Fundamentels of Robotics

Part 1 - Exam - Kinematics. Innopolis University, Fall 2020

Name: Hany Hamed
Group Number: BS18-Robotics
Midterm

Contents

Fundamentels of Robotics **1**

Q1 **2**

Q2 **2**

Q3 **3**

Q4 **4**

Q5 **4**

FoR, Innopolis University, Fall 2020
Author: Hany Hamed

Q1

- Roll-Pitch-Yaw is one of the representation for rotation angles. The rotations is performed with respect with fixed joints as a pre multiplication. It is widely used in drones.
- The computation is on the following way: $RPY : R(\phi, \theta, \psi) = R_z(\psi) * R_y(\theta) * R_x(\phi)$

  Such that R_x, R_y, R_z are the normal definition for the rotation matrices around the axis.
- In order to find the RPY angles from a given rotation matrix. We are solving the inverse problem. $R(\phi, \theta, \psi) =$
  $\begin{bmatrix} c_\theta c\psi & dc & dc \\ c_\theta s_{psi} & dc & dc \\ -s_\theta & c_\theta s\phi & c_\theta c_\phi \end{bmatrix}$

  such that dc is some terms that we don't care for now to solve the inverse problem:

  if $r_{31} \neq 1$ then m $= sign(cos(\theta))$ then $\phi = atan2(m * r_{32}, m * r_{33}), \theta = atan2(-r_{31}, m * \sqrt{r_{11}^2 + r_{21}^2}, \psi = atan2(m * r_{21}, m * r_{11})$

  if $r_{31} = + - 1$ then $\theta = asin(r_{31} = \{0, + - \pi/2\}$ and $m = sign(sin(\theta))$ and $\phi - m * \psi = atan2(m * r_{12}, r_{22}$
- One of the issues as Euler angles is the Gamble lock when two axes coincide together and make one case of singularity.

Q2

I have changed the index in order to make zero index and make L6 = l3

- $l_0 = 10\ (lengthunit)$
- $l_1 = 5\ (lengthunit)$
- $l_2 = 15\ (lengthunit)$
- $l_3 = 1\ (lengthunit)$

$T = T_{base}\, R_z(q_0)\, T_z(l_0)\, T_x(-l_1)\, R_x(q_1)\, T_z(l_2)\, T_z(q_2)\, R_z(q_3)\, R_x(q_4)\, R_z(q_5)\, T_z(l_3)\, T_{tool}$

Such that in the last rotation matrices are the spherical wrist.

Q3

$T = T_{base} T_z(l_0) T_{123} T_{456} T_z(l_3) T_{tool}$

$T_{123} = R_z(q_0) T_x(-l_1) R_x(q_1) T_z(l_2) T_z(q_2)$

$T_{456} = R_z(q_3) R_x(q_4) R_z(q_5)$

and then using Pieper solutions:

For the first 3 joints, get the symbolic expressions of T_{123} as T_{456} don't affect on position we can compare with the the last column of 123 transformation:

$x = -l_1 c_0 + l_2 s_0 s_1 + q_2 s_0 s_1 = -l_1 c_0 + (l_2 + q_2) s_0 s_1$

$y = -l_1 s_0 - l_2 s_1 c_0 - q_2 s_1 c_0 = -l_1 s_0 - (l_2 + q_2) s_1 c_0$

$z = l_2 c_1 + q_2 c_1 = (l_2 + q_2) c_1$ (3)

Then:

$x + l_1 c_0 = (l_2 + q_2) s_0 s_1$ (1)

$y + l_1 s_0 = -(l_2 + q_2) s_1 c_0$ (2)

Divide (1) over (2) and simplify, we get:

$y s_0 - l_1 - x c_0 = 0$, we can solve it numerically using fsolve function in python and get q_0

And then from (2) and (3), we get:

if($cos(q_0) \neq 0$) then $q_1 = atan2(y + l_1 s_0, c_0 z)$

else: then $q_1 = atan2(y + l_1 s_0, z)$

Then, if $cos(q_1) \neq 0$ then $q_2 = z/cos(q_1) - l_2$

else: $q_2 = z - l_2$

Also, the position part, it can be solved differently (Not sure):

- Looking from the side, get the projection and continue solving it for y
- Looking from the top, and solve for y

And for the spherical wrist, we will substitute in T_123 and then get the inverse and then solve it as in the assignment (Don't have time to write the full solution but it is in the code)

$$T_{456} = [\begin{array}{ccc} -\sin(q_3)\sin(q_5)\cos(q_4) + \cos(q_3)\cos(q_5) & -\sin(q_3)\cos(q_4)\cos(q_5) - \sin(q_5)\cos(q_3) & \sin(q_3)\sin(q \\ \sin(q_3)\cos(q_5) + \sin(q_5)\cos(q_3)\cos(q_4) & -\sin(q_3)\sin(q_5) + \cos(q_3)\cos(q_4)\cos(q_5) & -\sin(q_4)\cos(\\ \sin(q_4)\sin(q_5) & \sin(q_4)\cos(q_5) & \cos(q_4) \\ 0 & 0 & 0 \end{array}$$

Q4

- We will have a node as main controller and publish to different topic. A topic for each joint, the main controller ros node will publish to these topics with sensor_msgs/JointState message type, and the internal controller for each joint will subscribe for these topics (Assuming they are running ROS too).

  and the main controller ros node, listen to commands using maybe telope or user input or anything.

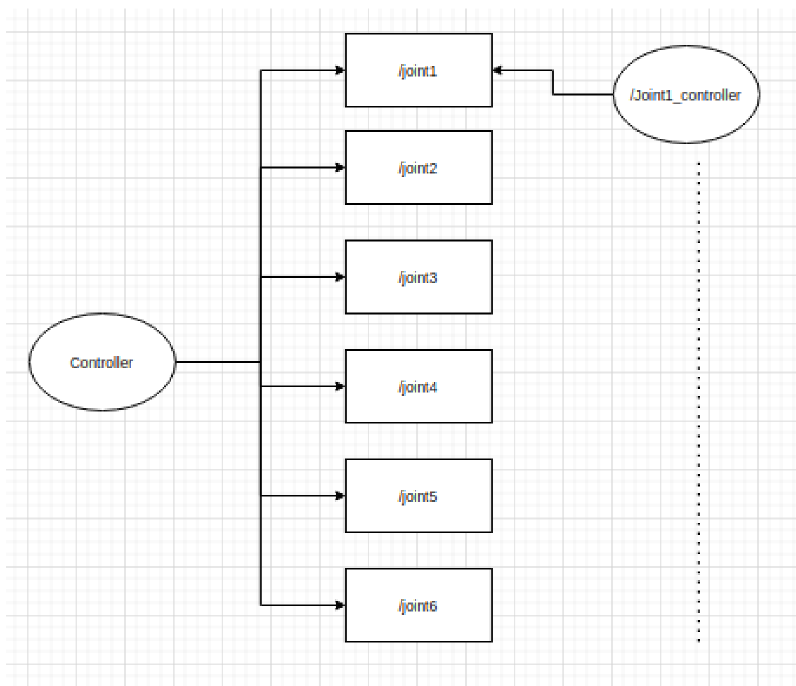

Figure: Draft architechture

Q5

Using skew theory, it is implemented in the code. The tests are being generated by gen_test.py using neihbourhood of the zero configuration