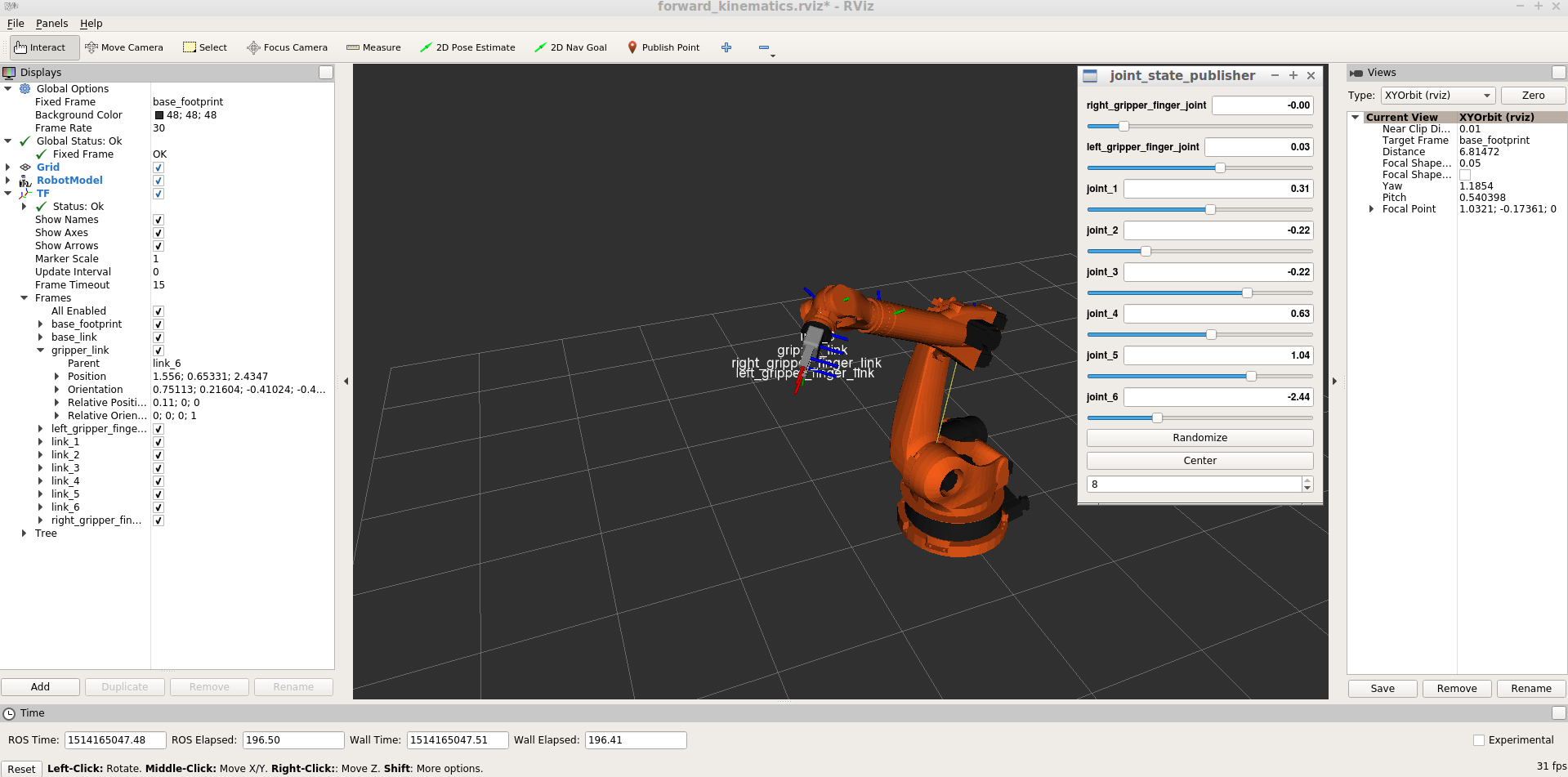
**Project: Kinematics Pick & Place**

**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.**

(1) Demo



(2) DH parameters:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Links** | **i - 1** | **alpha(i-1)** | **a(i-1)** | **d(i-1)** | **theta(i)** |
| 0->1 | 0 | 0 | 0 | 0 | 0 |
| 1->2 | 1 | - pi/2 | 0.35 | 0.75 | q2 - pi/2 |
| 2->3 | 2 | 0 | 1.25 | 0 | 0 |
| 3->4 | 3 | - pi/2 | - 0.054 | 0 | 0 |
| 4->5 | 4 | pi /2 | 0 | 1.5 | 0 |
| 5->6 | 5 | - pi/2 | 0 | 0 | 0 |
| 6->EE | 6 | 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.**

(1) Individual transformation matrices about each joint are:

Others are similar to above.

(2) Homogeneous transform between base\_link and gripper\_link: (Using sympy)

[ 7.49879891330929e-33\*((sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*cos(q5) - sin(q5)\*cos(q1)\*cos(q2 + q3))\*sin(q6) + 6.12323399573677e-17\*((sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*cos(q5) - sin(q5)\*cos(q1)\*cos(q2 + q3))\*cos(q6) - 1.0\*(sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*sin(q5) + 6.12323399573677e-17\*(sin(q1)\*cos(q4) + sin(q4)\*sin(q2 + q3)\*cos(q1))\*sin(q6) - 7.49879891330929e-33\*(sin(q1)\*cos(q4) + sin(q4)\*sin(q2 + q3)\*cos(q1))\*cos(q6) - 1.0\*cos(q1)\*cos(q5)\*cos(q2 + q3), -1.0\*((sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*cos(q5) - sin(q5)\*cos(q1)\*cos(q2 + q3))\*sin(q6) + 1.22464679914735e-16\*((sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*cos(q5) - sin(q5)\*cos(q1)\*cos(q2 + q3))\*cos(q6) + 1.22464679914735e-16\*(sin(q1)\*cos(q4) + sin(q4)\*sin(q2 + q3)\*cos(q1))\*sin(q6) + 1.0\*(sin(q1)\*cos(q4) + sin(q4)\*sin(q2 + q3)\*cos(q1))\*cos(q6), -1.22464679914735e-16\*((sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*cos(q5) - sin(q5)\*cos(q1)\*cos(q2 + q3))\*sin(q6) - 1.0\*((sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*cos(q5) - sin(q5)\*cos(q1)\*cos(q2 + q3))\*cos(q6) - 6.12323399573677e-17\*(sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*sin(q5) - 1.0\*(sin(q1)\*cos(q4) + sin(q4)\*sin(q2 + q3)\*cos(q1))\*sin(q6) + 1.22464679914735e-16\*(sin(q1)\*cos(q4) + sin(q4)\*sin(q2 + q3)\*cos(q1))\*cos(q6) - 6.12323399573677e-17\*cos(q1)\*cos(q5)\*cos(q2 + q3), -0.303\*(sin(q1)\*sin(q4) - sin(q2 + q3)\*cos(q1)\*cos(q4))\*sin(q5) + (1.25\*sin(q2) - 0.054\*sin(q2 + q3) + 1.5\*cos(q2 + q3) + 0.35)\*cos(q1) - 0.303\*cos(q1)\*cos(q5)\*cos(q2 + q3)],

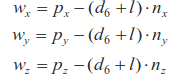
[-7.49879891330929e-33\*((sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*cos(q5) + sin(q1)\*sin(q5)\*cos(q2 + q3))\*sin(q6) - 6.12323399573677e-17\*((sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*cos(q5) + sin(q1)\*sin(q5)\*cos(q2 + q3))\*cos(q6) + 6.12323399573677e-17\*(sin(q1)\*sin(q4)\*sin(q2 + q3) - cos(q1)\*cos(q4))\*sin(q6) - 7.49879891330929e-33\*(sin(q1)\*sin(q4)\*sin(q2 + q3) - cos(q1)\*cos(q4))\*cos(q6) + 1.0\*(sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*sin(q5) - 1.0\*sin(q1)\*cos(q5)\*cos(q2 + q3), 1.0\*((sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*cos(q5) + sin(q1)\*sin(q5)\*cos(q2 + q3))\*sin(q6) - 1.22464679914735e-16\*((sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*cos(q5) + sin(q1)\*sin(q5)\*cos(q2 + q3))\*cos(q6) + 1.22464679914735e-16\*(sin(q1)\*sin(q4)\*sin(q2 + q3) - cos(q1)\*cos(q4))\*sin(q6) + 1.0\*(sin(q1)\*sin(q4)\*sin(q2 + q3) - cos(q1)\*cos(q4))\*cos(q6), 1.22464679914735e-16\*((sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*cos(q5) + sin(q1)\*sin(q5)\*cos(q2 + q3))\*sin(q6) + 1.0\*((sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*cos(q5) + sin(q1)\*sin(q5)\*cos(q2 + q3))\*cos(q6) - 1.0\*(sin(q1)\*sin(q4)\*sin(q2 + q3) - cos(q1)\*cos(q4))\*sin(q6) + 1.22464679914735e-16\*(sin(q1)\*sin(q4)\*sin(q2 + q3) - cos(q1)\*cos(q4))\*cos(q6) + 6.12323399573677e-17\*(sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*sin(q5) - 6.12323399573677e-17\*sin(q1)\*cos(q5)\*cos(q2 + q3), 0.303\*(sin(q1)\*sin(q2 + q3)\*cos(q4) + sin(q4)\*cos(q1))\*sin(q5) + (1.25\*sin(q2) - 0.054\*sin(q2 + q3) + 1.5\*cos(q2 + q3) + 0.35)\*sin(q1) - 0.303\*sin(q1)\*cos(q5)\*cos(q2 + q3)],

[ -7.49879891330929e-33\*(sin(q5)\*sin(q2 + q3) - cos(q4)\*cos(q5)\*cos(q2 + q3))\*sin(q6) - 6.12323399573677e-17\*(sin(q5)\*sin(q2 + q3) - cos(q4)\*cos(q5)\*cos(q2 + q3))\*cos(q6) - 6.12323399573677e-17\*sin(q4)\*sin(q6)\*cos(q2 + q3) + 7.49879891330929e-33\*sin(q4)\*cos(q6)\*cos(q2 + q3) - 1.0\*sin(q5)\*cos(q4)\*cos(q2 + q3) - 1.0\*sin(q2 + q3)\*cos(q5), 1.0\*(sin(q5)\*sin(q2 + q3) - cos(q4)\*cos(q5)\*cos(q2 + q3))\*sin(q6) - 1.22464679914735e-16\*(sin(q5)\*sin(q2 + q3) - cos(q4)\*cos(q5)\*cos(q2 + q3))\*cos(q6) - 1.22464679914735e-16\*sin(q4)\*sin(q6)\*cos(q2 + q3) - 1.0\*sin(q4)\*cos(q6)\*cos(q2 + q3), 1.22464679914735e-16\*(sin(q5)\*sin(q2 + q3) - cos(q4)\*cos(q5)\*cos(q2 + q3))\*sin(q6) + 1.0\*(sin(q5)\*sin(q2 + q3) - cos(q4)\*cos(q5)\*cos(q2 + q3))\*cos(q6) + 1.0\*sin(q4)\*sin(q6)\*cos(q2 + q3) - 1.22464679914735e-16\*sin(q4)\*cos(q6)\*cos(q2 + q3) - 6.12323399573677e-17\*sin(q5)\*cos(q4)\*cos(q2 + q3) - 6.12323399573677e-17\*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.**

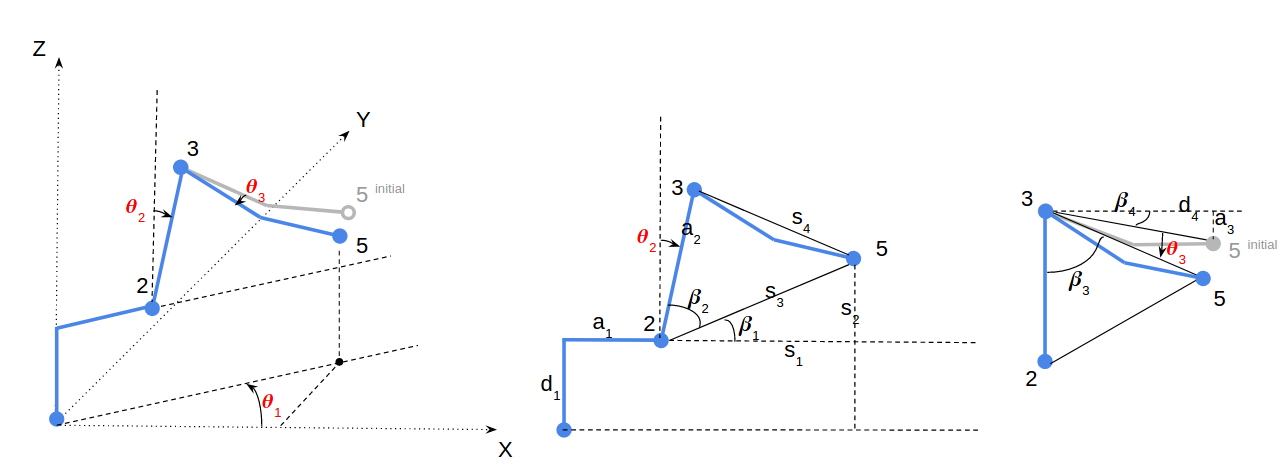
(1) Inverse Position:



To find nx, ny, and nz, the following equation will be used:

Rrpy = Rot(Z, yaw) \* Rot(Y, pitch) \* Rot(X, roll) \* R\_corr

To find theta1, theta2 and theta3, the following figure will be used:



Then, theta1 will be:

theta1 = atan2(wy, wx)

theta2 will be:

theta2 = pi/2 - beta1 - beta2

s1 = sqrt(wx^2 + wy ^2) – a1

s2 = wz – d1

s3 = sqrt(s1^2 + s2^2)

s4 = sqrt(a3^2 + d4^2)

beta1 = atan2(s2, s1)

beta2 = acos((a2^2 + s3^2 – s4^2)/(2\*a2\*s3))

theta3 will be:

theta3 = pi/2 – beta3 – beta4

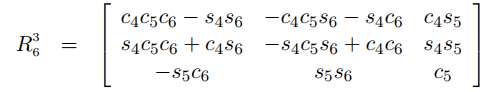
beta3 = acos((a2^2 + s4^2 – s3^2)/(2\*a2\*s4))

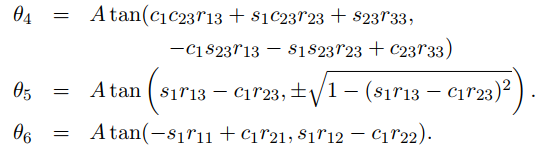
beta4 = atan2(a3, d4)

(2) Inverse Orientation:

R0\_6 = Rrpy

R3\_6 = inv(R0\_3) \* Rrpy





**Project Implementation**

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.



The robot has successfully finished the pick & place task.