Project: Kinematics Pick & Place

Writeup Template: You can use this file as a template for your writeup if you want to submit it as a markdown file, but feel free to use some other method and submit a pdf if you prefer.

Steps to complete the project:

1. Set up your ROS Workspace.

2. Download or clone the [project repository](https://github.com/udacity/RoboND-Kinematics-Project) into the ***src*** directory of your ROS Workspace.

3. Experiment with the forward_kinematics environment and get familiar with the robot.

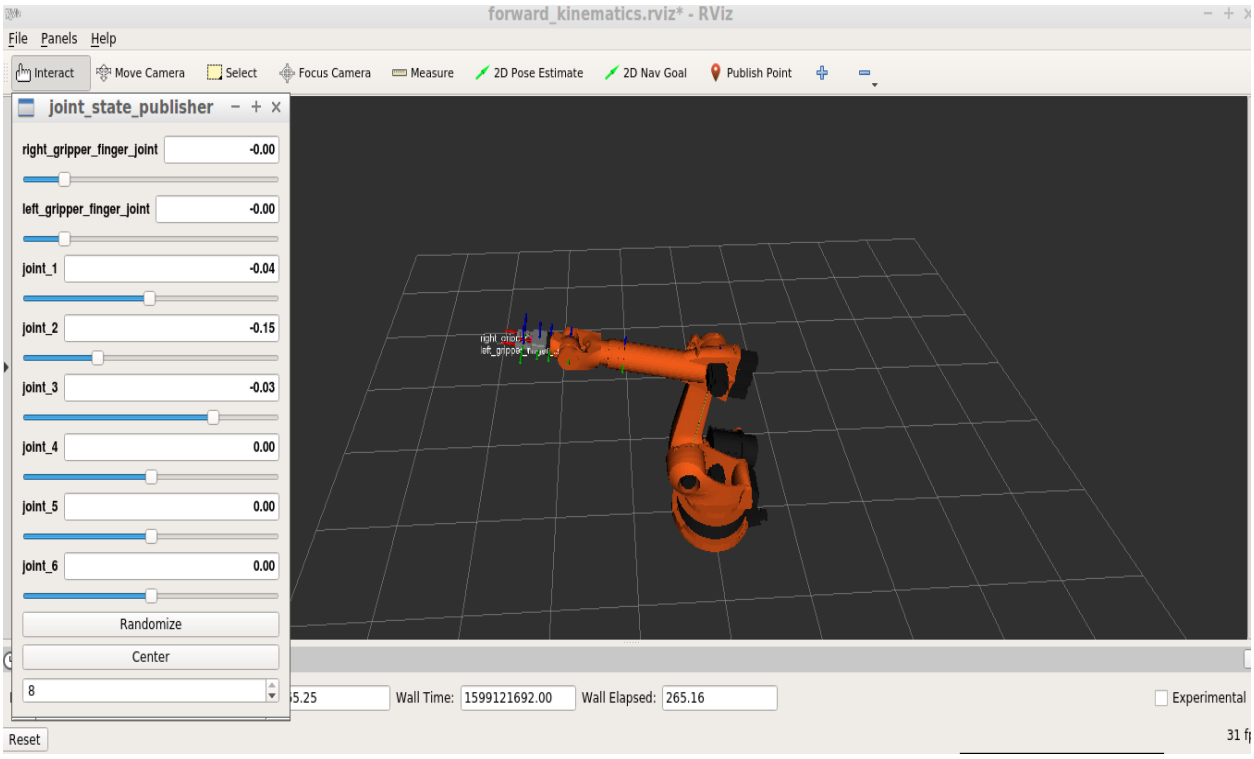
4. Launch in [demo mode](https://classroom.udacity.com/nanodegrees/nd209/parts/7b2fd2d7-e181-401e-977a-6158c77bf816/modules/8855de3f-2897-46c3-a805-628b5ecf045b/lessons/91d017b1-4493-4522-ad52-04a74a01094c/concepts/ae64bb91-e8c4-44c9-adbe-798e8f688193).

5. Perform Kinematic Analysis for the robot following the [project rubric](https://review.udacity.com/#!/rubrics/972/view).

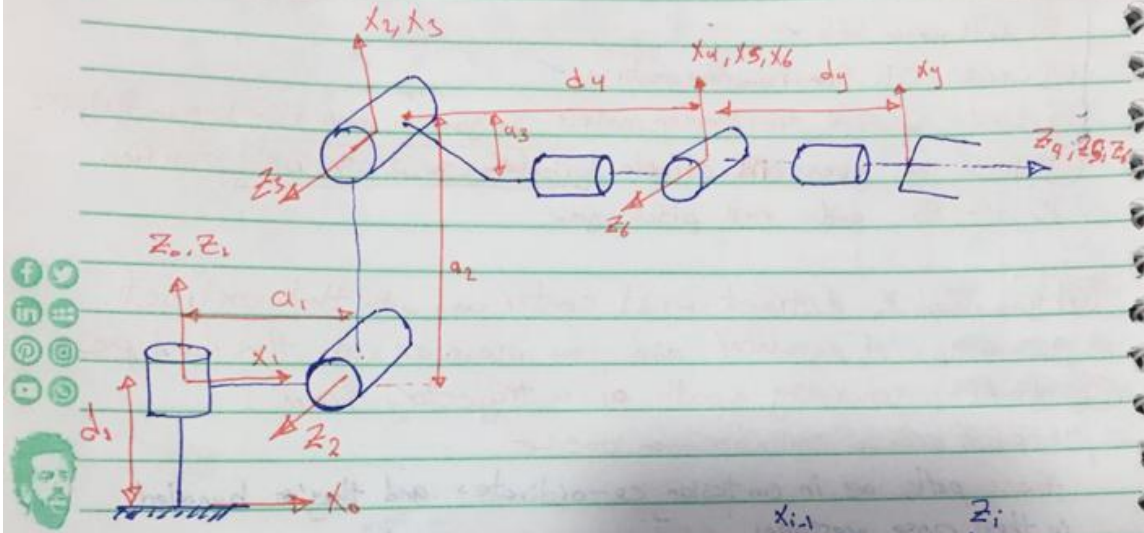
6. Fill in the `IK_server.py` with your Inverse Kinematics code.

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.



Page: Date:



$d_4 = \sqrt{0.054 + 1.50^2} =$

	X_{i-1}		Z_i	
Link	a	alpha	d	theta
1	0.35	Pi/2	0.75	theta1
2	1.25	0	0	theta + Pi/2
3	0.054	Pi/2	0	theta 3
4	0	-Pi/2	d4	theta 4
5	0	Pi/2	0	theta 5
6	0	0	0	theta 6
EE	0	0	0.303	0

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.

 # DH Parameters

'''

DH Table

Link	a	alpha	d	theta
1	a1	90	d1	*
2	a2	0	0	*
3	-a3	90	d3	*
4	0	-90	0	*
5	0	90	0	*
6	0	0	d6	*

'''

Page: Date:

To cal Transformation Matrix

$T_{0-1} \Rightarrow a=0 \quad \alpha=0 \quad d=d_1 \quad \theta=0$

$$T_{0-1} \Rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_{1-2} \Rightarrow a=a_1=0.35 \quad \alpha=-90 \quad d=0 \quad \theta=\theta_1$

$$T_{1-2} \Rightarrow \begin{bmatrix} C_1 & -S_1 & 0 & a_1 \\ 0 & 0 & 1 & 0 \\ -S_1 & -S_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_{2-3} \Rightarrow a=a_2 \quad \alpha=0 \quad d=0 \quad \theta_2=\theta_2+90$

$$T_{2-3} \Rightarrow \begin{bmatrix} C_2 & -S_2 & 0 & a_2 \\ S_2 & C_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_{3-4} \Rightarrow a=-a_3 \quad \alpha=-90 \quad d=d_3 \quad \theta=\theta_3$

$$T_{3-4} \Rightarrow \begin{bmatrix} C_3 & -S_3 & 0 & -a_3 \\ 0 & 0 & 1 & d_3 \\ -S_3 & -C_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_{4-5} \Rightarrow a=0 \quad \alpha=90 \quad d=0 \quad \theta_4$

$$T_{4-5} \Rightarrow \begin{bmatrix} C_4 & -S_4 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ S_4 & C_4 & 0 & 0 \end{bmatrix}$$

Date:

$T_{5-4} \Rightarrow a = 0 \quad \alpha = -90 \quad d = 0 \quad \theta = \theta_5$

$$\begin{bmatrix} C_5 & -S_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -S_5 & -C_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_{6-g} \Rightarrow a = 0 \quad \alpha = 0 \quad d = d_6 \quad \theta = \theta_6$

$$\begin{bmatrix} C_6 & -S_6 & 0 & 0 \\ S_6 & C_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{0-g} = T_{0-1} * T_{1-2} * T_{2-3} * T_{3-4} * T_{4-5} * T_{5-6} * T_{6-g}$$

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

And here's where you can draw out and show your math for the derivation of your theta angles.

The first containing joints 1, 2, and 3, as well as the base link, arm 1 and arm 2. This 3DOF (RRR) setup can be considered as the elbow section of the arm. The inverse joint angle calculation of this elbow can then be derived by using basic trigonometry and the law of cosine.

overview

Joint 1: Angle from position in the x-y plane

Joint 2: Law of cosine in the x, y, z planes and perpendicular to the z-plane.

Joint 3: The difference in the law of cosine angle a continuation of link 2.

The remaining joints make up the wrist section of the arm, and joint angles have been derived using trigonometric laws and current robot orientation and Position.

Joint 4: Tangent to a centre point in the centre of the pickup shelve location. I.e., joint three will rotate around the centre point as if it was the tangent of the circle with a zero angle when at the bottom, pi/2 rotation angle on the left side, pi rotation when verticle and 3*pi/2 on the right side.

Joint 5: Angle is derived from the most direct line to the target pointing the gripper at the target.

Joint 6: negative of Joint 4's angle.

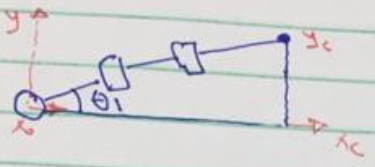
Below are the calculations of all 6 joint angles.

Page:

Date:

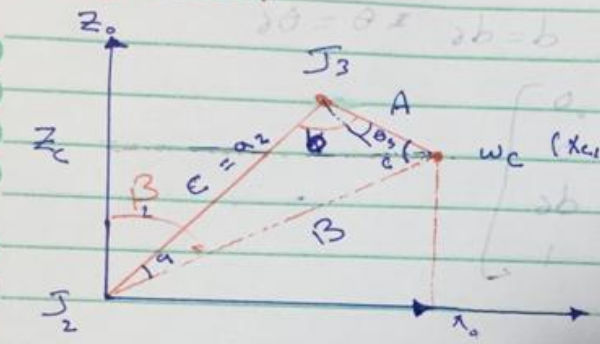
*Calculations of all 6 joint angles

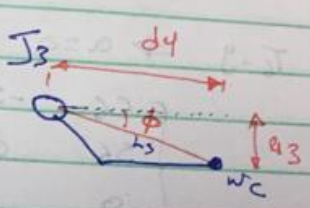
Joint (1)



$$\theta_1 = \tan^{-1}\left(\frac{Py_c}{Px_c}\right)$$

Joint (2)





$$x_c = P_x - d_6 \qquad x_0 = x_c - a_1 \cos\theta_1$$

$$z_c = P_z \qquad z_0 = z_c - d_1$$

$$y_c = P_y - d_6 \qquad y_0 = y_c - a_1 \sin\theta_1$$

$$\beta = \sqrt{x_c^2 + y_c^2 + z_c^2} \qquad \beta_2 = \tan^{-1}\left(\frac{x_c}{z_c}\right) \quad [1]$$

$$\cos(\alpha) = \left(\frac{B^2 + c^2 - A^2}{2BC}\right) \quad [2]$$

$$\therefore \boxed{\theta_2 = \beta_2 - \alpha} \quad [3]$$

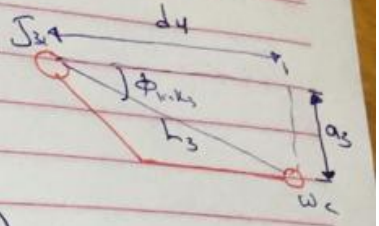
$$\alpha = \tan^{-1}\left(\frac{\pm\sqrt{1 - \cos^2\alpha}}{\cos\alpha}\right)$$

Date:

Join(3)

$$link(3) = \sqrt{a_3^2 + d_4^2}$$

$$\phi_{-link_3} = \tan^{-1}\left(\frac{a_3}{d_3}\right)$$

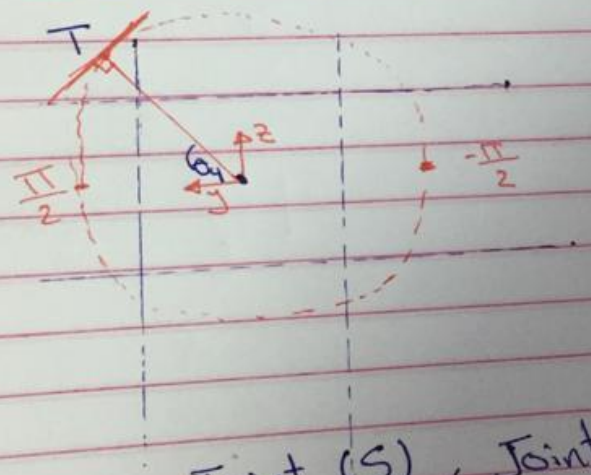


$$\cos(b) = (A^2 + c^2 - B^2)/(2Ac)$$

$$\Rightarrow b = \tan^{-1}\left(\frac{\pm\sqrt{1-\cos^2 b}}{\cos b}\right)$$

$$\theta_3 = b - \frac{\pi}{2} - \phi_{link_3}$$

Joint (4) (looking at the sketch plane)



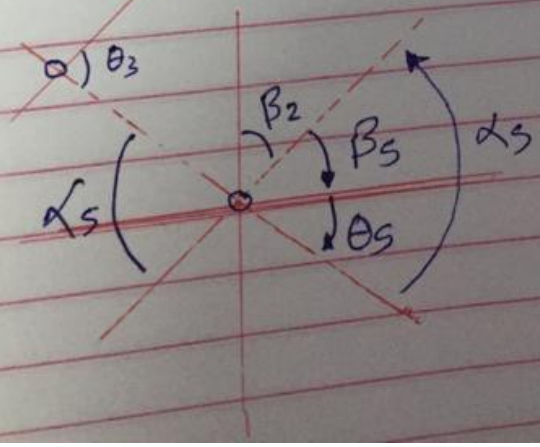
$$Z_o = d_1 + a_2$$

$$T_y = P_y$$

$$T_z = P_z - Z_o$$

$$\theta_4 = \frac{\pi}{4} \pm \tan^{-1}\left(\frac{T_z}{T_y}\right)$$

Joint (5), Joint (6)



$$\beta_5 = \frac{\pi}{2} - \beta_2$$

$$\theta_5 = \alpha_5 - \beta_5$$

$$\theta_6 = -\theta_4$$

The code for the homogeneous transform matrix calculation is located in the file IK_server.py, Lines 218 to 260. The code calculates the forward kinematics of the arm, and the returned transform matrix is then multiplied by rotation in the y then z axes. The rotation is required as the gripper reference frame is not in the same orientation as the base coordinate frame. The rotation order required is.

pi/2 about the y-axis.

pi about the z-axis

The rotation matrices code in implemented in the file IK_server.py, lines 46 to 80. the forward kinematics and rotation calculation is located at line 308.

roll: -0.0006073606506876212

pitch: 0.0003695153905658422

yaw: -0.00025537793197934487

T0_6=Matrix([

[0.717894479340743, 0.458023650217615, -0.52425361455446, 2.25323469114597],

[0.15094795990372, 0.632741176112161, 0.759508602619552, 0.970313341360017],

[0.679589751197727, -0.624382046421525, 0.385103661594485, 0.133727195518232],

[0, 0, 0, 1]])

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.

Here I'll talk about the code, what techniques I used, what worked and why, where the implementation might fail and how I might improve it if I were going to pursue this project further.

The code mentioned in the following outline can be found in unzip file the IK_server.py file located ./kuka_arm/scripts/.

The IK_server.py file is called externally with the desired gripper trajectory by the function handle_calculate_IK(), function lines 262 to 318. Once a valid arm trajectory has been passed into the function, the trajectory x, y, & z positions for specific points along the trajectory are first passed into the function arm_IK(), line 294, where the first three angles of the arm are

calculated. These angles correspond to the rotation around the room, theta 1, arm 1 position relative to the base frame, theta 2, and the arm 2 position relative to arm 1, theta 3.

The code for the function call arm_IK is located between lines 123 and 173. The code implements the calculations derived above with theta 1 calculated on line 134.

Note *

Before theta 2 and theta 3 are calculated, a new reference frame is calculated, removing the offsets from the base frame and the wrist. The code for the reference frame is on lines 138 to 151.

On the KR210 model there is an offset observed on arm 2 due to the mechanical positioning of the wrist motors. The correction of this offsets length and angle are located on lines 136 & 137.

Theta 2 angle calculations can now be calculated, as derived earlier, between lines 156 and 163.

Theta 3 angle calculation follows on from this and the code is located between lines 168 and 171.

After the calculation of the arm angles, the wrist angles are then calculated by calling the function **wrist_IK()** at line 296. The code for this function is located between lines 175 and 216.

The wrist rotation angle **theta 4** is calculated by creating a reference plane that represents the target shelf space, herein called target-frame, with an origin at the height and world frames y-axis value when the kr210 is at rest, theta 1 = 2 = 3 = 4 = 5 = 6 => zero.

Theta 4 is calculated so to keep theta 5's z-axis perpendicular to the target-frame origin point where angles closer to the ground will have an angle = 0, and the rotation angle will increase as the arm moves in a clockwise direction around the target-frame origin point amounting to a full rotation angle of 2*pi. That is as the robot looks to the left side of the target frame, **theta 4** rotation angle will be pi/2. The code is located between lines 190 and 197.

Theta 5 angle calculation follows on from this and uses the calculated arm end position to keep the gripper level at all times. The code is located between lines 200 and 209.

Theta 6 is the inverse of theta 4 to keep the gripper claws level when possible and the code is located at lines 211 and 214.

Screenshots of the completed pick and place process: