The Geometry of Self-Driving Cars

Susan Tarabulsi, Maria-Pia Younger, Dr. Anton Lukyanenko

Mason Experimental Geometry Lab

George Mason University

December 6, 2019

Introduction

- As Self-Driving cars are becoming more popular, there is a need to minimize the length or time in order to minimize cost. For this, we must first understand the geometry of Self-Driving cars
- Since Self-Driving cars have a turning radius, we cannot use Euclidean distance to measure path-length. This problem resembles what Lexter Eli Dubins described in his 1957's research paper, *On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents*, a paper in which Dubins explained the possible optimal paths between two points given a curvature constraint.
- In our project we aimed to study the geometry of Dubins paths. Using Matlab's and The Open Motion Planning Library (OMPL)'s DubinsStateSpace packages we were able to:
  - Analyze Dubin's paths and validate Dubin's 1957 research paper based on these observations
  - Calculate lengths and observe length changes as the initial and final directions change when having fixed initial and final position points:
    $$f : \{(t_1, t_2) : t_i \in \{0, \tfrac{\pi}{n}, \tfrac{2\pi}{n}, \tfrac{2n\pi}{n}\}\} \rightarrow \text{Shortest Path}$$
  - Categorize Dubin's paths, where:
    $f : \{\mathbb{R}x\mathbb{R}x[0, 2\pi]\} \rightarrow S = \{LRL, LSL, LSR, RLR, RSR, RSL\}$ where $\{L, R, S, LS, LR, RL, RS\} \in S$
  - Created a dynamic tool to see how these short paths behave given a minimum turning radius, and final position and direction points.

Dubins Model Overview

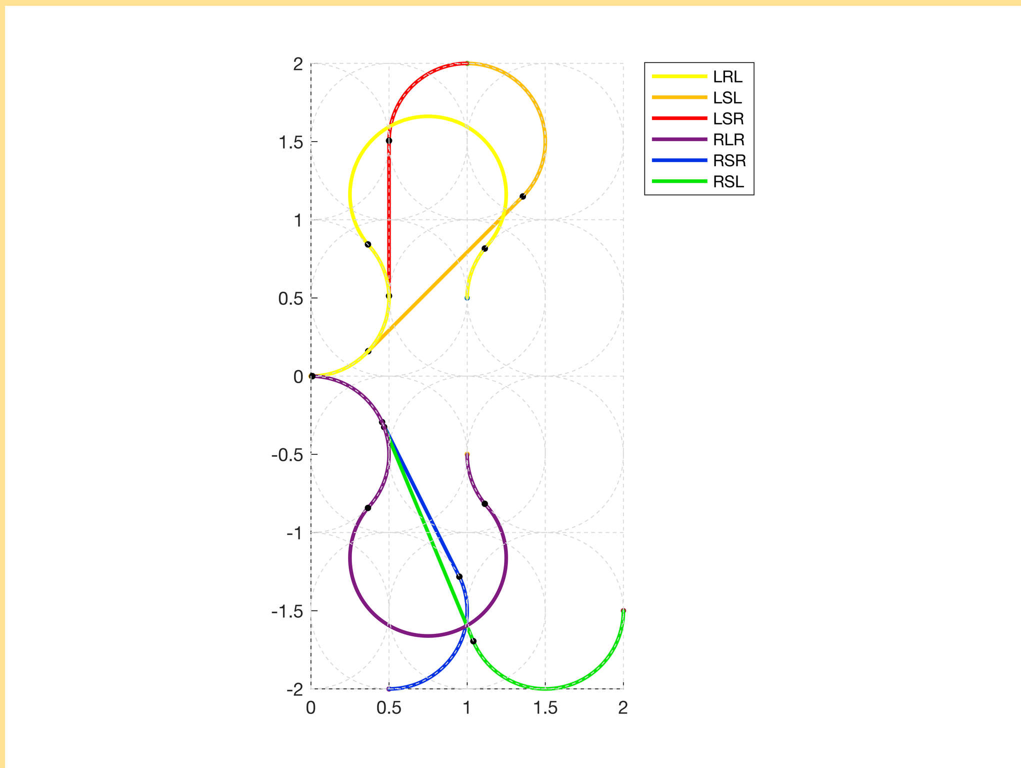
- The Dubins model is commonly used in the fields of robotics and control theory as a way to plan paths for wheeled robots, airplanes and underwater vehicles, and usual driving especially at high speed.
- An object can directly move forward only. To let objects move backwards, we would have to apply Reed-Shepp's algorithm which we will not discuss here.
- It can get anywhere if there are no obstacles; otherwise, it can get stuck.
- It cannot wiggle to change the angle.

The most effective way to get to the destination is to combine straight and full-turn motions. It is proven by L.E. Dubins that his model is effective and works in any space.

Graphics

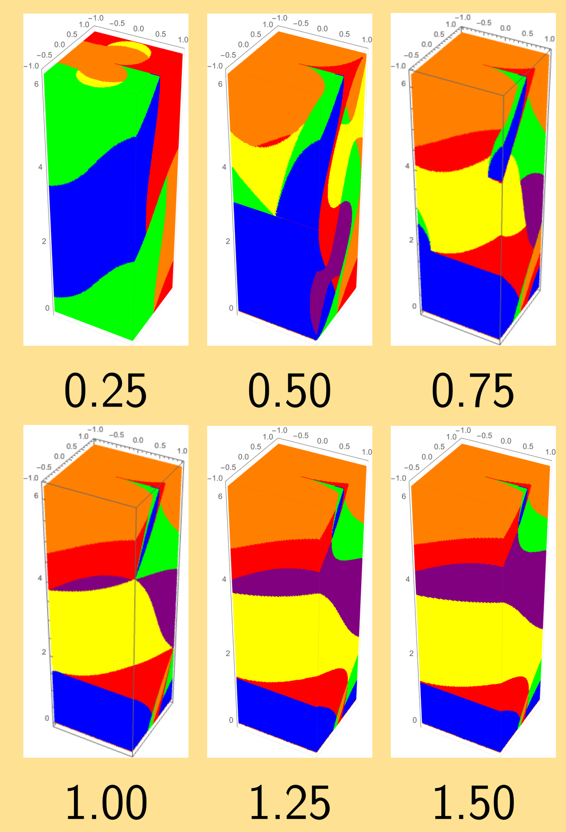
Dubins Paths Includes CSC & CCC Trajectories Only

Dubins paths consist of CCC and CSC configuration trajectories. The CSC trajectories include LSL, LSR, RSL, and RSR — a turn followed by a straight line followed by another turn (i.e., Left (L) or Right (R)). The CCC trajectories include LRL and RLR - a turned followed by a turn of the opposite direction. Each curve travels around the given radius constraint circle. See below:



Categorization & Visualization of Dubins Paths

To begin studying Dubins paths we categorized each path given a Minimum Turning Radius (MTR) constraint. Notice that because the position space is fixed between $[(-1, 1), (-1, 1), (0, 2\pi)]$ and for this experiment, the MTR changes between 0.25 to 1.50, we can see slices of the Dubins space from different ratio specs.



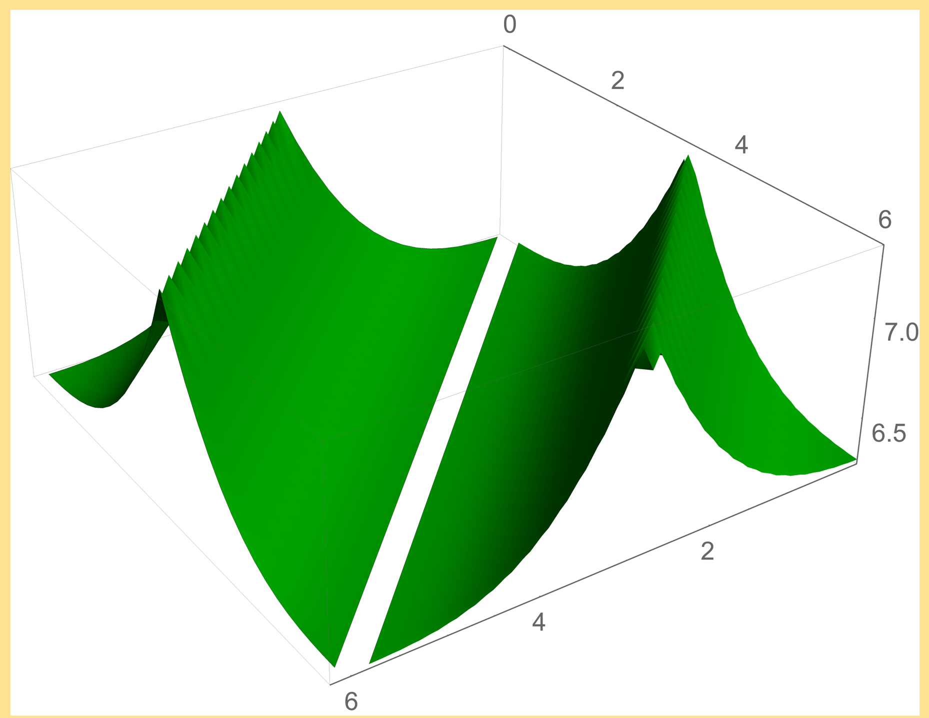
Dubins Length: Sampling of The $(0, 0)$ to $(0, 0)$ Slice

Next is an example of the visualization of Dubins lengths. For this example we used fixed initial and final positions (x_1, y_1), (x_2, y_2) at $(0, 0)$, $(0, 0)$ respectively. The coordinates of this graph represents $(\theta_1, \theta_2, length)$, where θ_1 is the initial direction, θ_2 is the ending direction, length is the outcome of the function f defined by:

$$f(x_1, y_1, \theta_1, x_2, y_2, \theta_2) = f_{x,y}(\theta_1, \theta_2)$$
$$\text{f:}\{(\theta_1, \theta_2) : \theta_i \in \{0, \tfrac{\pi}{n}, \tfrac{2\pi}{n}, \tfrac{2n\pi}{n}\}\} \rightarrow \text{length} = \text{Dubins Shortest Path}$$

This graph has a natural discontinuity when $\theta_1 = \theta_2$. This is because the position and direction do not change at these points, hence objects do not need to move. Note however, when results do not lie in the $\frac{\pi}{4}$ or 45 degree line, all points are continuous. Notice that both, "upper" and "lower", regions in reference to the 45 degree line are continuous but may not necessarily be differentiable.

Dubins Length Graph of The $(0, 0)$ to $(0, 0)$ Universe



Code Behind Dubins Algorithms

Below is a brief description of the algorithms derived from Dubins observations. We used Matlab's DubinsStateSpace package to retrieve path data for each (x_1, y_1, θ_1) to (x_2, y_2, θ_2), looped through a given space, categorized path types with the Path Algorithm; created images, videos, and other tools for analysis purposes. The Arc Length Algorithm was retrieved from other documentation sources. The length computation is a built-in function under the DubinsStateSpace package.

Path Algorithm

```
Let d = ""
For i = 2 to Rows From Interpolated Data
 θ_i = atan2(y_i − y_{i−1}, x_i − x_{i−1})
  If θ_{i−1} − θ_i = 0 Then
   d = "straight"
  Else If theta_i > theta_{i−1} Then
   d = "right"
  Else
   d = "left"
  End If
  If (θ_i − θ_{i−1} > π)
   If d = "right" Then
    d = "left"
   Else If d = "left" Then
    d = "right"
   End If
  End If
 d = d + d
 End For
Return d
```

Arc Length Algorithm

```
θ = atan2(y_i − y_{i−1}, x_i − x_{i−1})
If θ =< 0 And d ="left" Then
 θ = θ + 2π
else If θ > 0 And d ="right" then
 θ = θ − 2π
End If
Return (θ ∗ r)
```

Conclusion & Suggested Ongoing Work

We concluded that Dubins space is discontinuous. This was expected because Dubins space is asymmetrical. We also concluded that Dubins paper is correct based on experimental observations. As future work, it is suggested to analyze the Dubins slices (e.g., the $(0, 0, \theta_1)$ to $(0, 0, \theta_2)$ slice), also to analyze the 3D blocks and improve any tools created to understand Dubins space better.

Acknowledgments

We thank the Mason Experimental Geometry Lab founding director and co-director, Dr. Sean Lawton and Dr. Anton Lukyanenko, for the opportunity to conduct undergraduate mathematics research. A special thanks to Dr. Anton Lukyanenko - once again - for his guidance with this project, and the MEGL team for making this a great experience. Special thanks to Bram Bekker, Julian Benali, Savannah Crawford, and Dr. Jack Love.

References

1. Dubins, L. E. (1957). 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. doi: 10.2307/2372560
2. Gieseanw, A. (2013, July 4). A Comprehensive, Step-by-Step Tutorial to Computing Dubin's Paths. Retrieved from https://gieseanw.wordpress.com/2012/10/21/a-comprehensive-step-by-step-tutorial-to-computing-dubins-paths/.
3. Șucan, I. A., Moll, M., Kavraki, L. E. (n.d.). The Open Motion Planning Library. Retrieved from https://ompl.kavrakilab.org/.
4. stateSpaceDubins. (n.d.). Retrieved from https://www.mathworks.com/help/nav/ref/statespacedubins.html.
5. Reeds, J., Shepp, L. (1990). Optimal paths for a car that goes both forwards and backwards. Pacific Journal of Mathematics, 145(2), 367–393. doi: 10.2140/pjm.1990.145.367