Programming report for problems 1 and 2, by Carl Stott

Problem 1 part A)

this function takes a 3x3 rotation matrix R contained in SO(3) and returns

the axis angle representation of that R such that exp((w)^(theta))=R.

Where w is the unit axis of rotation (||w||=1) and theta is the angle of

rotation (theta will be between 0 and pi).

The code was written using the matrix logarithm of a rotation matrix, defined in slide 11 of week 2 lecture 2

The steps takin in my program are described below:

1. check to make sure that R is contained by SO(3).

to do that, make sure dimensions of R is 3x3 check that det(R)=1, and

that Rtranspose(R)=I. the use of the any() function reduces a matrix

or vector of logical values into a single logical, which is what the

or operator needs.

2. if R is an ID matrix, return theta=0 and w as undefined.

3. if trace R is -1, than theta is pi and we are in a singularity case.

4. if the final else condition is reached, R is in so(3), and trace(R)/=-1. So my axis and angle can be defined using equations taken from week 2 lecture 2 slideshow

Problem 1 part B)

%this function takes a 3x3 rotation matrix R contained in SO(3) and returns

%the unit quaternion.

%this function was written by Carl Stott on 2/5/2023.

% NOTE YOU NEED TO BE ABLE TO CALL THE sgn() FUNCTION FOR THIS TO WORK. The

% sgn() function is the same as matlab's sign() function but where sign()

% returns a 0, sgn() returns a 1. The sgn() function is included in HW1’s code package. Steps this code takes are described below

% 1. check to make sure that R is contained by SO(3).

% to do that, make sure dimensions of R is 3x3 check that det(R)=1, and

% that Rtranspose(R)=I. the use of the any() function reduces a matrix

% or vector of logical values into a single logical, which is what the

% or operator needs.

2) Once R is confirmed to be contained in SO(3), elements of the quartornian are calculated using math described on W3L1 slide 14.

Problem 1 part c)

this function takes a 3x3 rotation matrix R contained in SO(3) and returns

the ZYZ and roll pitch yaw (ZXY) representations. the output RPY refers to

the ZYZ representation. The steps this function takes are described below.

this function was written by Carl Stott on 2/5/2023.

1. check to make sure that R is contained by SO(3).

to do that, make sure dimensions of R is 3x3 check that det(R)=1, and

that Rtranspose(R)=I. the use of the any() function reduces a matrix

or vector of logical values into a single logical, which is what the

Boolean operators need.

2) %now that I'm sure we are dealing with a matrix in SO(3), I can define theta, phi, and psi for my ZYZ orientation. I choose theta between (0,pi), and use math described in week 3 lecture 1 slide 8. If sin(theta)=0, I am in a singularity and the program returns as such. After line 34 I have created the ZYZ orientation, so I am free to redefine my variables.

3) now I will redefine my phi, theta, and psi for my ZXY orientation W3L1 slide 9. Using -(3,1) to find theta because (3,1)=-sin(theta). I have a line that checks to see if theta is between -pi/2 and pi/2, theta is found by examining R(3,1) as shown in slide 10 of the w3L1 slideshow.

The elseif() statement checks to see if theta is between pi/2 and (3\*pi)/2, theta is found by examining R(3,1) as shown in slide 10 of the week 3 lecture 1 slideshow.

The final else() statement is only entered when theta is +/- pi/2, which presents a singularity for a ZYX rotation.

Problem 2 part A:

this function takes an axis and an angle representing a rotation and

return the rotation matrix representing that rotation.

this function was written by Carl Stott on 2/20/2022

step 1) My first if() statement checks to make sure that my user isn’t feeding me theta=0 and

checking to make sure w is a valid axis. If they aren’t, my code will tell you.

Step 2) now that I know my user isn’t feeding me garbage, I can compute R. source: Modern Robotics and Control (Sprong, Hutchinson, Vidyasagar)pg 58, eq 2.43. I break my w vector into its components so creating the final rotation matrix is easier to read in the code.

Problem 2 part B:

this function takes a quaternion and returns a rotation matrix

this function was written by Carl Stott on 2/20/2022

first, I check to make sure my user is feeding me a 4x1, if they are feeding me a crazy 4x1 that isn’t useful and claiming it’s a quaternion, that’s on them.

Second, I turn my quaternion into a unit quaternion, and break the 4x1 up into its components so the code is easier to read. Then my rotation matrix is created (called R) using the algorithm described in the lecture.