COMP127 Coursework1

Q1(a)

For an arbitrary 3D rotation matrix, we have

$$\boldsymbol{RR^T} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \begin{bmatrix} r_1 & r_4 & r_7 \\ r_2 & r_5 & r_8 \\ r_3 & r_6 & r_9 \end{bmatrix} = \boldsymbol{I}$$

Then for the first row we have

$$\begin{cases} r_1^2 + r_2^2 + r_3^2 = 1 \\ r_4^2 + r_5^2 + r_6^2 = 1 \\ r_7^2 + r_8^2 + r_9^2 = 1 \end{cases}$$

Then, we can find

$$r_1^2, r_2^2, r_3^2, r_4^2, r_5^2, r_6^2, r_7^2, r_8^2, r_9^2 \leq 1$$

Therefore

$$|r_i| \leq 1$$

Q1(b)

According to Rodrigues' Formula, we have rotation matrix:

$$\boldsymbol{R}_{(\boldsymbol{k},\boldsymbol{\theta})} = \begin{bmatrix} cos\theta + k_x^2(1 - cos\theta) & k_x k_y(1 - cos\theta) - k_z sin\theta & k_x k_z(1 - cos\theta) + k_y sin\theta \\ k_y k_x(1 - cos\theta) + k_z sin\theta & cos\theta + k_y^2(1 - cos\theta) & k_y k_z(1 - cos\theta) - k_x sin\theta \\ k_z k_x(1 - cos\theta) - k_y sin\theta & k_z k_y(1 - cos\theta) + k_x sin\theta & cos\theta + k_z^2(1 - cos\theta) \end{bmatrix}$$

Then, we have

$$\boldsymbol{R}_{(-\boldsymbol{k},-\boldsymbol{\theta})} = \begin{bmatrix} \cos(-\theta) + k_x^2(1 - \cos(1 - \theta)) & k_x k_y(1 - \cos(1 - \theta)) - \mu_z \sin(-\theta) & k_x k_z(1 - \cos(1 - \theta)) + k_y \sin(-\theta) \\ k_y k_x(1 - \cos(1 - \theta)) + k_z \sin(-\theta) & \cos(-\theta) + k_y^2(1 - \cos(1 - \theta)) & k_y k_z(1 - \cos(1 - \theta)) - k_x \sin(-\theta) \\ k_z k_x(1 - \cos(1 - \theta)) - k_y \sin(-\theta) & k_z k_y(1 - \cos(1 - \theta)) + k_x \sin(-\theta) & \cos(-\theta) + k_z^2(1 - \cos(1 - \theta)) \end{bmatrix}$$

$$= \begin{bmatrix} cos\theta + k_x^2(1 - cos\theta) & k_x k_y(1 - cos\theta) - \mu_z sin\theta & k_x k_z(1 - cos\theta) + k_y sin\theta \\ k_y k_x(1 - cos\theta) + k_z sin\theta & cos\theta + k_y^2(1 - cos\theta) & k_y k_z(1 - cos\theta) - k_x sin\theta \\ k_z k_x(1 - cos\theta) - k_y sin\theta & k_z k_y(1 - cos\theta) + k_x sin\theta & cos\theta + k_z^2(1 - cos\theta) \end{bmatrix}$$

$$= \boldsymbol{R}_{(\boldsymbol{k},\boldsymbol{\theta})}$$

Q1(c)

As we have one vector P which has coordinates of frame a$[x_a, y_a, z_a]$ and coordinates of 'frame b' $[x_b, y_b, z_b]$.

Each row of $a\boldsymbol{R}_b$ represents the projection of axes in 'frame b' to axes in 'frame a'.

In other words, assume

$$a\boldsymbol{R}_b = \begin{bmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{bmatrix}$$

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.

$$R_Z(\alpha) = \begin{bmatrix} cos\alpha & -sin\alpha & 0 \\ sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

To find its eigenvalues

$$\begin{vmatrix} cos\alpha - \lambda & -sin\alpha & 0 \\ sin\alpha & cos\alpha - \lambda & 0 \\ 0 & 0 & 1 - \lambda \end{vmatrix} = 0$$

$$(1 - \lambda)[(cos\alpha - \lambda)^2 + sin^2\alpha] = 0$$

Solve it, we have

$$\lambda_1 = 1, \lambda_2 = cos\alpha + i * sin\alpha, \lambda_3 = cos\alpha - i * sins\alpha$$

When $\alpha = 0, \lambda_1 = \lambda_2 = \lambda_3 = 1$

When $\alpha = \pi, \lambda_1 = 1, \lambda_2 = \lambda_3 = -1$

when $0 < \alpha < \pi, \lambda_1 = 1, \lambda_2 = cos\alpha + i * sin\alpha, \lambda_3 = cos\alpha - i * sins\alpha$

As for eigenvectors, substituting this ever-present eigenvalue into$(R_Z(\alpha) - \lambda I)_1 V = \mathbf{0},$ we have:

$$(R - \lambda I)v_i = \begin{bmatrix} cos\alpha - 1 & -sin\alpha & 0 \\ sin\alpha & cos\alpha - 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$(cos\alpha - 1)v_1 - sin\alpha v_2 = 0$$

$$(cos\alpha - 1)v_2 - sin\alpha v_1 = 0$$

Then, there are infinite solutions, and we choose one of them $v_1 = v_2 = 0, v_3 = 1$

Therefore, the eigenvector for eigenvalue $\lambda_1 = 1$ is $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$.

Q2(a)

For an extrinsic proper Euler rotation $\boldsymbol{Y-Z-Y}$ example

$$R_y(\gamma)R_z(0)R_y(\alpha) = \begin{bmatrix} cos\gamma & -sin\gamma & 0 \\ sin\gamma & cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\alpha & -sin\alpha & 0 \\ sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Where $\theta = \alpha + \gamma$

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

$$R_y(\gamma)R_z(0)R_y(\alpha) == \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix}$$

Where $\theta = \alpha + \gamma$

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 μ and angle θ, the equivalent quaternion is given by:

$$q = e^{\frac{\theta}{2}(u_x i + u_y j + u_z k)} = cos\frac{\theta}{2} + (u_x i + u_y j + u_z k)sin\frac{\theta}{2}$$

Therefore, we can decompose q with

$$\begin{cases} q_w = cos\frac{\theta}{2} \\ q_x = u_x sin\frac{\theta}{2} \\ q_y = u_y sin\frac{\theta}{2} \\ q_z = u_z sin\frac{\theta}{2} \end{cases}$$

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

$\boldsymbol{R}_{(\boldsymbol{u},\boldsymbol{\theta})}$

$$= \begin{bmatrix} cos\theta + u_x^2(1-cos\theta) & u_x u_y(1-cos\theta) - \mu_z sin\theta & u_x u_z(1-cos\theta) + u_y sin\theta \\ u_y u_x(1-cos\theta) + u_z sin\theta & cos\theta + u_y^2(1-cos\theta) & u_y u_z(1-cos\theta) - u_x sin\theta \\ u_z u_x(1-cos\theta) - u_y sin\theta & u_z u_y(1-cos\theta) + u_x sin\theta & cos\theta + u_z^2(1-cos\theta) \end{bmatrix}$$

For r_1, r_5, r_9,

$$\begin{aligned} r_1 &= cos\theta + u_x^2(1-cos\theta) \\ &= 1 - 1 + u_x^2 - u_x^2 cos\theta + cos\theta \\ &= 1 - (1 - u_x^2 + u_x^2 cos\theta - cos\theta) \\ &= 1 - (1-cos\theta)(1-u_x^2) \end{aligned}$$

$$
\begin{aligned}
&= 1 - \left(1 - 1 + 2sin^2\left(\frac{\theta}{2}\right)\right)(1 - u_x^2) \\
&= 1 - 2sin^2\left(\frac{\theta}{2}\right) + 2u_x^2 sin^2\left(\frac{\theta}{2}\right) \\
&= 1 - 2 + 2cos^2\left(\frac{\theta}{2}\right) + 2u_x^2 sin^2\left(\frac{\theta}{2}\right) \\
&= -1 + 2q_w^2 + 2q_x^2
\end{aligned}
$$

Because $q_w^2 + q_x^2 + q_y^2 + q_z^2 = 1$, hence

$$r_1 = 1 - 2q_y^2 - 2q_z^2$$

Since r_1, r_5, r_9 have the same pattern, we can similarly deduce that

$$r_5 = 1 - 2q_x^2 - 2q_z^2$$
$$r_9 = 1 - 2q_x^2 - 2q_y^2$$

For $r_2, r_3, r_4, r_6, r_7, r_8$

$$
\begin{aligned}
r_2 &= u_x u_y (1 - cos\theta) - \mu_z sin\theta \\
&= u_x u_y \left(1 - 1 + 2sin^2\left(\frac{\theta}{2}\right)\right) - 2\mu_z \sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\theta}{2}\right) \\
&= 2u_x \sin\left(\frac{\theta}{2}\right) u_y \sin\left(\frac{\theta}{2}\right) - 2\mu_z \sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\theta}{2}\right) \\
&= 2q_x q_y - 2q_w q_z
\end{aligned}
$$

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

$$
\begin{aligned}
r_3 &= 2q_x q_z + 2q_y q_w \\
r_4 &= 2q_x q_y + 2q_z q_w \\
r_6 &= 2q_y q_z - 2q_x q_w \\
r_7 &= 2q_x q_z - 2q_y q_w \\
r_8 &= 2q_y q_z + 2q_x q_w
\end{aligned}
$$

Therefore, the result is:

$$
\boldsymbol{R} = \begin{bmatrix} 1 - 2q_y^2 - 2q_z^2 & 2q_x q_y - 2q_z q_w & 2q_x q_z + 2q_y q_w \\ 2q_x q_y + 2q_z q_w & 1 - 2q_x^2 - 2q_z^2 & 2q_y q_z - 2q_x q_w \\ 2q_x q_z - 2q_y q_w & 2q_y q_z + 2q_x q_w & 1 - 2q_x^2 - 2q_y^2 \end{bmatrix}
$$

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 $\boldsymbol{q}$ to rotation matrix

$$\boldsymbol{q} = \begin{bmatrix} 1-2q_y^2-2q_z^2 & 2q_xq_y-2q_zq_w & 2q_xq_z+2q_yq_w \\ 2q_xq_y+2q_zq_w & 1-2q_x^2-2q_z^2 & 2q_yq_z-2q_xq_w \\ 2q_xq_z-2q_yq_w & 2q_yq_z+2q_xq_w & 1-2q_x^2-2q_y^2 \end{bmatrix}$$

And the correspond $-\boldsymbol{q}$ is

$$-\boldsymbol{q}$$

$$= \begin{bmatrix} 1-2q_y^2-2q_z^2 & 2(-q_x)(-q_y)-2(-q_z)(-q_w) & 2(-q_x)(-q_z)+2(-q_y)(-q_w) \\ 2(-q_x)(-q_y)+2(-q_z)(-q_w) & 1-2q_x^2-2q_z^2 & 2\left(-q_y\right)(-q_z)-(2-q_x-q_w) \\ 2(-q_x)(-q_z)-2\left(-q_y\right)(-q_w) & 2-q_y-q_z+2(-q_x)(-q_w) & 1-2q_x^2-2q_y^2 \end{bmatrix}$$

$$= \begin{bmatrix} 1-2q_y^2-2q_z^2 & 2q_xq_y-2q_zq_w & 2q_xq_z+2q_yq_w \\ 2q_xq_y+2q_zq_w & 1-2q_x^2-2q_z^2 & 2q_yq_z-2q_xq_w \\ 2q_xq_z-2q_yq_w & 2q_yq_z+2q_xq_w & 1-2q_x^2-2q_y^2 \end{bmatrix}$$

=q

$\boldsymbol{q}$ and $-\boldsymbol{q}$.has the same rotation matrix. Therefore, $\boldsymbol{q}$ is equivalent to $-\boldsymbol{q}$.

Q3(b)

1. When one of $\boldsymbol{R}_a$ and $\boldsymbol{R}_b$ is identity matrix.
2. When both $\boldsymbol{R}_a$ and $\boldsymbol{R}_b$ rotating around the same axes.
3. When both are 2-D rotation
4. When $\boldsymbol{R}_a$ and $\boldsymbol{R}_b$are inverse to each other, which means $\boldsymbol{R}_a\boldsymbol{R}_b = \boldsymbol{R}_b\,\boldsymbol{R}_a = \boldsymbol{I}$

Q4(b)

$$\begin{bmatrix}\phi\\ \theta\\ \psi\end{bmatrix} = \begin{bmatrix} atan2(2(q_w q_x + q_y q_z), 1 - 2(q_x^2 + q_y^2)) \\ asin2(2(q_w q_y - q_z q_x)) \\ atan2(2(q_w q_z + q_x q_y), 1 - 2(q_y^2 + q_z^2)) \end{bmatrix}$$

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 $[x, y, z]$ generated by $[u_x\theta, u_y\theta, u_z\theta]$. The first step is to convert quaternion expression to rotation matrix $\boldsymbol{R}$. Following the relationship:

$$\boldsymbol{R} = \begin{bmatrix} 1 - 2q_y^2 - 2q_z^2 & 2q_x q_y - 2q_z q_w & 2q_x q_z + 2q_y q_w \\ 2q_x q_y + 2q_z q_w & 1 - 2q_x^2 - 2q_z^2 & 2q_y q_z - 2q_x q_w \\ 2q_x q_z - 2q_y q_w & 2q_y q_z + 2q_x q_w & 1 - 2q_x^2 - 2q_y^2 \end{bmatrix}$$

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

$$\theta = arcos(\frac{R_{1,1} + R_{2,2} + R_{3,3} - 1}{2})$$

$$\boldsymbol{\mu} = \frac{1}{2sin\theta}\begin{bmatrix} R_{3,2} - R_{2,3} \\ R_{1,3} - R_{3,1} \\ R_{2,1} - R_{1,2} \end{bmatrix}$$

The response can be expressed as

$$[u_x\theta, u_y\theta, u_z\theta]$$

Q5(a)

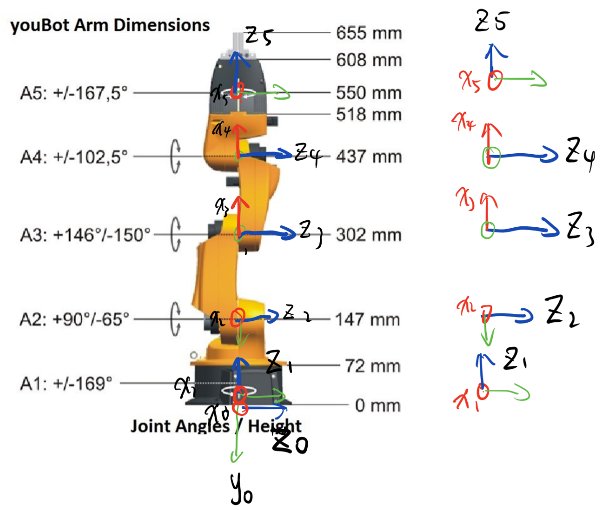


Table 1:D-H parameters for 5a				
i	α(rad)	a(m)	θ(rad)	d(m)
1	0	0	0	0.072
2	$-\frac{\pi}{2}$	0	0	0.075
3	0	0.155	$-\frac{\pi}{2}$	0
4	0	0.135	0	0
5	$\frac{\pi}{2}$	0	$\frac{\pi}{2}$	0

Figure2: D-H frames for Q5a

Firstly, all the z 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, O_1 is determined by the intersection between z_0 and z_1, and for all other origins, we locate origins at the intersections of their joints and themselves. After that, each x_i is defined along the common normal of z_i and z_{i-1}. 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 x_1 and x_0 along $z_0$is 0.072 m, and the distance between z_1 and z_0 along x_0 *is* zero. 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:

$$^{i-1}T_i = \begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i sin\alpha_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where θ_i, d, α, a are D-H parameters for each joint.

$$^0T_n = {}^0T_1\ {}^1T_2 \ldots {}^{n-1}T_n$$

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”.

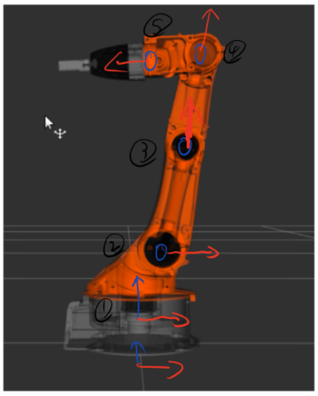
```
sub = rospy.Subscriber('/joint_states', JointState, fkine_wrapper, br)
```

Figure 3:

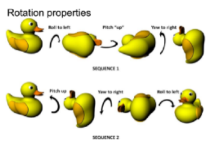
After that, the callback function will firstly initiate some container to hold input and output information. Then it will split data and send them to forward_kinematics function, which will generate DH transformer. In the end, the processed data will be sent to /tf topic. In addition, the input messages 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



	α	a	θ	d
1	0	0.024	0	0.096
2	90°	0	0	0.019
3	0	0.155	77.5°	0
4	0	0.135	0	0
5	0	0.130	77.5°	0
6	0	0	0	0.055



α a θ d

	x	y	z	R	P	Y
j1	0.024	0	0.096	0	0	170
j2	0.033	0	0.019	0	-65	0
j3	0	0	0.155	0	146	0
j4	0	0	0.135	0	-102.5	0
j5	-0.002	0	0.130	0	0	167.5

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 10.1.1.468.5407.