**COMP127 Coursework1**

**Q1(a)**

For an arbitrary 3D rotation matrix, we have

Then for the first row we have

Then, we can find

Therefore

**Q1(b)**

According to Rodrigues’ Formula, we have rotation matrix:

Then, we have

**Q1(c)**As we have one vector P which has coordinates of frame a and coordinates of ‘frame b’ .

Each row of represents the projection of axes in ‘frame b’ to axes in ‘frame a’.

In other words, assume

The first row means the vector of ‘frame b’, which has x-axis a, y-axis b, c-axis c, is projected to x-axis in ‘frame a’, and the number is unitised.

**Q1(d)**

As we have an axis/angle rotation matrix of rotating around Z axis.

To find its eigenvalues

Solve it, we have

When ,

When ,

when ,

As for eigenvectors, substituting this ever-present eigenvalue into**,** we have:

Then, there are infinite solutions, and we choose one of them

Therefore, the eigenvector for eigenvalue is .

**Q2(a)**

For an extrinsic proper Euler rotation example

Where

Example of intrinsic x-y-z Tait-Bryan rotation

Where

Why should we avoid?

Because, if gimbal lock happens, joints or robots will lose some of their dimensions, which mean they will not rotate in certain orientations. When it happens to an android’s knee, the robot may fall and causes consequences.

**Q2(b)**

For a rotation with unit vector and angle , the equivalent quaternion is given by:

Therefore, we can decompose q with

All we need to do is using the four equations above to represent Axis-angle representation.

For

Because , hence

Since have the same pattern, we can similarly deduce that

For

Similarly, the other items which share the same pattern can be listed below:

Therefore, the result is:

**Q2(c)**

What rotation representation would you suggest using in the following cases:

• Nano-robot with very limited memory storage

Axis-angle representation, only three parameters are required

• Nano-robot with very limited computational power

Rotational Matrix, it doesn’t need extra calculation except one

• iPhone navigation system

Euler Angles, matches human intuition

• Robotic arm with 6 DOF

Quaternions

**Q3(a)**

To achieve it, convert quaternion to rotation matrix

And thecorrespond is

=**q**

and **.**has the same rotation matrix. Therefore, is equivalent to **.**

**Q3(b)**

1. When one of and is identity matrix.
2. When both and rotating around the same axes.
3. When both are 2-D rotation
4. When and are inverse to each other, which means

**Q4(b)**

Text

Description automatically generated

Figure 1: Quaternions to Euler Angle [1]

To convert a quaternion representation to a Euler angle Z-Y-X representation (Tait-Bryan, extrinsic), we must calculate each angle from quaternion representation. As is shown in Figure 1, the relationship between quaternions is listed.

**Q4(c)**

To obtain the Rodrigues representation, which has response generated by . The first step is to convert quaternion expression to rotation matrix .

Following the relationship:

Then, using the rotation matrix to find rotation angle and the normal unit vector:

The response can be expressed as

**Q5(a)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table 1:D-H parameters for 5a | | | | |
| i | (rad) | (m) | (rad) | (m) |
| 1 | 0 | 0 | 0 | 0.072 |
| 2 |  | 0 | 0 | 0.075 |
| 3 | 0 | 0.155 |  | 0 |
| 4 | 0 | 0.135 | 0 | 0 |
| 5 |  | 0 |  | 0 |

Diagram

Description automatically generated

Figure2: D-H frames for Q5a

Firstly, all the axis should be chosen along their rotation axis through corresponding joints. Then, the frame 0 is defined at the bottom following the right-hand rule. Thirdly, is determined by the intersection between and, and for all other origins, we locate origins at the intersections of their joints and themselves. After that, each is defined along the common normal of and . All the y-axes are determined by right-hand rule as x and z axis have been set.

Applying some analysis on joint1:

It is obvious that the distance between and along is 0.072 m, and the distance between andalongzero. Both angles of x axis and z axis are zero as the direction of the frame does not change.

Similar analysis can be applied to all other joints and the result is shown in Table 1.

**Q5(b)**

To obtain standard\_dh, we are going to use two equations:

Where are D-H parameters for each joint.

The chain rule applies to D-H transform.

As for ‘’ fkine wrapper( )” function, it is a callback function of main( ). The code below shows that there is a subscriber who subscribes to /joint\_states topics. When a message comes into the subscriber, it will active callback function “fkine wrapper( )” and pass message to it with variable “JointState”.



Figure 3:

After that, the callback function will firstly initiate some container to hold input and output information. Then it will split data and send them to forward\_kinematics function, which will generate DH transformer. In the end, the processed data will be sent to /tf topic. In addition, the input messages are passed to forward\_kinematics and eventually affect the generation of DH transformer.

**Q5(c)**

This question asks students to generate D-H parameters with the data in urdf file. As the position and orientation in urdf is relative and has a regular for urdf to point its Z axis to the next joint. Therefore, we can apply the D-H convention to generate parameters easily. The result is shown below in figure 4

Calendar

Description automatically generated

Figure 4: D-H parameters generated from urdf

Most of the data is extracted from urdf, because we know the position and orientation of each joint and the convention of urdf format. They can be calculated easily. However, there are two angles (Two 77.5 degrees) are hard to extract from urdf. To tackle this problem, the relative orientation of joint 3 and joint 5 in RViz are collected to calculate the exact angle for joint 3 and 5. What’s more, the offsets are the parameters that can calibrate initial D-H position to right place. Therefore, they are the relative orientation in reference to previous joint.

**Reference**

[1] Blanco, Jose-Luis (2010). "A tutorial on se (3) transformation parameterizations and on-manifold optimization". University of Malaga, Tech. Rep. [CiteSeerX](https://en.wikipedia.org/wiki/CiteSeerX_(identifier)" \o "CiteSeerX (identifier)) [10.1.1.468.5407](https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.468.5407).