Introduction to Vision and Robotics

Coursework report

by Ivan Sun s1800640 & Michitatsu Sato s1807428

Part 2: Robot Vision

Algorithm:

In this part we used two python files named image1.py and image2.py to receive the image information from the two cameras and then process it. In image1.py we can get the image from camera01 and then get the coordinates of the y axis and z axis. If the joints might not be visible from the camera01, we cannot get the z coordinates(y will be the same as the ones which block the joints) and we will set the y coordinate 0 and we will get the y coordinates in the other py file. And we will publish the result as “/robot/y_z”. The same for image2.py and we will publish “/robot/x_z”. Now we create a py file named joint_state_estimation.py which will subscribe the two previous publishers using two callback functions and use another function to estimate the angles.

Validation:

t = 20

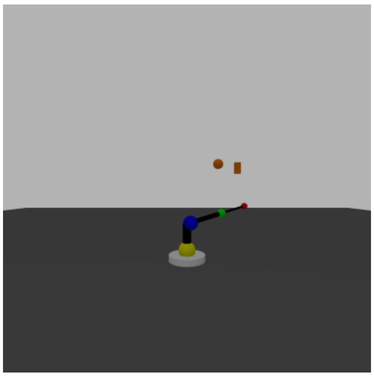
joint2 = (pi/2)*Sin[(pi/15)*20] = -1.360

joint3 = (pi/2)*Sin[(pi/18)*20] = -0.537

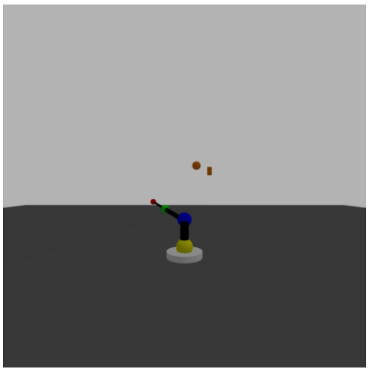
joint4 = (pi/2)*Sin[(pi/20)*20]] = 0.0

estimation:

joint2 = -1.289 joint3 = -0.542 joint4 = -0.096



image_copy_1.png



image_copy_2.png

t = 50

joint2 = (pi/2)*Sin[(pi/15)*50] = -1.360

joint3 = (pi/2)*Sin[(pi/18)*50] = 1.010

joint4 = (pi/2)*Sin[(pi/20)*50] = 1.571

estimation:

joint2 = -1.530 joint3 = 0.986 joint4 = 1.648

3. Robot Control

Part 3.1: Forward Kinematics

The coordinate of the end effector can be obtained by the part of the Transformation Matrix (*T(q)*) for the Robot joints.

To obtain the T(q), first D-H table have to be constructed (q = [$\theta_1, \theta_2, \theta_3, \theta_4$]).

	α	a	d	θ
Link 1	$\frac{\pi}{2}$	0	2.5	$\theta_1 + \frac{\pi}{2}$
Link	$\frac{\pi}{2}$	0	0	$\theta_2 + \frac{\pi}{2}$
Link 3	$-\frac{\pi}{2}$	3.5	0	θ_3
Link 4	0	3	0	θ_4

D-H Table for Robot

Next, from D-H Table construct the transform matrix for each joint (frames).

$$A_i^{i-1} = R_{z,\theta_i}\, Trans_{z,d_i}\, Trans_{x,a_i}\, R_{x,\alpha_i} = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i)\cos(\alpha_i) & \sin(\theta_i)\sin(\alpha_i) & a_i\cos(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) & -\cos(\theta_i)\sin(\alpha_i) & a_i\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ therefore,

$$A_1^0 = \begin{bmatrix} \cos(\theta_1 + \frac{\pi}{2}) & 0 & \sin(\theta_1 + \frac{\pi}{2}) & 0 \\ \sin(\theta_1 + \frac{\pi}{2}) & 0 & -\cos(\theta_1 + \frac{\pi}{2}) & 0 \\ 0 & 1 & 0 & 2.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} A_2^1 = \begin{bmatrix} \cos(\theta_2 + \frac{\pi}{2}) & 0 & \sin(\theta_2 + \frac{\pi}{2}) & 0 \\ \sin(\theta_2 + \frac{\pi}{2}) & 0 & -\cos(\theta_2 + \frac{\pi}{2}) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} A_3^2 = \begin{bmatrix} \cos(\theta_3) & 0 & -\sin(\theta_3) & 3.5\cos(\theta_3) \\ \sin(\theta_3) & 0 & \cos(\theta_3) & 3.5\sin(\theta_3) \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} A_4^3 = \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & 3\cos(\theta_4) \\ \sin(\theta_4) & \cos(\theta_4) & 0 & 3\sin(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T(q) = A_1^0 A_2^1 A_3^2 A_4^3 = \begin{bmatrix} \cos\left(\theta_1 + \frac{\pi}{2}\right) & 0 & \sin\left(\theta_1 + \frac{\pi}{2}\right) & 0 \\ \sin\left(\theta_1 + \frac{\pi}{2}\right) & 0 & -\cos\left(\theta_1 + \frac{\pi}{2}\right) & 0 \\ 0 & 1 & 0 & 2.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\left(\theta_2 + \frac{\pi}{2}\right) & 0 & \sin\left(\theta_2 + \frac{\pi}{2}\right) & 0 \\ \sin\left(\theta_2 + \frac{\pi}{2}\right) & 0 & -\cos\left(\theta_2 + \frac{\pi}{2}\right) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta_3) & 0 & -\sin(\theta_3) & 3.5\cos(\theta_3) \\ \sin(\theta_3) & 0 & \cos(\theta_3) & 3.5\sin(\theta_3) \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta_4) & -\sin(\theta_4) & 0 & 3\cos(\theta_4) \\ \sin(\theta_4) & \cos(\theta_4) & 0 & 3\sin(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \sin(\theta_1)\sin(\theta_2) & \cos(\theta_1) & -\sin(\theta_1)\cos(\theta_2) & 0 \\ -\sin(\theta_2)\cos(\theta_1) & \sin(\theta_1) & \cos(\theta_1)\cos(\theta_2) & 0 \\ \cos(\theta_2) & 0 & \sin(\theta_2) & 2.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta_3)\cos(\theta_4) & -\cos(\theta_3)\sin(\theta_4) & -\sin(\theta_3) & 3.5\cos(\theta_3) + 3\cos(\theta_3)\cos(\theta_4) \\ \sin(\theta_3)\cos(\theta_4) & -\sin(\theta_3)\sin(\theta_4) & \cos(\theta_3) & 3.5\sin(\theta_3) + 3\sin(\theta_3)\cos(\theta_4) \\ -\sin(\theta_4) & -\cos(\theta_4) & 0 & -3\sin(\theta_4) \\ 0 & 0 & 0 & 1 \end{bmatrix} = A_2^0 A_4^2$$

Coordinate of the end-effector (x, y, z) is corresponding to the T(q)'s last element of 1st, 2nd and 3rd row. Which means to obtain the coordinate of end-effector, whole matrix T(q) does not have to be calculated. By using corresponding row and column of $A_2^0, and\ A_4^2$, the desired part of the T(q) can be calculated (corresponding row and column is colored in above).

Estimated end-effector coordinate (x, y, z) is:

$$x_e = \sin(\theta_1)\,(\sin(\theta_2)\cos(\theta_3)\,(3\cos(\theta_4) + 3.5) + 3\cos(\theta_4)\sin(\theta_2)) + \cos(\theta_1)\sin(\theta_3)\,(3\cos(\theta_4) + 3.5)$$

$$y_e = \cos(\theta_1)\,(\sin(\theta_2)\cos(\theta_3)\,(-3\cos(\theta_4) - 3.5) - 3\cos(\theta_2)\sin(\theta_4)) + \sin(\theta_1)\sin(\theta_3)\,(3\cos(\theta_4) + 3.5)$$

$$z_e = \cos(\theta_2)\cos(\theta_3)(3\cos(\theta_4) + 3.5) - 3sin(\theta_2)\,sin(\theta_4) + 2.5$$

Below Table is result of comparing estimated end-effector coordinate using Forward kinematics with the coordinate obtained by the computer vision implemented in previous part. The average error (difference in Euclidian distance between two points) is 0.81m. This is quite big. However, this does not mean the two point is distant through all the 10 experiments. Actually, the two points are sometimes very near, and sometimes very far away. By observing the image obtained from camera 1 and 2, it can be concluded that when the end-effector is distant from camera, it makes computer vision difficult to calculate the accurate coordinate of the end-effector. The maximum error of the z-coordinate is less than maximum error of the x and y coordinate. This also can be evidence of the conclusion considered above (z-coordinate will be less sensitive to the distance between end-effector and cameras).

$q = [\theta_1, \theta_2, \theta_3, \theta_4]$	Forward Kinematics xyz-coordinate	Computer Vision xyz-coordinate	$q = [\theta_1, \theta_2, \theta_3, \theta_4]$	Forward Kinematics xyz-coordinate	Computer Vision xyz-coordinate
[1.6, 1.6, 1.6, -1.6]	[-0.11, 3.41, 5.5]	[0.16 ,3.96, 5.28]	[0.8, 1.6, 1.6, -1.6]	[2.37, 2.46, 5.5]	[2.51, 3.36, 5.32]
[1.6, -1.6, 1.6, 1.6]	[-0.09, 3.41, 5.5]	[0.0, 3.96, 5.28]	[-0.8, -1.6, -1.6, 1.6]	[-2.39, 2.44, 5.5]	[-2.0, 2.68, 5.32]
[1.6, 1.6, -1.6, -1.6]	[0.09, -3.41, 5.5]	[0.32, -3.4, 5.32]	[1, 1, 1, -1]	[3.14, 3.1, 6.12]	[3.12, 4.32, 6.08]
[-1.6, 1.6, 1.6, -1.6]	[-0.09, -3.41, 5.5]	[0.04, -3.36, 5.32]	[1, 0.3, 1.8, 0.5]	[4.04, 4.51, 0.74]	[4.2, 6.6, 0.16]
[-1.6, 1.6, -1.6, -1.6]	[0.11, 3.41, 5.5]	[0.36, 4.04, 5.32]	[-1, -1, 0.5, 0.5]	[4.75, -0.45, 6.62]	[5.52, -0.44, 6.92]

End-effector position estimated by Forward kinematics and Computer vision with 10 different joints state

Part 3.2: Closed-loop Control