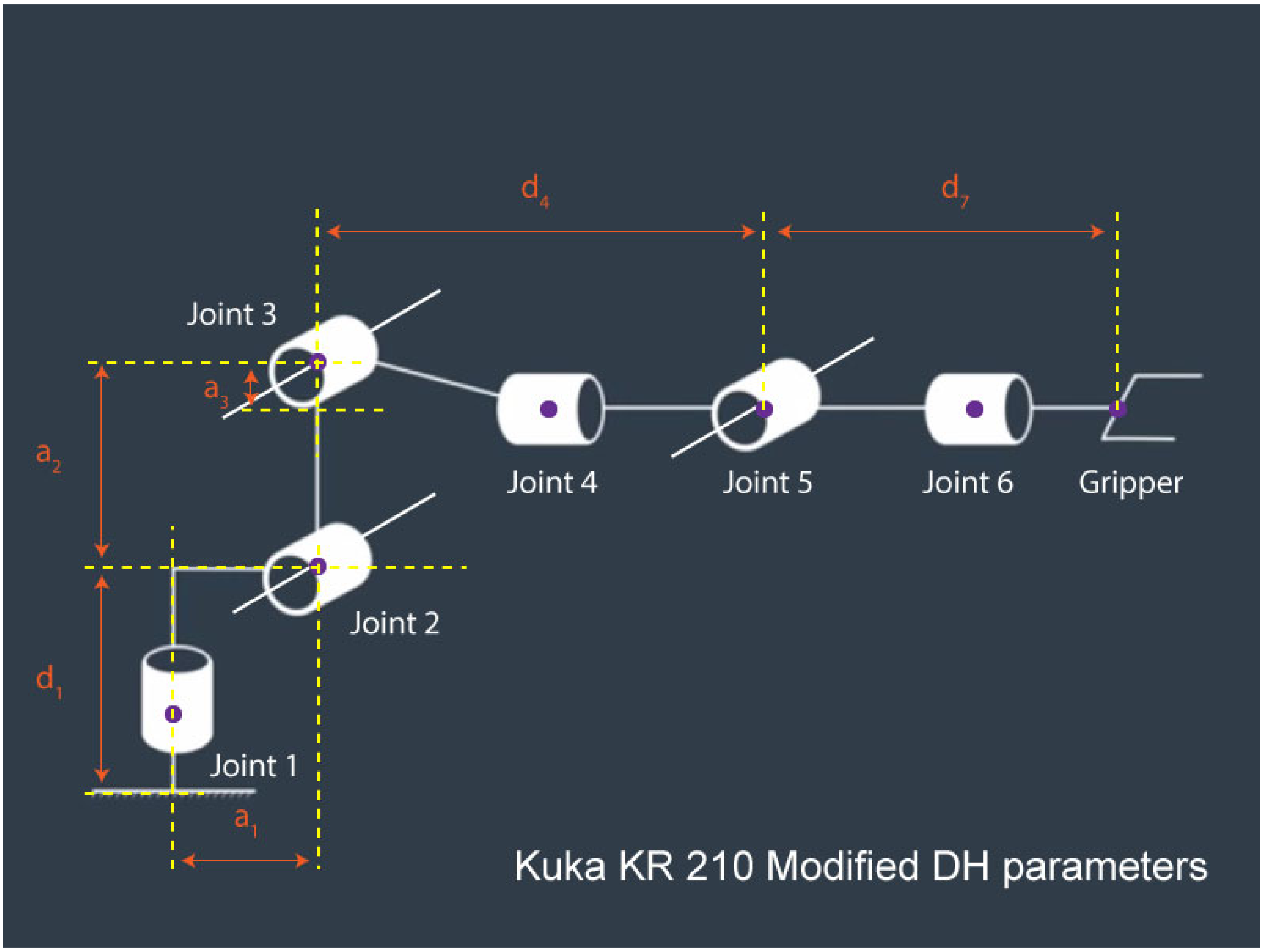
Project: Kinematics Pick & Place

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.



i	α_{i-1}	$\mathbf{a}_{i-1}$	$\mathbf{d}_i$	θ_i
1	0	0	0.75	
2	$-\frac{\pi}{2}$	0.35	0	$\theta_2 = \theta_2 - \frac{\pi}{2}$
3	0	1.25	0	
4	$-\frac{\pi}{2}$	-0.054	1.50	
5	$\frac{\pi}{2}$	0	0	
6	$-\frac{\pi}{2}$	0	0	
7	0	0	0.303	$\theta_7 = 0$

Above is the table containing the Modified 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.

$$T_1^0 = \begin{bmatrix} cos(\theta_1) & -sin(\theta_1) & 0 & 0 \\ sin(\theta_1) & cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0.75 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_2^1 = \begin{bmatrix} sin(\theta_2) & cos(\theta_2) & 0 & 0.35 \\ 0 & 0 & 1 & 0 \\ cos(\theta_2) & -sin(\theta_2) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^2 = \begin{bmatrix} cos(\theta_3) & -sin(\theta_3) & 0 & 1.25 \\ sin(\theta_3) & cos(\theta_3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_4^3 = \begin{bmatrix} cos(\theta_4) & -sin(\theta_4) & 0 & -0.054 \\ 0 & 0 & 1 & 1.50 \\ -sin(\theta_4) & -cos(\theta_4) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_5^4 = \begin{bmatrix} cos(\theta_5) & -sin(\theta_5) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ sin(\theta_5) & cos(\theta_5) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_6^5 = \begin{bmatrix} cos(\theta_6) & -sin(\theta_6) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -sin(\theta_6) & -cos(\theta_6) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_G^6 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.303 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

Given the target orientation $roll$, $pitch$ and yaw we have the following rotation matrices:

$$R_{roll} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(roll) & -sin(roll) \\ 0 & sin(roll) & cos(roll) \end{bmatrix}$$
$$R_{pitch} = \begin{bmatrix} cos(pitch) & 0 & sin(pitch) \\ 0 & 1 & 0 \\ -sin(pitch) & 0 & cos(pitch) \end{bmatrix}$$
$$R_{yaw} = \begin{bmatrix} cos(yaw) & -sin(yaw) & 0 \\ sin(yaw) & cos(yaw) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

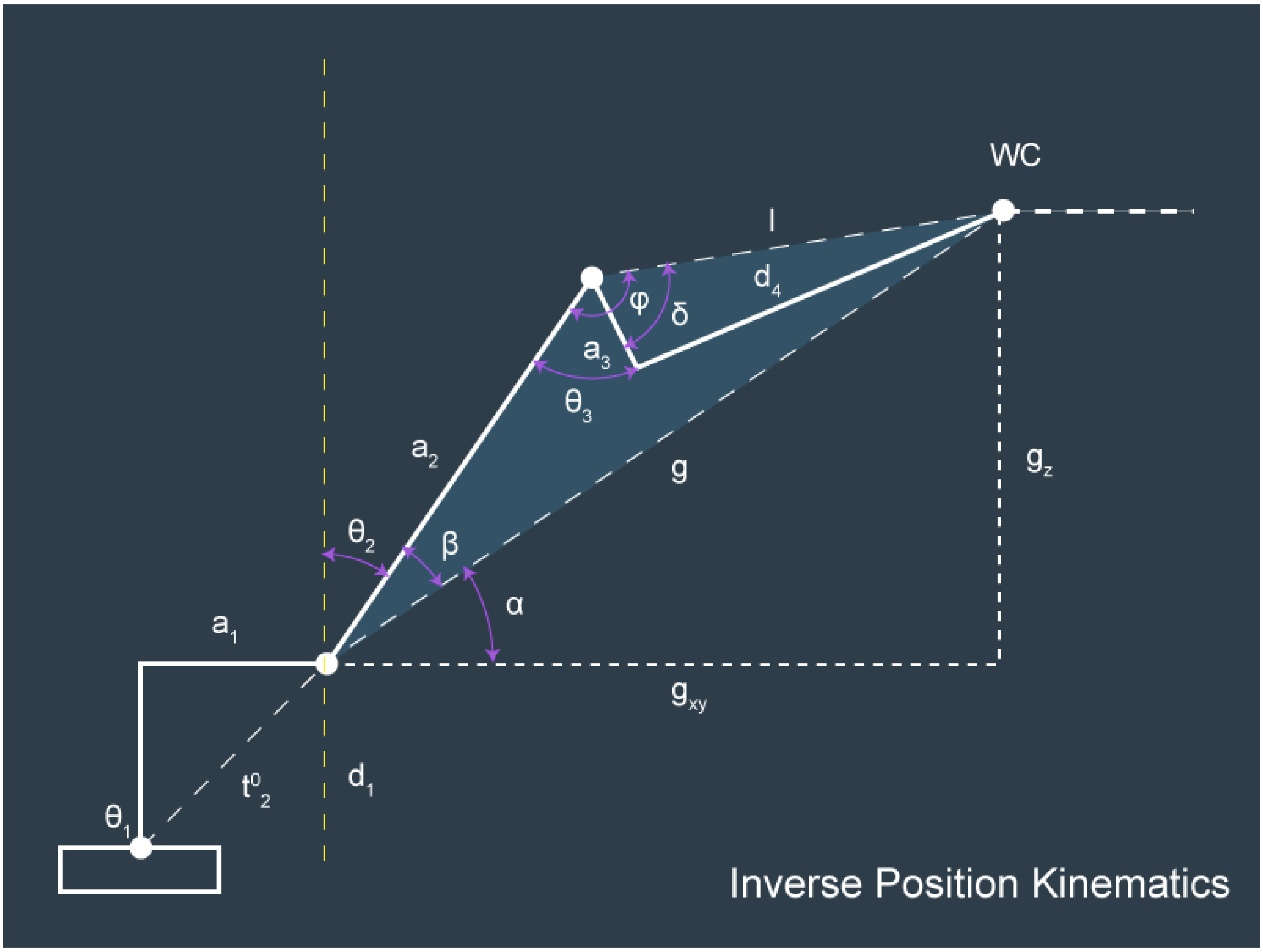
And thus:

$$R_6^0 = R_{yaw} \cdot R_{pitch} \cdot R_{roll} = \begin{bmatrix} l_x & m_x & n_x \\ l_y & m_y & n_y \\ l_z & m_z & n_z \end{bmatrix} \tag{2}$$

Given also the target position p_x, p_y and p_z , the complete target matrix can now be defined as follows:

$$T_G^0 = \left[\begin{array}{c|c} & p_x \\ R_6^0 & p_y \\ & p_z \\ \hline 0 & 1 \end{array} \right] = \begin{bmatrix} l_x & m_x & n_x & p_x \\ l_y & m_y & n_y & p_y \\ l_z & m_z & n_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

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



Inverse Position Kinematics

First we need to calculate the Wrist Center vector WC.

$$WC = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} - d_7 \cdot R_6^0 \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \implies \begin{cases} WC_x = p_x - (d_7 \cdot n_x) \\ WC_y = p_y - (d_7 \cdot n_y) \\ WC_z = p_z - (d_7 \cdot n_z) \end{cases} \tag{4}$$

Now that we have the WC, calculating θ_1 is fairly trivial.

$$\theta_1 = atan2(WC_y, WC_x) \tag{5}$$

Before calculating θ_2 and θ_3 we need to calculate the length of the sides of the triangle $[a_2, l, g]$:

$$l = \sqrt{a_3^2 + d_4^2} \tag{6}$$

$$g = WC - t_2^0 = WC - \begin{bmatrix} a_1 \cdot cos(\theta_1) \\ a_1 \cdot sin(\theta_1) \\ d_1 \end{bmatrix} = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix}$$

$$|g| = \sqrt{g_x^2 + g_y^2 + g_z^2}$$

Using the cosine rule...

$$A^2 = B^2 + C^2 - 2BCcos(\alpha)$$
$$\alpha = arccos\left(\frac{B^2 + C^2 - A^2}{2BC}\right)$$

... and the conversion into $atan2$ as follows ...

$$arccos(x) = atan2(\sqrt{1-x^2}, x)$$

We can now start finding θ_3:

$$\begin{aligned}\varphi &= atan2(d_4, a_3)\\ D &= \left(\frac{l^2 + a_2^2 - |g|^2}{2la_2}\right)\\ \delta &= atan2(\sqrt{1-D^2}, D)\\ \theta_3 &= \varphi - \delta\end{aligned} \tag{7}$$

And also θ_2:

$$\begin{aligned}g_{xy} &= \sqrt{g_x^2 + g_y^2}\\ \alpha &= atan2(g_z, g_{xy})\\ D &= \left(\frac{|g|^2 + a_2^2 - l^2}{2|g|a_2}\right)\\ \beta &= atan2(\sqrt{1-D^2}, D)\\ \theta_2 &= \frac{\pi}{2} - \alpha - \beta\end{aligned} \tag{8}$$

Inverse Orientation Kinematics

Given our target matrix R_6^0 and R_3^0 created using θ_1, θ_2 and θ_3 we can find R_6^3:

$$R_6^3 = (R_3^0)^T \cdot R_6^0 = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{9}$$

From the Forward Kinematics we can also find the following:

$$R_6^3 = \begin{bmatrix} -sin(\theta_4)*sin(\theta_6)+cos(\theta_4)*cos(\theta_5)*cos(\theta_6) & -sin(\theta_4)*cos(\theta_6)-sin(\theta_6)*cos(\theta_4)*cos(\theta_5) & sin(\theta_5)*cos(\theta_4) \\ sin(\theta_5)*cos(\theta_6) & -sin(\theta_5)*sin(\theta_6) & cos(\theta_5) \\ -sin(\theta_4)*cos(\theta_5)*cos(\theta_6)-sin(\theta_6)*cos(\theta_4) & sin(\theta_4)*sin(\theta_6)*cos(\theta_5)-cos(\theta_4)*cos(\theta_6) & sin(\theta_4)*sin(\theta_5) \end{bmatrix} \tag{10}$$

So, first we calculate θ_5:

$$\theta_5 = atan2(\sqrt{r_{13}^2 + r_{33}^2}, r_{23}) \tag{11}$$

This gives us 2 two possibilities:

$$
\text{if } sin(\theta_5) < 0 \begin{cases} \theta_4 = atan2(-r_{33}, r_{13}) \\ \theta_6 = atan2(r_{22}, -r_{21}) \end{cases}
$$
$$
\text{if } sin(\theta_5) \geq 0 \begin{cases} \theta_4 = atan2(r_{33}, -r_{13}) \\ \theta_6 = atan2(-r_{22}, r_{21}) \end{cases} \tag{12}
$$

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.

Target Rotation Matrix and DH Correction

The target rotation matrix `R0_6` as is described in eqs. (2) is implemented in lines 119-140.

In my code I also implemented a correction step for the target rotation matrix `R0_6`. This is done by first creating a Modified DH transformation matrix `R_corr` in lines 58-69. After that the target rotation matrix `R0_6` is multiplied with this matrix in line 147.

Inverse Position Kinematics

In the lines 150-193, I have implemented the inverse position kinematics.

First the code for computing `theta1` is implemented in lines 150-159 as described in equations (4) and (5) .

Then some helper variables which are described in equation (6) are calculated in lines 162-171. These will be used in the following two steps.

The `theta3` angle as described in equation (7) is computed in lines 174-180 and the `theta2` angle as described in equation (8) is computed in lines 185-191.

Inverse Orientation Kinematics

The angles `theta4` , `theta5` and `theta6` are computed in lines 196-207 as described in equations (11) and (12) .

Results

The robot arm successfully grasps the items on the shelf. The gripper is always oriented properly towards the items and is also correctly oriented when releasing the items into the bucket. When moving from start position to end position the gripper does like to rotate around a lot, but this is less of an issue given that the arm does not hit anything and it correctly grasps the items and correctly drops them. The downside of the excessive rotation of the gripper is that it takes a little more time than would otherwise be needed to carry out the movements.

Possibly the implementation could be improved with additional code making sure that the angles stay within the limits which are mentioned in the URDF description. In addition there might be some uncaught singularities or other mathematical quirks that have been overlooked, but in the tests I did, these did not show up.

References

Bringing a standard 6-Axis industry robot into FreeCAD for simulation. link

Kuka. KR 210-2 - KR 210 L180-2 - KR 210 L150-2 Technical Data. PDF

Milford Robotics. Theory videos for Introduction to Robotics ENB339. Youtube Playlist

Piotrowski, Norbert and Barylski, Adam. Modelling a 6-DOF manipulator using Matlab software. PDF

Rubric Points from The Udacity Robot Nano Degree program.

Slabaugh, Gregory. Computing Euler angles from a rotation matrix. PDF

Wikipedia. Inverse Trigonometric Functions. link

Wikipedia. Atan2. link