UDACITY ROBOTICS KINEMATICS PROJECT

Cheshta Dhingra

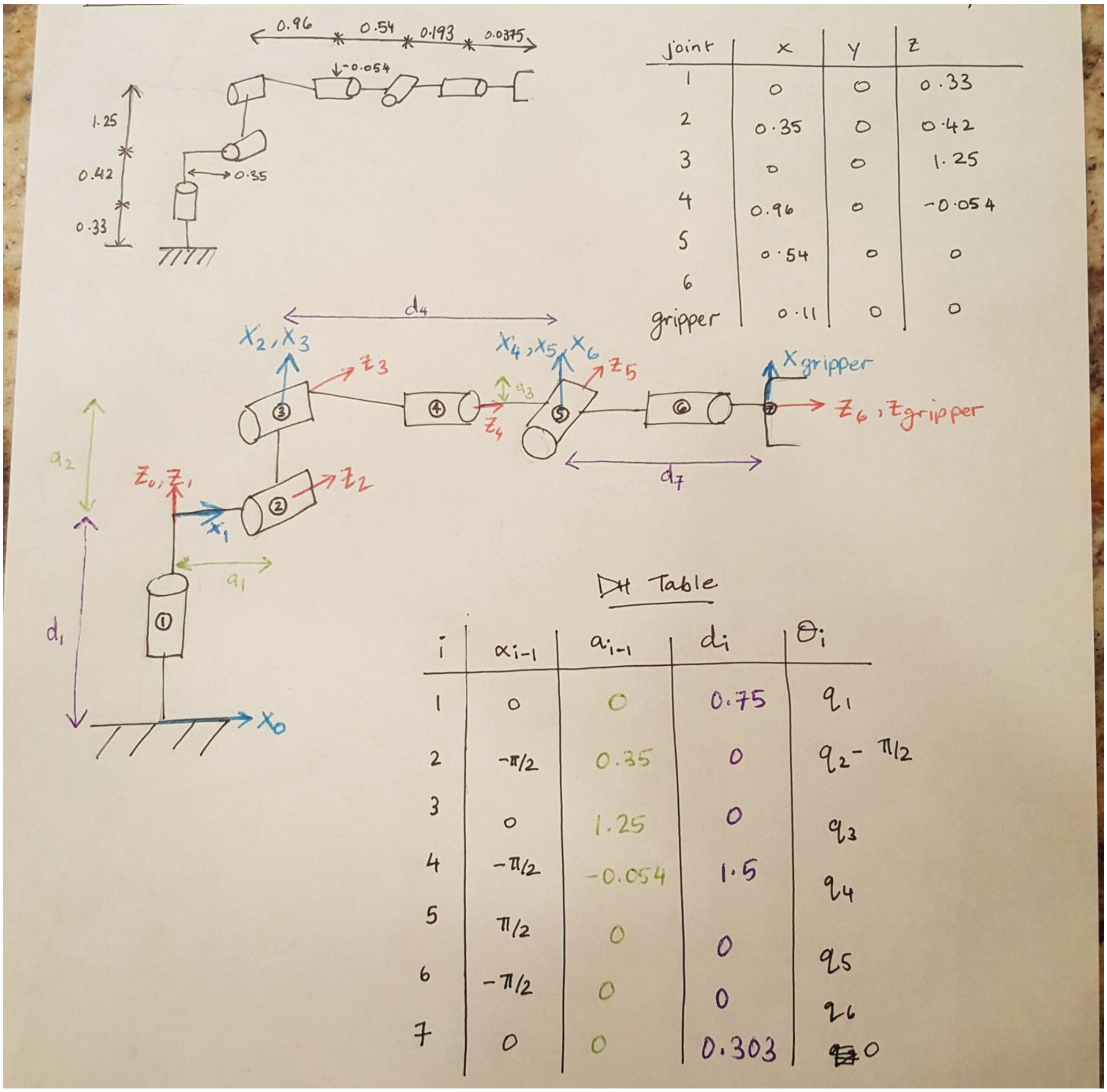
September 25th, 2017

1. Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf.

You're reading it!

Kinematic Analysis

1. Run the forward_kinematics demo and evaluate the kr210.urdf.xacro file to perform kinematic analysis of Kuka KR210 robot and derive its DH parameters.



2. Using the DH parameter table you derived earlier, create individual transformation matrices about each joint. In addition, also generate a generalized homogeneous transform between base_link and gripper_link using only end-effector(gripper) pose.

In order to calculate the homogenous transform matrices, I used a function called TF_matrix, which takes in values for *alpha, a, d and q* and generates the modified DH transformation matrix. Below are the outputs of the individual transform matrices about each joint, as well as the generalized homogenous transform between the base link and the gripper, which is calculate by multiplying individual transform matrices:

```
T0_1 = [[       cos(q1),   -sin(q1),     0,       0],
        [       sin(q1),    cos(q1),     0,       0],
        [            0,          0,     1,    0.75],
        [            0,          0,     0,       1]]
```

```
T1_2 =   [[     sin(q2),    cos(q2),     0,     0.35],
        [           0,          0,     1,        0],
        [     cos(q2),    -sin(q2),     0,        0],
        [           0,          0,     0,        1]]
```

```
T2_3 =  [[      cos(q3),    -sin(q3),    0,      1.25],
        [       sin(q3),     cos(q3),    0,         0],
        [            0,          0,     1,         0],
        [             0,          0,     0,         1]]
```

```
T3_4 =  [[      cos(q4),    -sin(q4),      0,   -0.054],
        [            0,          0,     1,      1.5],
        [     -sin(q4),    -cos(q4),     0,        0],
        [             0,          0,     0,        1]]
```

```
T4_5 =  [[      cos(q5),     -sin(q5),     0,         0],
        [            0,           0,    -1,        0],
        [     sin(q5),       cos(q5),     0,        0],
        [            0,           0,     0,        1]]
```

```
T5_6 =  [[       cos(q6),    -sin(q6),      0,        0],
        [             0,          0,      1,        0],
        [      -sin(q6),    -cos(q6),      0,        0],
        [              0,          0,      0,        1]]
```

```
T6_Gripper = [[       1,             0,      0,        0],
             [       0,             1,      0,        0],
             [       0,             0,      1,   0.303],
             [       0,             0,      0,        1]]
```

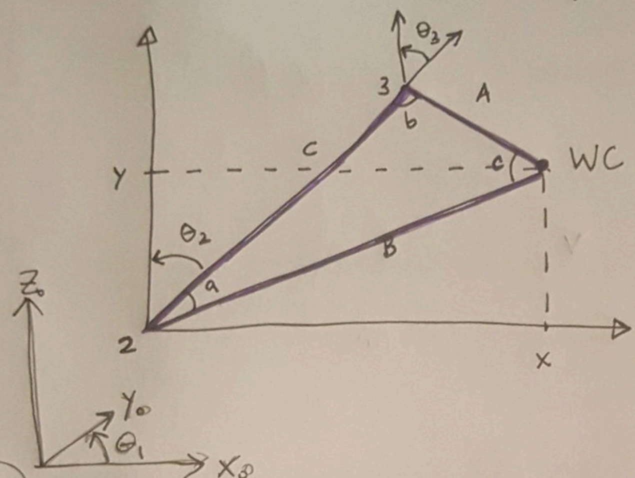
```
T0_Gripper =
```

((sin(q1)*sin(q4) + sin(q2 + q3)* cos(q1)*cos(q4))*cos(q5) + sin(q5) *cos(q1)*cos(q2 + q3))*cos(q6) - (-sin(q1)*cos(q4) + sin(q4)*sin(q2 + q3)*cos(q1))*sin(q6)	-((sin(q1)*sin(q4) + sin(q2 + q3)*cos(q1)*cos(q4)) *cos(q5) + sin(q5) *cos(q1)*cos(q2 + q3))*sin(q6) + (sin(q1)*cos(q4) - sin(q4)*sin(q2 + q3)*cos(q1))*cos(q6)	-(sin(q1)* sin(q4)+sin(q2 + q3)*cos(q1) *cos(q4))*sin(q5) + cos(q1)* cos(q5)*cos(q2 + q3)	-0.303* sin(q1)* sin(q4)*sin(q5) + 1.25*sin(q2)*cos(q1) - 0.303 *sin(q5)* sin(q2 + q3)* cos(q1)*cos(q4) - 0.054*sin(q2 + q3)* cos(q1) + 0.303*cos(q1)*cos(q5)*cos(q2 + q3) +1.5*cos(q1)*cos(q2 + q3) + 0.35*cos(q1)
((sin(q1)*sin(q2 + q3)*cos(q4) - sin(q4)*cos(q1))*cos(q5) + sin(q1)*sin(q5)*cos(q2 + q3))*cos(q6) - (sin(q1)*sin(q4)*sin(q2 + q3) + cos(q1)*cos(q4))*sin(q6)	-((sin(q1)*sin(q2 + q3)*cos(q4) - sin(q4)*cos(q1)) *cos(q5) + sin(q1)*sin(q5) *cos(q2 + q3))*sin(q6) - (sin(q1)*sin(q4) *sin(q2 + q3) + cos(q1)*cos(q4)) *cos(q6)	-(sin(q1)* sin(q2 + q3)*cos(q4) - sin(q4)*cos(q1)) *sin(q5) + sin(q1)*cos(q5) *cos(q2 + q3)	1.25*sin(q1)*sin(q2) - 0.303* sin(q1)* sin(q5) *sin(q2 + q3)* cos(q4) - 0.054* sin(q1) *sin(q2 + q3) + 0.303* sin(q1)* cos(q5) *cos(q2 + q3) + 1.5* sin(q1) *cos(q2 + q3) + 0.35* sin(q1) + 0.303* sin(q4)* sin(q5)*cos(q1)
-(sin(q5)*sin(q2 + q3) - cos(q4)*cos(q5)*cos(q2 + q3))*cos(q6) - sin(q4)*sin(q6)*cos(q2 + q3)	(sin(q5)*sin(q2 + q3) - cos(q4)*cos(q5)*cos(q2 + q3))*sin(q6) - sin(q4)*cos(q6)*cos(q2 + q3)	sin(q5)*cos(q4)*cos(q2 + q3) - sin(q2 + q3)*cos(q5)	-0.303* sin(q5)* cos(q4)*cos(q2 + q3) - 0.303* sin(q2 + q3)* cos(q5) - 1.5* sin(q2 + q3) + 1.25*cos(q2) - 0.054*cos(q2 + q3) + 0.75
0	0	0	1

3. Decouple Inverse Kinematics problem into Inverse Position Kinematics and inverse Orientation Kinematics; doing so derive the equations to calculate all individual joint angles.

Joint Angle Calculation Using Geometric IK method

$$WC = EE - 0.303\, z_0^{EE} = \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix}$$



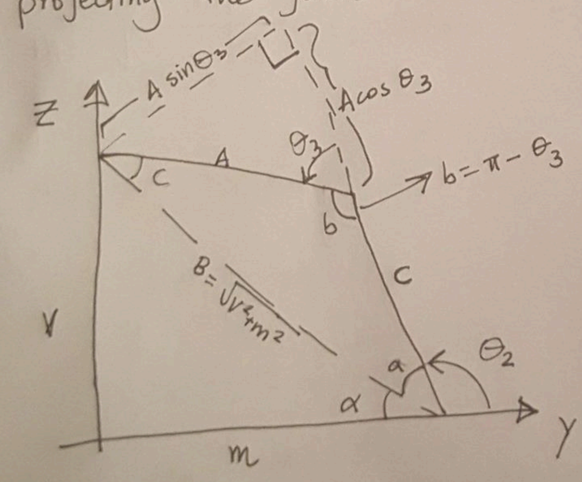
$A = \sqrt{1.5^2 + 0.054^2} = 1.501$

$C = 1.25$

1

$\theta_1 = \operatorname{atan2}(W_y, W_x)$

projecting the joints onto the z-y plane...



$V = W_z - d_1 = W_z - 0.75$

$m = \sqrt{W_x^2 + W_y^2} - a_1$

$\;\;= \sqrt{W_x^2 + W_y^2} - 0.35$

$B = \sqrt{V^2 + m^2}$

link 3 offset = $\operatorname{atan2}(a_3, d_3)$

Cosine LAW

$A^2 + C^2 = B^2 + 2AB\cos B$

$A^2 + C^2 = V^2 + m^2 + 2AB\cos(\pi - \theta_3) = V^2 + m^2 + 2AB\cos(\theta_3)$

$$\therefore \cos(\theta_3) = \frac{(A^2 + C^2 - v^2 - m^2)}{2AC}$$

$$\therefore \sin(\theta_3) = \pm\sqrt{1 - \cos^2\theta_3}$$

(2)

$$\left.\begin{aligned} \alpha &= \text{atan2}(v, A) \\ a &= \text{atan2}(a\sin\theta_3,\ C + a\cos\theta_3) \\ \theta_2 &= \pi - \alpha - a \end{aligned}\right\} \quad q_2 = \frac{\pi}{2} - \text{atan2}(v, A) - \text{atan2}(a\sin\theta_3,\ C + a\cos\theta_3)$$

(3)

$$\theta_3 = \text{atan}(\sin\theta_3, \cos\theta_3) \Big\} \quad q_3 = \text{atan2}(\sin\theta_3,\ \cos\theta_3) - \frac{\pi}{2} - \text{offset}$$

$$\varphi = \text{atan2}(a_3, d_3)$$

$$R_0^3 = R_0^1 \times R_1^2 \times R_2^3.$$

Since $R_0^6 = R_0^3 \times R_3^6$

$$R_3^6 = (R_0^3)^{-1}(R_0^6) \Rightarrow (R_0^3)^T(R_0^6).$$

<u>Transpose = Inverse</u>

<u>Euler Angle Transformation.</u>

From FK:

$$R_3^6 = \begin{bmatrix} -S_4S_6 + C_4C_5C_6 & -S_4C_6 - C_4C_5C_6 & -C_4C_5 \\ S_5C_6 & -S_5S_6 & C_5 \\ -C_4S_6 - S_4C_5C_6 & -C_4C_6 + S_4C_5S_6 & S_4S_5 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

From this, we know

$$C_5 = r_{23}$$

$$S_5 = \pm\sqrt{1 - r_{23}^2}$$

(5) $q_5 = \text{atan2}(S_5, C_5)$

For each pose, there are 4 possible solutions: 2 for wrist center, 2 for Euler angle transformation.

(4), (6)

if $s_5 > 0$	$s_5 < 0$	~~$s_5 = 0$~~
$q_4 = \text{atan2}(r_{33}, -r_{13})$ $q_6 = \text{atan2}(-r_{22}, r_{21})$	$q_4 = \text{atan2}(-r_{33}, r_{13})$ $q_6 = \text{atan2}(r_{22}, -r_{21})$	$R = \begin{bmatrix} C_{4+6} & S_{4+6} & 0 \\ -S_{4+6} & -C_{4+6} & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $\Rightarrow q_4 + q_6 = \text{atan2}(r_{12}, r_{11})$ infinite solution.

Project Implementation

1. Fill in the `IK_server.py` file with properly commented python code for calculating Inverse Kinematics based on previously performed Kinematic Analysis. Your code must guide the robot to successfully complete 8/10 pick and place cycles. Briefly discuss the code you implemented and your results.

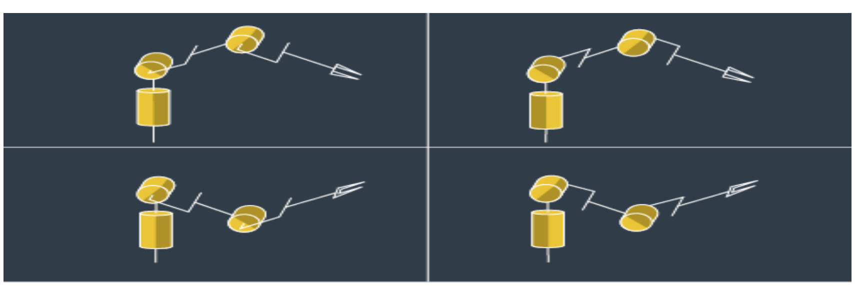
1. I completed the DH parameter table for the Kuka KR210
2. I calculate the location of the spherical wrist center, i.e., point O4 / O5 / O6 in the DH model, relative to the base frame using the following equation:

$$^{0}\boldsymbol{r}_{WC/0} = {}^{0}\boldsymbol{r}_{EE/0} - d \cdot {}^{0}_{6}R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} - d \cdot {}^{0}_{6}R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

3. Using trigonometry, I solved for the first 3 joint angles
4. Once the first three joint variables are known, I calculated ${}^{0}_{3}R$ via application of homogeneous transforms up to the WC.
5. I found a set of Euler angles corresponding to the rotation matrix:

$${}^{3}_{6}R = \left({}^{0}_{3}R\right)^{-1} {}^{0}_{6}R = \left({}^{0}_{3}R\right)^{T} {}^{0}_{6}R$$

6. I chose the correct solution among the set of possible solutions (2 per WC position, and 2 more per Euler angle transformation, as shown below) based on the least possible distance between poses.



In order to improve my submission further, I would like to explore the error plot for my end effector position generated by my joint angle commands. While working on this project, I had a trouble with executing the proper motion before I implemented the minimum distance portion. I received help from a friend who is not in this course to implement this idea, and wanted to credit him fairly.