from [30] directly applies. When the trajectory moves beyond a plane, we will show that the same conclusions still hold.

First, between trajectory Y_1 and Y_2, formed by rotating around P and Q on trajectory Y, since each rotation is still within a single plane, even though the two rotations may be around two different planes, the result holds that if P is further from G than Q, then G_1 is further from G than G_2.

Now, let us consider a trajectory Y_k formed by a sequence of rotations around points on Y by a total amount no larger than σ, where the rotations may not be on the same plane. Denote the distance the endpoint moved by the rotation around a point X as $d(G^X)$. No matter on which plane is the rotation, $d(G^X)$ is the same for the same rotation angle. Therefore, for two given rotations around points A and B, the endpoints moved by $d(G^A)$ and $d(G^B)$ respectively. If the two rotations are not in the plane, the total movement of endpoint G can be computed as $d(G^A) \cdot d(G^B) \cdot \cos(\theta)$ where θ is the angle between the two movement vectors. The angle θ reaches the maximum when the two movements are on the same plane. Therefore, given the trajectory Y_k formed by a sequence of rotations around points on Y, $\|G_k - G\|$ is upper bounded by the cases when all rotations are in the same plane, which is then upper bounded by rotating σ around the furthest point on Y to G. □

The screw controls are bounded by its rotational components, so Lemma 4 applies. We therefore can also directly extend the following theorem directly for 3D.

Theorem 1 (Theorem 3 from [30]). *Consider two spatial trajectories Y and Y' with the same control sequence, both starting at S and having time cost $t = \sum_{i=1}^{n} t_i$ and $t' = \sum_{i=1}^{n} t'_i$ respectively. Denote the point that Y' passes through at t' as G', and the point* *that Y passes at time t as G. Then, for any $\varepsilon > 0$, there exist $\delta > 0$ such that if $\sum_{i=1}^{n} |t'_i - t_i| < \delta$, then $\|G' - G\| < \delta$. Specifically, let δ be the minimum of $\frac{\varepsilon}{v^T_{\max}}$ and $\frac{\sqrt{(\omega_{\max} t_{\max} v^C_{\max}/2 + v^C_{\max})^2 + 2\omega_{\max} v^C_{\max} \varepsilon} - (\omega_{\max} t_{\max}/2+1) v^C_{\max}}{\omega_{\max} v^C_{\max}}$*

Last but not least, we need to show that by sampling sufficiently small, the resulting trajectory can be arbitrarily close to the desired goal.

Theorem 2. *Consider two spatial trajectories Y and Y' with the same control sequence, where every control on both trajectories will maintain a constant dot product with a vector λ and λ' respectively. Let the durations of the first two linear independent controls be t_1 (t'_1) and t_2 (t'_2) on Y (Y'). There exist a most sensitive segment k on both trajectories with duration t_k and t'_k respectively, such that if $|t_1 - t'_1| < \Delta t$, $|t_2 - t'_2| < \Delta t$, and $|t_k - t'_k| < \Delta t$, then $\sum_{i=1}^{n} |t'_i - t_i| < n\Delta t$.*

Proof. First, we can show that because $|t_1 - t'_1| < \Delta t$ and $|t_2 - t'_2| < \Delta t$, λ and λ' are close both in magnitude and in orientation. Given the ith control, if the duration of this segment changes from t_i to t'_i, we can use the following equations to compute the location and orientation of the next control if control i is rotation.

$$(\omega_i \times \overrightarrow{c_i g}) \cdot \lambda = 1 \tag{4}$$

$$((|\omega_i| R(t_i) \omega_{i+1}) \times (\overrightarrow{c_i g} - |\omega_i| R(t_i) \overrightarrow{c_i c_{i+1}})) \cdot \lambda = 1 \tag{5}$$

If the ith control is translation, then the location of the next control can also be easily computed using the following equations,

$$v_i \cdot \lambda = 1 \tag{6}$$

$$(\omega_j \times (\overrightarrow{c_j g} - v_i \cdot t_i)) \cdot \lambda = 1 \tag{7}$$

Therefore, given the first three linear independent segments, which are computed using the durations for the first two linear independent controls, a λ can be computed. When the duration changes from t_1 (t_2) to t'_1 (t'_2), the tangential velocities change by a small amount both in magnitude and in direction, so the resulting λ and $\lambda' = \lambda + \Delta\lambda$ are similar both in scale and in direction.

Let us first consider the ith control being translation, and the $i+1$th control is rotation. Extending from equation 7 by replacing t_i and λ with $t_i + \Delta t_i$ and $\lambda + \Delta\lambda$ respectively, one can derive that Δt is a function of $\Delta\lambda$ scaled by $1/|v_i|$, we omit the derivation steps in this proof. But geometrically, because rotation axes are on the surface of concentric spheres, a translation connecting two rotation controls can be viewed as a displacement to form the correct distance between the rotation axis and the goal. This displacement can only change by a small amount when λ changes to $\lambda + \Delta\lambda$, so the slower the translation, the larger the Δt_i will need to be. If the $i+1$th control is also a translation, the switch can happen at any time and Δt can be arbitrarily small. Therefore, the slowest translation may need to change its duration the most given a small change in λ, thus is the most sensitive translation control.

$$|\omega_i| R(t_i) \cdot v_j \cdot \lambda = 1 \tag{8}$$

Let us then consider rotation control. Extending equation 5, one can similarly derive that Δt is a function of $\Delta\lambda$ scaled by $1/|\omega_i|$ if the next control is also a rotation. If the next control is a translation, one can extend equation 8 to derive that Δt again is a function of $\Delta\lambda$ scaled by $1/|\omega_i|$. Geometrically, recall that all rotation axis locates on the surface of spheres, but at each point, only one possible direction for rotation axis satisfies the necessary condition. Therefore, to form the appropriate tangential velocity for the next control, a similar distance needs to be covered by the current control when λ becomes $\lambda + \Delta\lambda$. Thus the rotational control with the smallest angular velocity is the most sensitive rotation action. Given a screw action, either the slowest translation component or slowest rotation component will dominate the sensitivity. Over the entire trajectory, the comparison among the most sensitive of each type yields the most sensitive control.

Therefore, given that the most sensitive segment and the first two linear independent segments' durations do not change more than Δt, all other segment durations will not change more than Δt, then the total duration difference is no larger than $n\Delta t$. $\square$

5.6 Synthesis of time-optimal control sequences

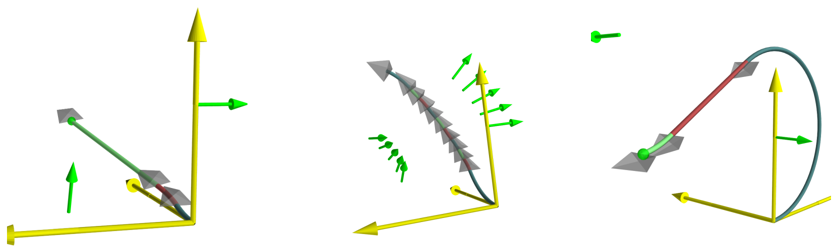
The proposed algorithm cannot find a closed form solution for the time-optimal trajectory between an arbitrary pair of start and goal. However, using the geometric interpretations of the necessary conditions, we can build a synthesis of all possible time-optimal control sequences for all goals, and store them for future inquiry. We can employ Algorithm 1 to find the time-optimal trajectory for each given goal position, but the reader can easily tell that there are many repeated computations for the same or similar λ, especially for nearby goals. So, we can simplify the procedure needed to find the synthesis.

Algorithm 2: Find synthesis of time-optimal control sequences to all goals

```
for All possible first control do
    for Sample first duration at resolution $\varepsilon$ do
        Compare all reachable points and current reaching duration; if Found a
          new shorter path then
            Update the shortest time and control sequence to this point;
        for All possible second control do
            for ... do
                ...;
```

6 Planning algorithm examples and results

Using Dubins airplane as an example, we conducted experiments to show how our algorithm can find the time-optimal trajectories for the Dubins airplane for a given resolution. Figure 3 shows the computed time-optimal trajectories for the given goal using Algorithm 1. The airplane is represented by the gray tetrahedron, the yellow frame is the



(a) Trajectory found to reach $(2, 0.57, 0.66)$ following controls pitch(up)-yaw(left)-translation. Euclidean distance 2.182, and the duration of the trajectory is 2.24.

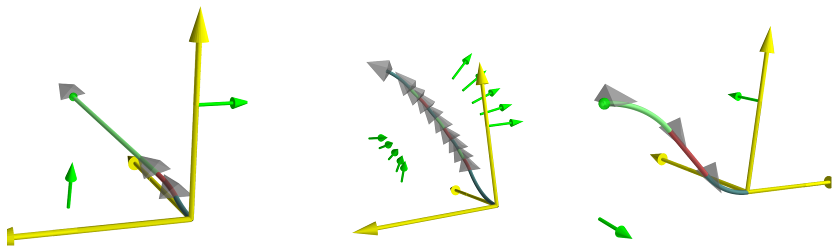
(b) Trajectory found to reach $(1, 1, 2)$ following controls pitch(up)-yaw(left) repeatedly. Euclidean distance 2.44, and the duration of the trajectory is 2.915.

(c) Trajectory found to reach $(-2, 0, 1)$ following controls pitch(up)-translation-pitch(down). Euclidean distance 2.23, and the duration of the trajectory is 5.297.

Fig. 3: A few trajectories found using Algorithm 1 for Dubins airplane to reach given endpoints.

world frame, and the green arrows are rotation axes. We implemented the procedure in python, running on a modern desktop. With a resolution of 0.03 for the durations, each search for the goal takes on the order of 300 seconds on a 2018 commodity desktop computer system.

In addition to the Dubins airplane, we also conducted experiments with a 3D Reeds-Shepp airplane, by adding symmetric reverse controls. To show the similarity and difference between Dubins airplane and Reeds-Shepp system, we ran the algorithm to find trajectories to reach the same set of goals; a few examples are shown in Figure 3, 5, and 1.



(a) Trajectory found to reach $(2, 0.57, 0.66)$ following controls pitch(up)-yaw(left)-translation. Euclidean distance 2.182, and the duration of the trajectory is 2.24.

(b) Trajectory found to reach $(1, 1, 2)$ following controls pitch(up)-yaw(left) repeatedly. Euclidean distance 2.44, and the duration of the trajectory is 2.915.

(c) Trajectory found to reach $(-2, 0, 1)$ following controls pitch(back-up)-translation(back)-pitch(back-down). Euclidean distance 2.23, and the duration of the trajectory is 2.36.

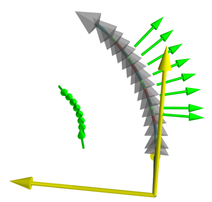
Fig. 4: A few trajectories found using Algorithm 1 for Reed-Shepp airplane to reach given endpoints.

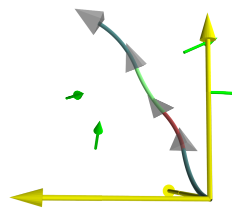
When the goal is along the positive x-axis, the resulting trajectory for Dubins and Reed-Shepp airplanes are the same; when the endpoint is along the negative direction of x-axis, the trajectory duration for Reeds-Shepp airplane uses some reverse actions.

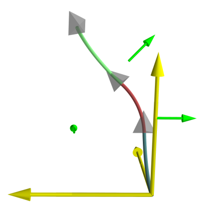
6.1 Trajectories with many switches

The proposed algorithm can also find trajectories with many segments if no switching cost is specified, as shown in Figure 3b and 4b. What is more, given different search resolutions for the trajectory duration, the number of segments may change, as shown in Figure 5a. At a resolution of 0.05, the fastest trajectory reaching $(1,1,2)$ has 10 segments. The number of segments increases to 16 when the resolution increase to 0.03.

We have not proven existence of optimal trajectories for Dubins systems, so the existence of trajectories with many switches is concerning. Fortunately, by adding a small switching cost, existence of optimal trajectories is guaranteed, and we expect much greater numerical stability for the search. For example, as shown in Figure 5b, by adding a 0.01 switching cost, the number of segments decreased to four from 16 at the same resolution of 0.03. With higher switching cost, the number of segments on the fastest trajectory found by Algorithm 1 decreases even further.


(a) Trajectory found to reach goal $(1,1,2)$ at resolution 0.03 with no switching cost. Euclidean distance 2.44, total duration of trajectory is 2.96 with 16 segments.


(b) Trajectory found to reach goal $(1,1,2)$ at resolution 0.03 with switching cost 0.01. Euclidean distance 2.44, total duration of trajectory is 3.02 with 4 segments.


(c) Trajectory found to reach goal $(1,1,2)$ at resolution 0.03 with switching cost 0.1. Euclidean distance 2.44, total duration of trajectory is 3.23 with 3 segments.

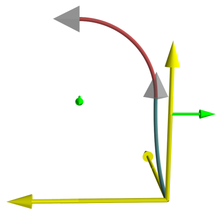

(d) Trajectory found to reach goal $(1,1,2)$ at resolution 0.03 with switching cost 0.3. Euclidean distance 2.44, total duration of trajectory is 3.24 with 2 segments.

Fig. 5: A few trajectories found using Algorithm 1 for Dubins airplane to reach $(1,1,2)$ with different switching costs.

7 Conclusions

In this work, we presented a generic numerical approach to find the kinematic time-optimal trajectories for 3D rigid bodies. We showed that in order for a trajectory to be time-optimal, there exist a λ vector so that some necessary conditions must be satisfied. The proposed approach starts by finding valid λ vectors and compare the fastest trajectory for each λ. We showed that for any given endpoint, the proposed approach can find a trajectory that reaches the endpoint within a given resolution and approximately optimal.

For future work, we would like to extend the current approach to find trajectories that can reach any given endpoint configurations rather than just locations. We would also like to further generalize the algorithms to include more generic systems. Also, we would like to explore how can the proposed approaches be integrated with existing planners to find efficient trajectories even with obstacles.

References

[1] Pankaj K. Agarwal, Jean-Claude Latombe, Rajeev Motwani, and Prabhakar Raghavan. "Nonholonomic path planning for pushing a disk among obstacles". In: *Proceedings of the 1997 IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico, USA, April 20-25, 1997*. 1997, pp. 3124–3129.

[2] Ron Alterovitz, Kenneth Y. Goldberg, and Allison M. Okamura. "Planning for Steerable Bevel-tip Needle Insertion Through 2D Soft Tissue with Obstacles". In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005, April 18-22, 2005, Barcelona, Spain*. 2005, pp. 1640–1645.

[3] Devin J. Balkcom, Andrei A. Furtuna, and Weifu Wang. "The Dubins car and other arm-like mobile robots". In: 2018.

[4] Devin J. Balkcom, Ajay Kannan, Yu-Han Lyu, Weifu Wang, and Yinan Zhang. "Metric cells: Towards complete search for optimal trajectories". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015, Hamburg, Germany, September 28 - October 2, 2015*. 2015, pp. 4941–4948.

[5] Devin Balkcom, Paritosh A. Kavathekar, and Matthew T. Mason. "Time-optimal trajectories for an omni-directional vehicle". In: *International Journal of Robotics Research* 25.10 (2006), pp. 985–999.

[6] Devin Balkcom and Matthew T. Mason. "Time optimal trajectories for bounded velocity differential drive vehicles". In: *International Journal of Robotics Research* 21.3 (2002), pp. 199–218.

[7] John M Blatt. "Optimal control with a cost of switching control". In: *The ANZIAM Journal* 19.3 (1976), pp. 316–332.

[8] Jean-Daniel Boissonnat, André Cérézo, and Juliette Leblond. "Shortest paths of bounded curvature in the plane". In: *Journal of Intelligent and Robotic Systems* 11.1-2 (1994), pp. 5–20.

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

[10] Hamid Reza Chitsaz, Steven M. LaValle, Devin J. Balkcom, and Matthew T. Mason. "Minimum Wheel-Rotation Paths for Differential-Drive Mobile Robots". In: *I. J. Robotics Res.* 28.1 (2009), pp. 66–80.

[11] M Chyba and T Haberkorn. "Designing efficient trajectories for underwater vehicles using geometric control theory". In: *24rd International Conference on Offshore Mechanics and Artic Engineering*. 2005.

[12] Guy Desaulniers. "On shortest paths for a car-like robot maneuvering around obstacles". In: *Robotics and Autonomous Systems* 17.3 (1996), pp. 139–148.

[13] Irina S. Dolinskaya and Alvaro Maggiar. "Time-optimal trajectories with bounded curvature in anisotropic media". In: *I. J. Robotics Res.* 31.14 (2012), pp. 1761–1793.

[14] L. E. Dubins. "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents". In: *American Journal of Mathematics* 79.3 (July 1957), pp. 497+.

[15] Andrei Furtuna. "Minimum time kinematic trajectories for self-propelled rigid bodies in the unobstructed plane". PhD thesis. Dartmouth College, 2011.

[16] Andrei A. Furtuna and Devin Balkcom. “Generalizing Dubins curves: minimum-time sequences of body-fixed rotations and translations in the plane”. In: *International Journal of Robotics Research* 29.6 (2010), pp. 703–726.

[17] Paolo Robuffo Giordano and Marilena Vendittelli. “Shortest Paths to Obstacles for a Polygonal Dubins Car”. In: *IEEE Trans. Robotics* 25.5 (2009), pp. 1184–1191.

[18] Sertac Karaman and Emilio Frazzoli. “Sampling-based Algorithms for Optimal Motion Planning”. In: *Int. J. Rob. Res.* 30.7 (June 2011), pp. 846–894.

[19] Sertac Karaman and Emilio Frazzoli. “Sampling-based algorithms for optimal motion planning”. In: *I. J. Robotics Res.* 30.7 (2011), pp. 846–894.

[20] Steven M. LaValle and James J. Kuffner Jr. “Randomized Kinodynamic Planning”. In: *I. J. Robotics Res.* 20.5 (2001), pp. 378–400.

[21] Yu-Han Lyu and Devin Balkcom. “Optimal trajectories for planar rigid bodies with switching costs”. In: *International Journal of Robotics Research* (2015).

[22] Yu-Han Lyu, Andrei A. Furtuna, Weifu Wang, and Devin Balkcom. “The bench mover’s problem: minimum-time trajectories, with cost for switching between controls”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 106–112.

[23] J. A. Reeds and L. A. Shepp. “Optimal paths for a car that goes both forwards and backwards”. In: *PACIFIC JOURNAL OF MATHEMATICS* (1990).

[24] P. Soueres and J. P. Laumond. “Shortest paths synthesis for a car-like robot”. In: *IEEE Transactions on Automatic Control* 41.5 (May 1996), pp. 672–688.

[25] Philippe Souères and J-D Boissonnat. “Optimal trajectories for nonholonomic mobile robots”. In: *Robot motion planning and control*. Springer, 1998, pp. 93–170.

[26] Héctor J Sussmann. “The Markov-Dubins problem with angular acceleration control”. In: *Decision and Control, 1997., Proceedings of the 36th IEEE Conference on*. Vol. 3. IEEE. 1997, pp. 2639–2643.

[27] Héctor J. Sussmann and Guoqing Tang. *Shortest Paths For The Reeds-Shepp Car: A Worked Out Example Of The Use Of Geometric Techniques In Nonlinear Optimal Control.* Tech. rep. 1991.

[28] Marilena Vendittelli, Jean-Paul Laumond, and Carole Nissoux. “Obstacle distance for car-like robots”. In: *IEEE Trans. Robotics and Automation* 15.4 (1999), pp. 678–691.

[29] Weifu Wang and Devin Balkcom. “Analytical time-optimal trajectories for an omni-directional vehicle”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2012, pp. 4519–4524.

[30] Weifu Wang and Devin Balkcom. “Sampling extremal trajectories for planar rigid bodies”. In: *Algorithmic Foundations of Robotics (WAFR)*. 2012, pp. 331–347.