ME 570: Robot Motion Planning

Homework 2 Report

By Cameron Cipriano

09/30/2021

**Problem 1: Rotations**

Question 1.1 \_report\_ – Geometric Reasoning of the Rotation Matrices

This rotation matrix rotates the and -axes counterclockwise around the -axis

A picture containing chart

Description automatically generated

This rotation matrix rotates the and -axes counterclockwise around the -axis

Chart

Description automatically generated with low confidence

This rotation matrix rotates the and -axes counterclockwise around the -axis

Chart

Description automatically generated

This rotation matrix reflects the and -axes across the -axis and then rotates them counterclockwise.

Diagram

Description automatically generated

**Problem 2: Free Configuration Space for a Two-Link Manipulator**

Question 2.1 \_report\_ – Coordinate Transformations

1. The coordinates of the point given in , denoted as :  
     
    where and is the simple rotation matrix
2. The coordinates of the point given in denoted as :  
     
   : To compute this, we need to perform   
     
   : Plugging this into the above, converting form   
     
      
    **where** **and**   
     
   Also important is is, again, the simple rotation matrix:  
      
     
   We now have:

Question 2.1 \_code\_ – TwoLink.kinematic\_map()

Code based on the previous report question worked very well

Question 2.1 \_optional\_ – TwoLink.kinematic\_map(i)

Question 2.2 \_code\_ – TwoLink.is\_collision()

I had a little bit of trouble with this method because the output was consistently incorrect. I decided to dig deeper into the python solution code that I am using, and I believe the Polygon.is\_collision() method is returning the incorrect value. The method is using logical OR on flag\_points based on if the point is\_visible(). However, this logical OR is determining a flag\_point to say there is a collision. The return of is\_visible() is thus being used to indicate a collision instead of a non-collision, so I had to logically reverse the returned array to really determine if there was a collision.

Question 2.2 \_report\_ – TwoLink.plot\_collision()

Chart, line chart

Description automatically generated

Question 2.2 \_optional\_ – TwoLink.free\_space()

**Problem 4: Charts for the Circle Using Rotations**

Question 4.1 \_report\_ – Show is a rotation and that

To show that is a rotation , we have to show that and

This demonstrates that is a rotation

Given , having a map that is , which by definition, is the coordinate space of the unit circle

Question 4.1 \_optional\_ – Two Charts for a Circle

We need two charts for a circle because, by definition, a chart maps ***open regions*** in one space to ***open regions*** in another. Because it is an open region, there must be at least two regions such that their union covers the entire area.

Question 4.2 \_optional\_ – Generate Vectors and

**Problem 5: Charts for the Torus Using Rotations**

Question 5.1 \_code\_ – Torus.phi()

Question 5.1 \_report\_ – Create Atlas for Torus

The torus that we are interested in is created via the product of 2 circles, . Because each circle needs a minimum of 2 charts to cover, we thus determine that there needs to be a minimum of 4 charts to cover (using rectangular charts).

The charts that I will use need to cover the entire space with a little overlap so that the open-set overlaps.

Question 5.2 \_report\_ – Torus.plot\_charts()

Question 5.2 \_optional\_ – Plot All Charts Simultaneously

Question 5.3 \_report\_ – Explain the Atlas for the Torus

Each chart should not overlap itself, and no part should be left uncovered due to the definition of a chart. Charts are homeomorphisms that map open regions in to the topological space. Because these regions are open, they cannot overlap themselves

Question 5.4 \_report\_ – Compute the Tangent   of the curve

Tangent of this curve:

Question 5.2 \_code\_ – Torus.phi\_push\_curve()

Question 5.5 \_report\_ – Torus.plot\_charts\_curves

**Problem 6: Jacobians and End Effector Velocities**

Question 6.1 \_report\_ – Expression for as a function of

Text, letter

Description automatically generated

Question 6.1 \_optional\_ – Find the Jacobian from Previous Question

Diagram

Description automatically generated

Question 6.1 \_code\_ – TwoLink.jacobian()

Question 6.2 \_report\_ – Find for Set of Values

Question 6.3 \_report\_ – me570\_hw2.torus\_twolink\_plot\_jacobian()

Question 6.4 \_report\_ – Relationship between \_report 5.5\_ and \_report 6.3\_