Project: Kinematics Pick & Place

Vladimir Kobal

April, 2018

1 Kinematic Analysis

1.1 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.

RViz TF Display (Figure 1) provides very clear and handy schematics for URDF coordinate frames of a robot and their relations. It helps to interpret values from a urdf.xacro file and to grasp a structure of a robot. Based on the information about joints from the urdf.xacro file and schematics from TF Display a kinematic diagram for Kuka KR210 robot (Figure 2) was constructed and DH parameters (Table 1) were obtained.

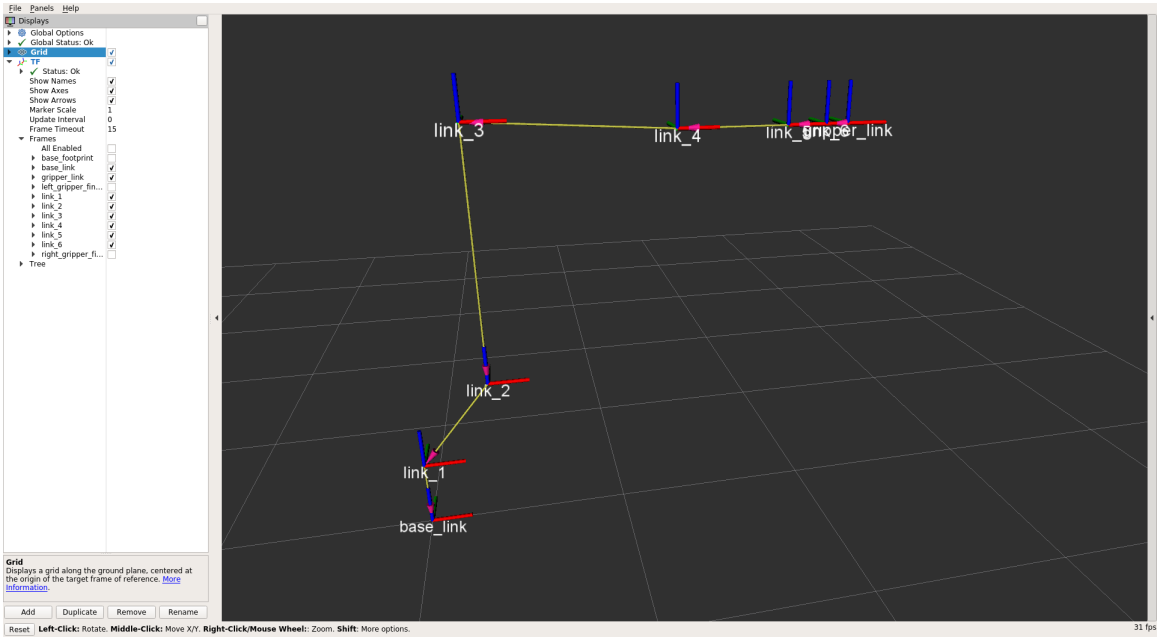

Figure 1: RViz TF Display

α_{i-1} are determined by rotations around x_i axes.

All a_{i-1} parameters correspond to the values from the urdf.xacro file.

d_1 is combined from offsets for base $\rightarrow$ joint 1 and joint 1 $\rightarrow$ joint 2: $d_1 = 0.33 + 0.42 = 0.75$.

d_4 is combined from offsets for joint 3 $\rightarrow$ joint 4 and joint 4 $\rightarrow$ joint 5: $d_4 = 0.96 + 0.54 = 1.5$.

d_G is combined from offsets for joint 5 $\rightarrow$ joint 6 and joint 6 $\rightarrow$ gripper: $d_G = 0.193 + 0.11 = 0.303$.

All θ_i rotations are variable except for link 6, where $\theta_G = 0$. θ_2 is a special case because it has a constant $-\pi/2$ offset.

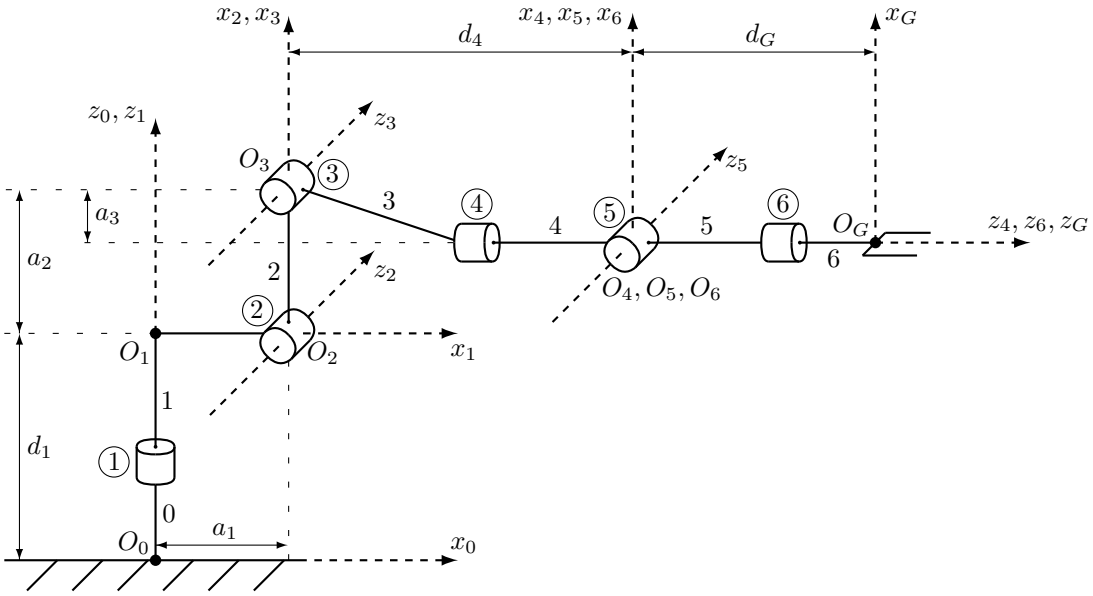

Figure 2: Kinematic diagram

Links	$\boldsymbol{\alpha_{i-1}}$	$\boldsymbol{a_{i-1}}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$
$0 \to 1$	0	0	0.75	θ_1
$1 \to 2$	$-\frac{\pi}{2}$	0.35	0	$-\frac{\pi}{2} + \theta_2$
$2 \to 3$	0	1.25	0	θ_3
$3 \to 4$	$-\frac{\pi}{2}$	-0.054	1.5	θ_4
$4 \to 5$	$\frac{\pi}{2}$	0	0	θ_5
$5 \to 6$	$-\frac{\pi}{2}$	0	0	θ_6
$6 \to G$	0	0	0.303	0

Table 1: DH Parameters

1.2 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.

Individual modified DH transformation is

$$ {}^{i-1}_{i}T = R_{x_{i-1}}(\alpha_{i-1})\ D_{x_{i-1}}(a_{i-1})\ R_{z_i}(\theta_i)\ D_{z_i}(d_i) \tag{1} $$

$$ {}^{i-1}_{i}T = \begin{bmatrix} cos(\theta_i) & -sin(\theta_i) & 0 & a_{i-1} \\ sin(\theta_i)cos(\alpha_{i-1}) & cos(\theta_i)cos(\alpha_{i-1}) & -sin(\alpha_{i-1}) & -sin(\alpha_{i-1})d_i \\ sin(\theta_i)sin(\alpha_{i-1}) & cos(\theta_i)sin(\alpha_{i-1}) & cos(\alpha_{i-1}) & cos(\alpha_{i-1})d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

If we substitute DH parameters we get seven individual transformation matrices for every joint (Equations 3 - 9).

$$ {}^{0}_{1}T = \begin{bmatrix} cos(\theta_1) & -sin(\theta_1) & 0 & 0 \\ sin(\theta_1) & cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0.75 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3} $$

$$ {}^{1}_{2}T = \begin{bmatrix} sin(\theta_2) & cos(\theta_2) & 0 & 0.35 \\ 0 & 0 & 1 & 0 \\ cos(\theta_2) & -sin(\theta_2) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4} $$

$$ {}^{2}_{3}T = \begin{bmatrix} cos(\theta_3) & -sin(\theta_3) & 0 & 1.25 \\ sin(\theta_3) & cos(\theta_3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5} $$

$$ {}^{3}_{4}T = \begin{bmatrix} cos(\theta_4) & -sin(\theta_4) & 0 & -0.054 \\ 0 & 0 & 1 & 1.5 \\ -sin(\theta_4) & -cos(\theta_4) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6} $$

$$ {}^{4}_{5}T = \begin{bmatrix} cos(\theta_5) & -sin(\theta_5) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ sin(\theta_5) & cos(\theta_5) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7} $$

$$ {}^{5}_{6}T = \begin{bmatrix} cos(\theta_6) & -sin(\theta_6) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -sin(\theta_6) & -cos(\theta_6) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8} $$

$$ {}^{6}_{G}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.303 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9} $$

The total transformation from the base to the gripper is ${}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T\,{}^{6}_{G}T = {}^{0}_{G}T$. Orientation of DH and URDF gripper frames is different (Figure 3). We want to get the total transformation to URDF coordinates so we have to apply additional rotation $Rcorr = R_z(\pi)R_y(-\pi/2)$, where rotations about axes are defined by matrices 10, 11, and 12 (R_z provided for future reference).

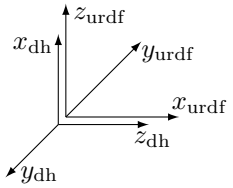

Figure 3: Gripper frames

$$ R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) & -sin(\alpha) \\ 0 & sin(\alpha) & cos(\alpha) \end{bmatrix} \tag{10} $$

$$ R_y = \begin{bmatrix} cos(\alpha) & 0 & sin(\alpha) \\ 0 & 1 & 0 \\ -sin(\alpha) & 0 & cos(\alpha) \end{bmatrix} \tag{11} $$

$$ R_z = \begin{bmatrix} cos(\alpha) & -sin(\alpha) & 0 \\ sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{12} $$

$$ Tcorr = \begin{bmatrix} Rcorr & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{13} $$

${}^{0}_{G}TTcorr$ will provide us with the total transformation to a gripper position in URDF coordinates. We can use this transformation directly for solving Forward Kinematics problem, when $\theta_1, \theta_2, ..., \theta_6$ are known. For solving Inverse Kinematics problem, when we know a gripper pose, but not $\theta_1, \theta_2, ..., \theta_6$, we have to define the total transformation using the gripper pose which is defined by p_x, p_y, p_z, $roll$, $pitch$ and yaw.

We are rolling the gripper about x axis in URDF coordinates, so rotation matrix for obtaining a gripper orientation $R_{rpy} = R_z(yaw)R_y(pitch)R_x(roll)$. For the sake of brevity, let $r = roll$, $p = pitch$, $y = yaw$. Translation part of the transformation consists from position coordinates of the gripper, thus the total transformation in terms of URDF gripper coordinates is

$$ {}_{G_{\text{urdf}}}{}^{0}T = \begin{bmatrix} & & & p_x \\ & R_{rpy} & & p_y \\ & & & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \tag{14} $$

$$ = \begin{bmatrix} cos(p)cos(y) & sin(p)sin(r)cos(y) - sin(y)cos(r) & sin(p)cos(r)cos(y) + sin(r)sin(y) & p_x \\ sin(y)cos(p) & sin(p)sin(r)sin(y) + cos(r)cos(y) & sin(p)sin(y)cos(r) - sin(r)cos(y) & p_y \\ -sin(p) & sin(r)cos(p) & cos(p)cos(r) & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{15} $$

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.

To calculate θ_1, θ_2 and θ_3 we can use a position of the wrist center(WC) which is calculated as an offset d_G along x_{urdf} axis

$$ \begin{bmatrix} WC_x \\ WC_y \\ WC_z \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} - d_G R_{rpy} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{16} $$

We can derive the equations to calculate θ_1, θ_2 and θ_3 using Figures 4 and 5. O_0, O_1, O_2 and WC are on a plain. $z' || z_1$ and x' is on the x_1 axis.

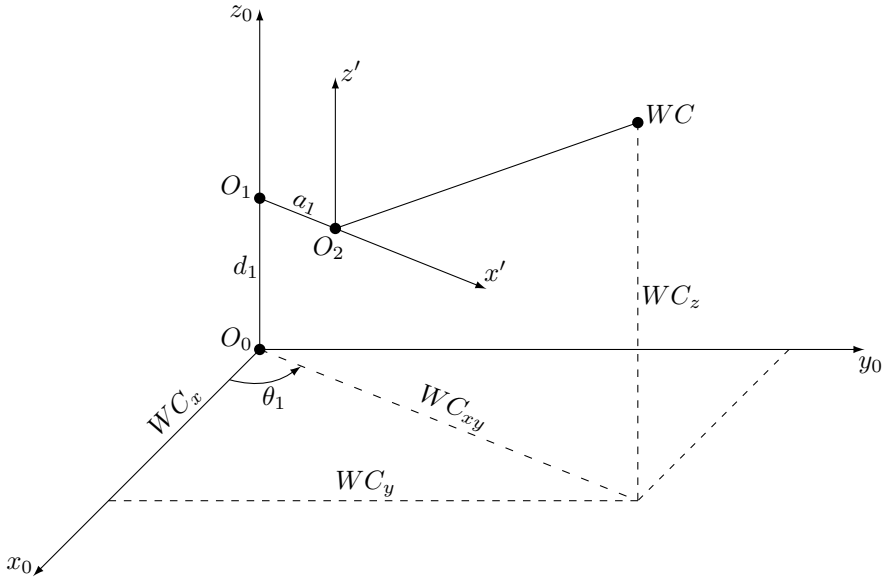

Figure 4: Calculation of θ_1

From Figure 4 it is clear that $\theta_1 = atan2(WC_y, WC_x)$.

$$ WC_{xy} = \sqrt{WC_x^2 + WC_y^2} \tag{17} $$

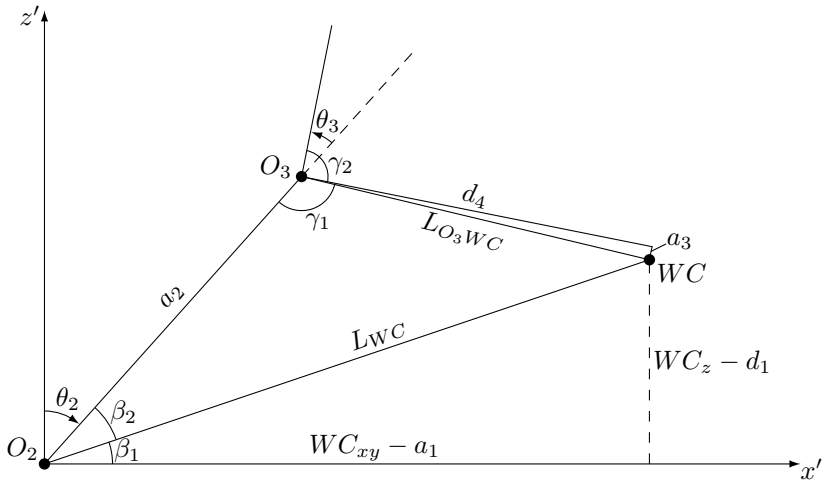

Figure 5: Calculation of θ_2, θ_3

$$L_{WC} = \sqrt{(WC_{xy} - a_1)^2 + (WC_z - d_1)^2} \tag{18}$$

$$L_{O_3WC} = \sqrt{d_4^2 + a_3^2} \tag{19}$$

Now that we have calculated all the lengths it is possible to proceed to the angles.

$$\beta_1 = atan2((WC_z - d_1), (WC_{xy} - a_1)) \tag{20}$$

$$\beta_2 = acos(\frac{a_2^2 + L_{WC}^2 - L_{O_3WC}^2}{2 \cdot a_2 \cdot L_{WC}}) \tag{21}$$

$$\theta_2 = \pi/2 - (\beta + \beta_2) \tag{22}$$

$$\gamma_1 = acos(\frac{a_2^2 + L_{O_3WC}^2 - L_{WC}^2}{2 \cdot a2 \cdot L_{O_3WC}}) \tag{23}$$

$$\gamma_2 = \pi/2 - atan2(a_3, d_4) \tag{24}$$

$$\theta_3 = \pi - (\gamma_1 + \gamma_2) \tag{25}$$

It is time to solve Inverse Orientation Kinematics problem and calculate θ_4, θ_5 and θ_6. We can start with equation ${}^0_G TTcorr = {}^0_{G_{\text{urdf}}}T$. A wrist center position is explicitly defined by θ_1, θ_2 and θ_3. Joints 4, 5 and 6 don't translate the wrist center, they only rotate it. We can decompose homogeneous transformation and concentrate on the rotational part ${}^0_G RRcorr = {}^0_{G_{\text{urdf}}}R$. As was mentioned before the transformation between the wrist center and the gripper is just a transfer without rotation that's why a rotational transformation for the wrist center is the same as for the gripper.

$${}^0_3R\,{}^3_6R\,{}^6_G RRcorr = R_{rpy} \tag{26}$$

$${}^6_G R = I \tag{27}$$

$${}^0_3R^{-1}\,{}^0_3R\,{}^3_6RIRcorrRcorr^{-1} = {}^0_3R^{-1}R_{rpy}Rcorr^{-1} \tag{28}$$

$${}^3_6R = {}^0_3R^{-1}R_{rpy}Rcorr^{-1} \tag{29}$$

The right side of equation is known. Left side is a rotational transformation

$$ {}^{3}_{6}R = R_x R_y R_x = \tag{30} $$

$$ \begin{bmatrix} -sin(\theta_4)sin(\theta_6) + cos(\theta_4)cos(\theta_5)cos(\theta_6) & -sin(\theta_4)cos(\theta_6) - sin(\theta_6)cos(\theta_4)cos(\theta_5) & -sin(\theta_5)cos(\theta_4) \\ sin(\theta_5)cos(\theta_6) & -sin(\theta_5)sin(\theta_6) & cos(\theta_5) \\ -sin(\theta_4)cos(\theta_5)cos(\theta_6) - sin(\theta_6)cos(\theta_4) & sin(\theta_4)sin(\theta_6)cos(\theta_5) - cos(\theta_4)cos(\theta_6) & sin(\theta_4)sin(\theta_5) \end{bmatrix} \tag{31} $$

Let us denote the known rotation as

$$ {}^{0}_{3}R^{-1} R_{rpy} Rcorr^{-1} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{32} $$

Then $r_{21} = sin(\theta_5)cos(\theta_6)$, $r_{22} = -sin(\theta_5)sin(\theta_6)$, $r_{23} = cos(\theta_5)$, $r_{13} = -sin(\theta_5)cos(\theta_4)$, $r_{33} = sin(\theta_4)sin(\theta_5)$

$$ \theta_4 = atan2(r_{33}, -r_{13}) \tag{33} $$

$$ \theta_5 = acos(r_{23}) \tag{34} $$

$$ \theta_6 = atan2(-r_{22}, r_{21}) \tag{35} $$

$atan2$ gives us the range $(-180°, 180°]$, but joints 4 and 6 have the range $(-350°, 350°)$. So we have to use some additional logic for these angles. We also have to make a decision about the sign of θ_5. We will use the shortest angular distance for a joint rotation to solve these problems.

2 Project Implementation

2.1 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.

In this program we are using symbolic computation. So symbols are created at first for DH parameters and Euler angles (line #31). DH parameters table (line #38) is filled with constant values from Table 1. All those matrices from Section 1.2 which are needed for Inverse Kinematics are prepared before the service starts to listen for requests (lines #47-88). Matrices are also simplified in order to reduce request processing time and improve calculation precision.

In the main cycle θ_i are calculated based on formulas from Section 1.3 (lines #152-202). θ_4 and θ_6 are special ones because joints 4 and 5 have extended rotation limits. In order to minimize frequent wrist turning over, previous θ_4, θ_5 and θ_6 values are used to calculate and choose the shortest angular distance for every joint for reaching a new pose (line #188). Calculated θ_4, θ_5 and θ_6 are preserved for the next cycle (line #203). For the first cycle current angle values for joints 4, 5 and 6 are requested from gazebo (line #119).

For the end effector position error evaluation Forward Kinematics matrices are calculated before the service starts (line #92). Error evaluation code is disabled by default (lines #24,113,209,238). Calculated error values are stored into a .csv file (line #208). Errors for 653 positions (24 paths) where stored (Figure 6). Based on that sample we can conclude that the errors are quite small (less than $1.7 \cdot 10^{-14}$ and basically tend to be near 10^{-15}) and aren't critical for the robot operations.

The robot successfully completes 10/10 pick and place cycles and tracks the planned trajectory. Example of pick and place process is depicted in Figure 7.

There are still a lot of potential failures with the code because received data are not verified for correctness, nor θ_1, θ_2, θ_3 and θ_5 are verified for the limits. Joint 1 also has a slightly extended range of movement which needs a solution like we used for θ_4 and θ_6. The program might be better structured. We don't need so many global variables, the majority of them could be hidden inside functions or classes. For the sake of performance much faster methods could be used instead of .subs, or even hardcoded matrices without symbols at all. So there are still a lot of space for improving the code.

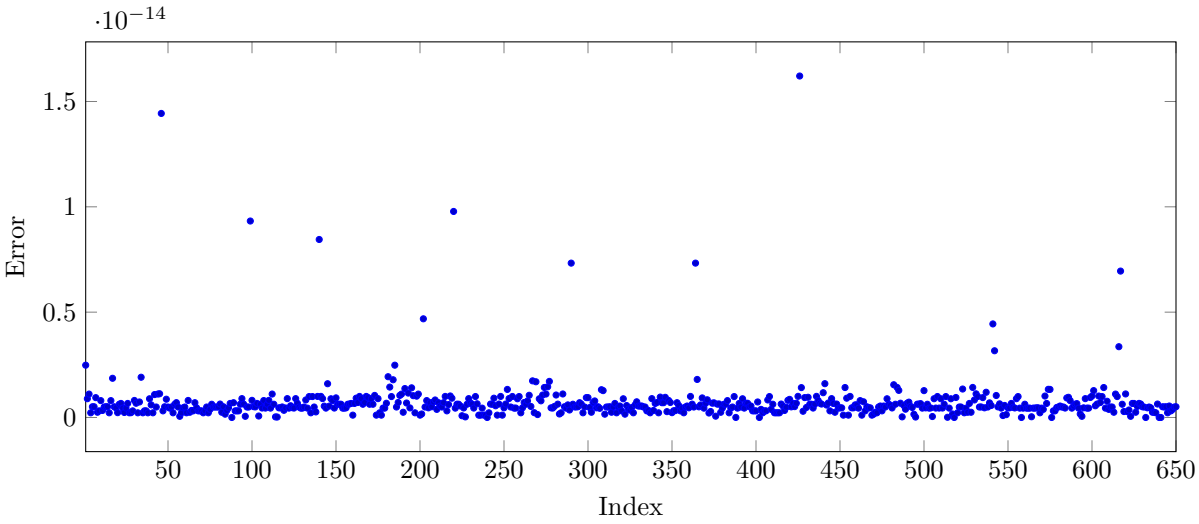

Figure 6: Error evaluation

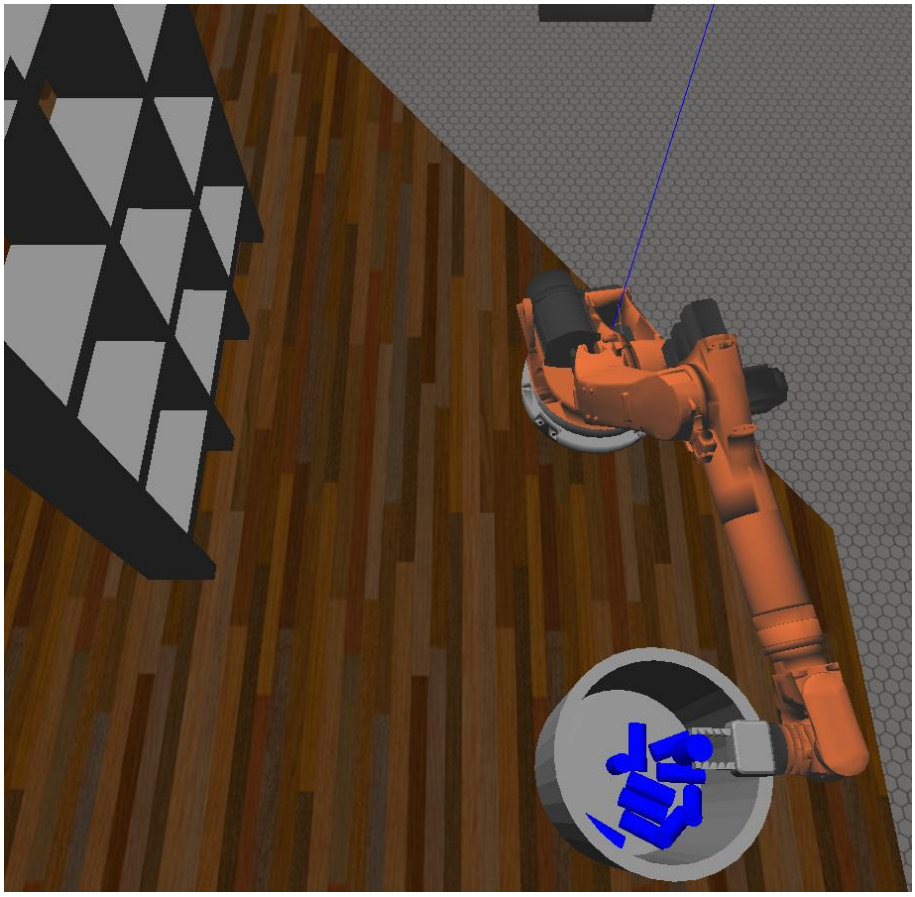

Figure 7: Completed pick and place process