

writeup

After many epic fail, this time it is so close to hit the target!!

<https://vimeo.com/259490068>

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.

You can find a screenshot when we run the robotic arm in the demo mode. The numerical values for the parameters is in the kr210.urdf.xarco file. It contains a lot of section. Movable joints are defined in the joints section in urdf file.



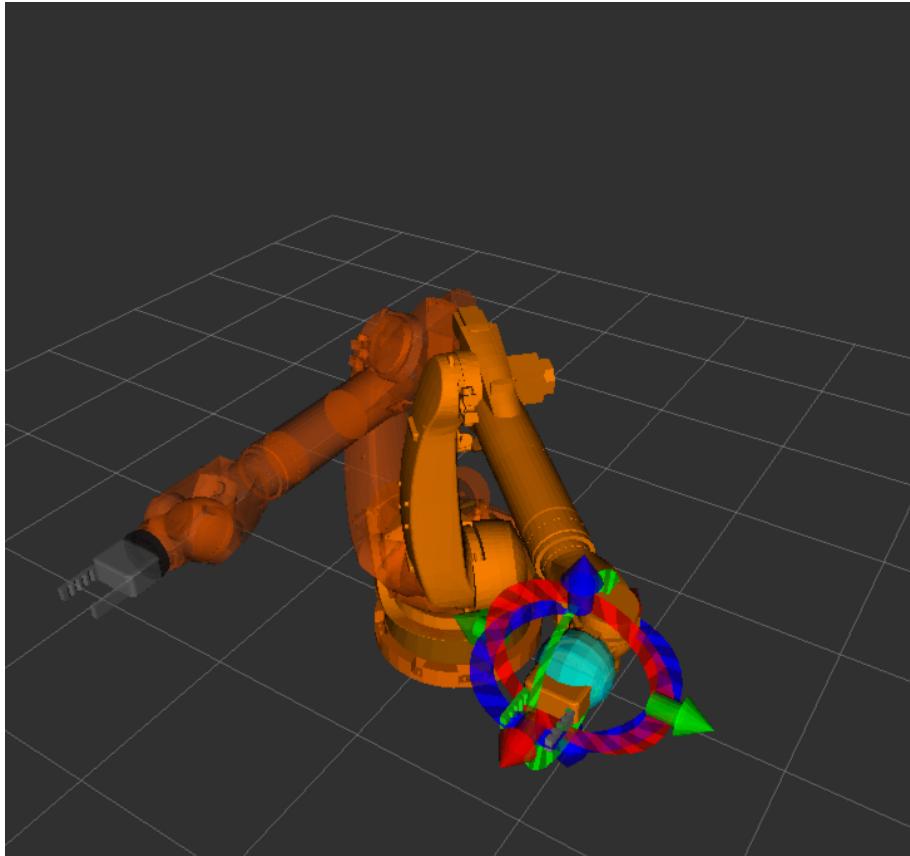
You can find the parameter values for each link below.

link_1	<input checked="" type="checkbox"/>
Alpha	1
Show Trail	<input type="checkbox"/>
Show Axes	<input type="checkbox"/>
Position	0; 0; 0.33
Orientation	0; 0; -1.2088e-05; 1
link_2	<input checked="" type="checkbox"/>
Alpha	1
Show Trail	<input type="checkbox"/>
Show Axes	<input type="checkbox"/>
Position	0.35; -8.4616e-06; 0.75
Orientation	2.6487e-10; 2.1911e-05; -1.2088e-05; 1
link_3	<input checked="" type="checkbox"/>
Alpha	1
Show Trail	<input type="checkbox"/>
Show Axes	<input type="checkbox"/>
Position	0.35005; -8.463e-06; 2
Orientation	-9.0734e-11; -7.5061e-06; -1.2088e-05; 1
link_4	<input checked="" type="checkbox"/>
Alpha	1
Show Trail	<input type="checkbox"/>
Show Axes	<input type="checkbox"/>
Position	1.3101; -3.1672e-05; 1.946
Orientation	1.0547e-05; -7.5062e-06; -1.2088e-05; 1

link_5	<input checked="" type="checkbox"/>
Alpha	1
Show Trail	<input type="checkbox"/>
Show Axes	<input type="checkbox"/>
Position	1.8501; -4.4727e-05; 1.946
X	1.85006
Y	-4.4727e-05
Z	1.94602
Orientation	1.0547e-05; -4.5811e-05; -1.2088e-05; 1
X	1.05469e-05
Y	-4.58105e-05
Z	-1.20883e-05
W	<input type="checkbox"/>
link_6	<input checked="" type="checkbox"/>
Alpha	1
Show Trail	<input type="checkbox"/>
Show Axes	<input type="checkbox"/>
Position	2.0431; -4.9393e-05; 1.946
Orientation	1.5457e-05; -4.5811e-05; -1.2088e-05; 1
right_gripper_finger_link	<input checked="" type="checkbox"/>
Alpha	1
Show Trail	<input type="checkbox"/>
Show Axes	<input type="checkbox"/>
Position	2.3031; -0.08246; 1.9461
Orientation	1.5457e-05; -4.5811e-05; -1.2088e-05; 1

You can interact with robotic arm when you run the following command.

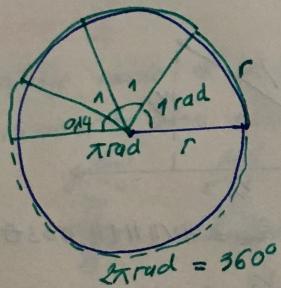
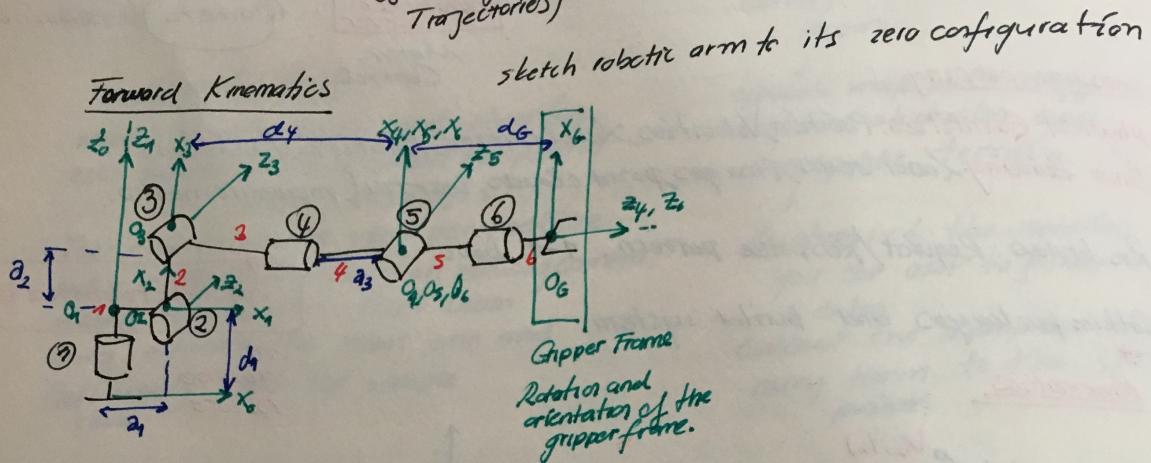
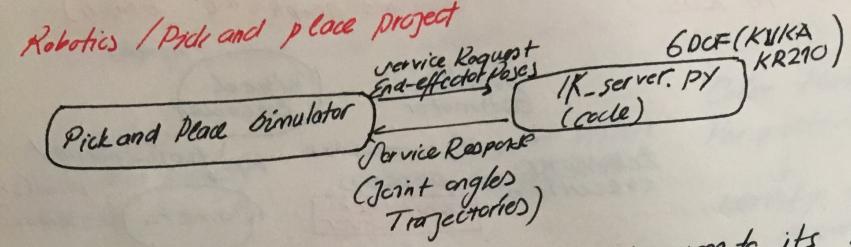
```
roslaunch kr210_claw_moveit demo.launch
```



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.

You can find the required explanation in my notes below.

Robotics / Pick and place project



- Degrees of freedom
- Joint angle
- Reference frame
- Serial manipulators
- Configuration space
- Joint space

kukararm/nrdf/
kr210.urdf.xacro

Euler angles → Komposit bir hareket roll, pitch, yaw gibi us hale getirilebilir.

$${}^A_B R_{xyz} = R_z(\alpha) R_y(\beta) R_x(\gamma)$$

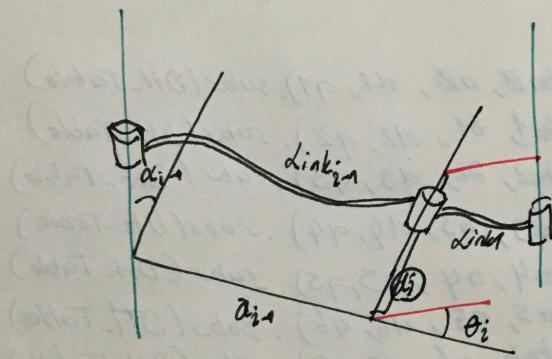
Attaching a reference frame to each link
 Homogeneous transform from the fixed base link to link 1 ... link $(n-1)$ to link n

$${}^0 T = {}^0 T {}^1 T {}^2 T \dots {}^{N-1} T$$

each trans requires
 wix independent parameters
 frame i relative to $i-1$
 three for position
 three for orientation

Denavit-Hartenberg (DH)
 only requires 4 parameters
 to describe position and orientation
 of neighbouring reference frames.

There are different
~~conventions~~
 we use the one in
John J. Craig's book



We can see all four
 parameters and how to
 calculate them.

Now, let's apply this and calculate DH table for Kuka KR210

i	α_{i-1}	a_{i-1}	d_i	θ_i
T_1^0	\emptyset	\emptyset	0,75	q_1
T_2^1	$-\pi/2$	0,35	0	$-\pi/2 + q_2$
T_3^2	\emptyset	1,25	0	q_3
T_4^3	$-\pi/2$	0,054	1,5	q_4
T_5^4	$\pi/2$	\emptyset	\emptyset	q_5
T_6^5	$-\pi/2$	\emptyset	\emptyset	q_6
T_7^6	\emptyset	\emptyset	0,303	\emptyset

TF DH Transformation matrix

$${}^{i-1}_iT \rightarrow \begin{bmatrix} \cos(q) & -\sin(q) & 0 & a \\ \sin(q)*\cos(\alpha) & \cos(q)*\cos(\alpha) & -\sin(\alpha) & -\sin(\alpha)*d \\ \sin(q)*\sin(\alpha) & \cos(q)*\sin(\alpha) & \cos(\alpha) & \cos(\alpha)*d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

explanation: how to calculate this matrix

$${}^{i-1}_iT = R_x(\alpha_{i-1}) D_x(a_{i-1}) \\ R_z(\theta_i) D_z(d_i)$$

constructed as a sequence of four basic transformation two rotation and two translation as shown above

You can use this transformation matrix for each joint.
just replace the α, a, d and q values.

$$T_{0-1} = \text{TF_Matrix}(\alpha_0, a_0, d_1, q_1). \text{subs(DH-Table)}$$

$$T_{1-2} = \text{TF_Matrix}(\alpha_1, a_1, d_2, q_2). \text{subs(DH-Table)}$$

$$T_{2-3} = \text{TF_Matrix}(\alpha_2, a_2, d_3, q_3). \text{subs(DH-Table)}$$

$$T_{3-4} = \text{TF_Matrix}(\alpha_3, a_3, d_4, q_4). \text{subs(DH-Table)}$$

$$T_{4-5} = \text{TF_Matrix}(\alpha_4, a_4, d_5, q_5). \text{subs(DH-Table)}$$

$$T_{5-6} = \text{TF_Matrix}(\alpha_5, a_5, d_6, q_6). \text{subs(DH-Table)}$$

$$T_{6-EE} = \text{TF_Matrix}(\alpha_6, a_6, d_7, q_7). \text{subs(DH-Table)}$$

And transform matrix from base to gripper.

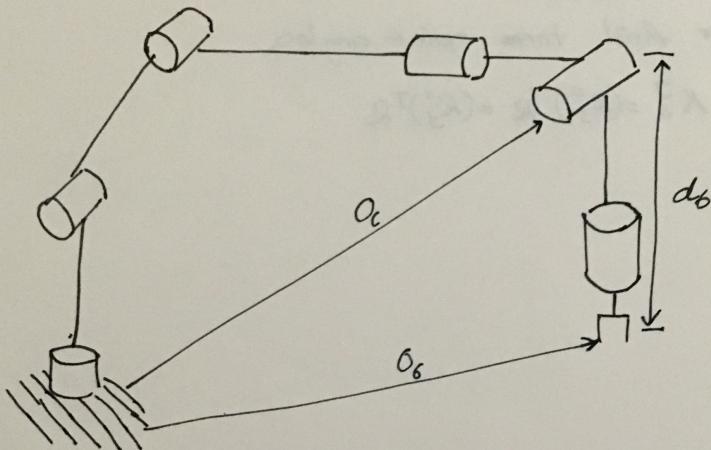
$$T_{0-EE} = T_{0-1} * T_{1-2} * T_{2-3} * T_{3-4} * T_{4-5} * T_{5-6} * T_{6-EE}$$

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

WC → Wrist Center
EE → End Effector.

Decoupling Inverse Kinematics into Inverse Position Kinematics and Inverse Rotation Kinematics.

The robotic arm has 6DOF serial manipulator.
First three joints could be used to control the position of the wrist center
the last three joints would orient the end effector as needed



Two sets of equations representing rotational and positional equations

$$R^o(q_1, \dots, q_6) = R$$

$$O^o(q_1, \dots, q_6) = O$$

the origin of the tool frame ($O \rightarrow$ desired coordinates)

$$O = O_c^o + d_6 R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad R \text{ express the direction of } z_s$$

$$O_c^o = O - d_6 R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \text{Leave } O_c^o \text{ alone on left side of the eq}$$

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} o_x - d_6 r_{13} \\ o_y - d_6 r_{23} \\ o_z - d_6 r_{33} \end{bmatrix} \quad O = [o_x \ o_y \ o_z]^T, \ O_c^o = [x_c \ y_c \ z_c]^T$$

since $[x_c \ y_c \ z_c]^T$ are determined from the first three joint angles, our forward kinematics expression now allow us to solve for the first three joint angles decoupled from the final three

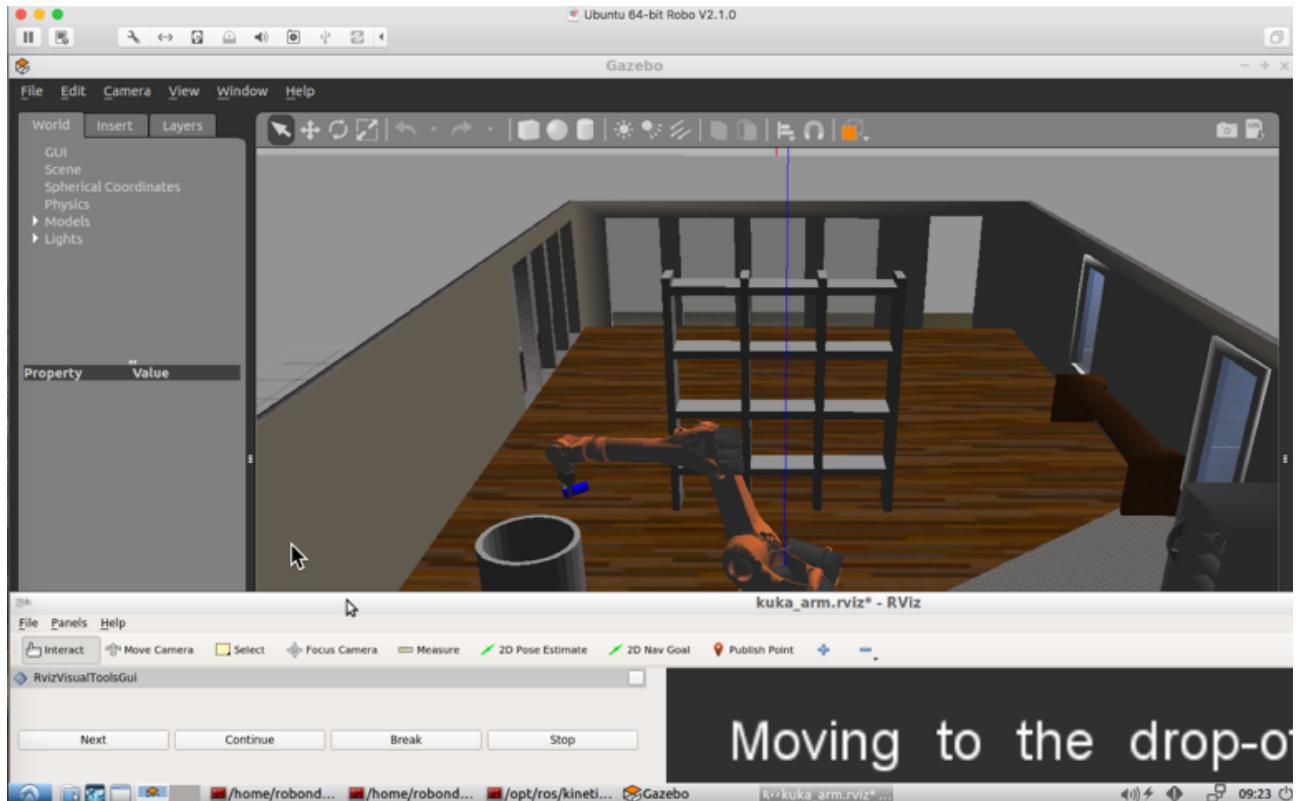
$$R = R_3^o R_6^s$$

To solve the final three joint angles.

$$K_6^3 = (R_3^o)^{-1} R = (R_3^o)^T R$$

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.

You can check source code for the detailed explanation. I have added detailed explanation to the source code.



Also, I have uploaded a video which robotic arm successfully carry object to the target location. You can find it in the following link.

<https://vimeo.com/259565752>

optimizations:

There might be a lot of pose in the trajectory. We evaluate all of them in a for loop, but forward kinematics could be build prior to for loop and only evaluation work inside the loop. Also, You can create a class structure and initialize it when IK_server is starting.

Another optimization technique, you can save transformation matrices and rotation matrices as pickle file and load them. Also, some people use numpy to make everything to get a better performance.

