Robotics

Academic Year 2021/2022

1st lab assignment- Introduction to Serial Manipulators

Gaia Morillo 1101499 Marco Mussato 1101412 João Gonçalves 85211

Table of Contents

1. Introduction

The purpose of this project is to simulate a solution for a tiling problem using the Niryo One robot. The task consists in moving 4 blocks from an initial position (stacked on top of each other) to a final one in a fixed layout, as represented in Figure 1. Niryo One has six revolute joints and since it is a rigid body in three dimensions, it means that it is characterized by 6 Degrees of Freedom (DoF), three translational and three rotational. To solve this problem, it was necessary to focus in two areas:

Figure 1 – Problem layout

1. Direct Kinematics and joint uncertainties: the study of the direct kinematics allowed us to define the configuration of the robot in space, finding out the position and orientation of the end-effector. Using the direct kinematics it was possible to add normally distributed noise to the joints in the simulation, analysing how they affect the robot. A solution was developed showing the precision of the robot in simulation with added uncertainties.
2. Strategies to enhance precision: due to the mistakes caused by the uncertainties in the movement of the robot, it is necessary to find ways to overcome this issue. Using the direct kinematics and data obtained from the physical robot in the laboratory, another solution was computed for the problem, specifically for that robot.

2. Direct kinematics

The goal of the direct kinematics problem is to compute the configuration of the end effector (position and orientation), given the value of the angle of each joint. The final solution is obtained through a series of multiplication between transformations matrices.

In order to compute those matrices, the initial step is to define a set of frames that are capable of correctly defining the motion of the robot. Different conventions can be used to reach the same outcome. In this project it was chosen to use the convention shown in Figure 2.

In each frame the *x axis* is pointing left, the *y axis* is pointing towards the observer and the *z axis* is pointing upwards.

Once the convention is chosen, it's mandatory to know where the rotation happens in each joint and which is the change in position between every frame. Please note that each distance in the translation vector is in mm and each value can be found in Figure 2.

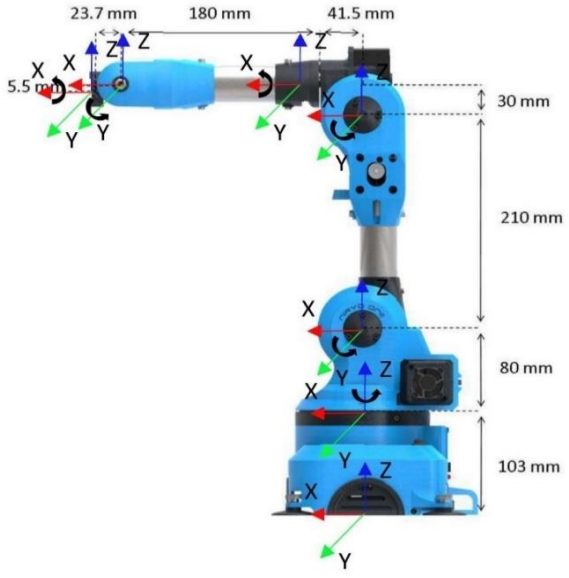

Figure 2 – Robot geometry and frames

For the first pair of frames, from world frame (frame 0) to frame 1 (the frame for the first link), since the rotation happens only along the *z axis*, the rotation and translation matrices are:

$$ {}^{0}_{1}P = \begin{bmatrix} 0 \\ 0 \\ 103 \end{bmatrix} \quad {}^{0}_{1}R_z = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

This leads to the following transformation matrix:

$$ {}^{0}_{1}T = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 103 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Moving to the transformation from frame 1 to frame 2, since the rotation happens only along the *y axis*, the rotation and translation matrices are:

$$ {}^{1}_{2}P = \begin{bmatrix} 0 \\ 0 \\ 80 \end{bmatrix} \quad {}^{1}_{2}R_y = \begin{bmatrix} \cos(\theta_2) & 0 & \sin(\theta_2) \\ 0 & 1 & 0 \\ -\sin(\theta_2) & 0 & \cos(\theta_2) \end{bmatrix} $$

This leads to the following transformation matrix:

$$ {}^{1}_{2}T = \begin{bmatrix} \cos(\theta_2) & 0 & \sin(\theta_2) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_2) & 0 & \cos(\theta_2) & 80 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

For the next pair, from frame 2 to frame 3, the rotation happens, again, only along the *y axis*, hence the rotation and translation matrices:

$$ {}^{2}_{3}P = \begin{bmatrix} 0 \\ 0 \\ 210 \end{bmatrix} \quad {}^{2}_{3}R_y = \begin{bmatrix} \cos(\theta_3) & 0 & \sin(\theta_3) \\ 0 & 1 & 0 \\ -\sin(\theta_3) & 0 & \cos(\theta_3) \end{bmatrix} $$

This leads to the following transformation matrix:

$$ {}^{2}_{3}T = \begin{bmatrix} \cos(\theta_3) & 0 & \sin(\theta_3) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_3) & 0 & \cos(\theta_3) & 210 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Then, for the next pair, from frame 3 to frame 4, the rotation happens only along the *x axis*, this

leads to the following rotation and translation matrices:

$$ {}^{3}_{4}P = \begin{bmatrix} 41.5 \\ 0 \\ 30 \end{bmatrix} \; {}^{3}_{4}R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_4) & -\sin(\theta_4) \\ 0 & \sin(\theta_4) & \cos(\theta_4) \end{bmatrix} $$

This leads to the following transformation matrix:

$$ {}^{3}_{4}T = \begin{bmatrix} 1 & 0 & 0 & 41.5 \\ 0 & \cos(\theta_4) & -\sin(\theta_4) & 0 \\ 0 & \sin(\theta_4) & \cos(\theta_4) & 30 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Then, for the penultimate pair of frames, from frame 4 to frame 5, the rotation happens, again, only along the *y axis.* For this reason, the rotation and translation matrices are:

$$ {}^{4}_{5}P = \begin{bmatrix} 180 \\ 0 \\ 0 \end{bmatrix} \; {}^{4}_{5}R_y = \begin{bmatrix} \cos(\theta_5) & 0 & \sin(\theta_5) \\ 0 & 1 & 0 \\ -\sin(\theta_5) & 0 & \cos(\theta_5) \end{bmatrix} $$

This leads to the following transformation matrix:

$$ {}^{4}_{5}T = \begin{bmatrix} \cos(\theta_5) & 0 & \sin(\theta_5) & 180 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_5) & 0 & \cos(\theta_5) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

For the last pair of frames, from frame 5 to frame 6, since the rotation still happens just along the *x axis*, the last rotation and translation matrices are:

$$ {}^{5}_{6}P = \begin{bmatrix} 23.7 \\ 0 \\ -5.5 \end{bmatrix} \; {}^{5}_{6}R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_6) & -\sin(\theta_6) \\ 0 & \sin(\theta_6) & \cos(\theta_6) \end{bmatrix} $$

This leads to the following transformation matrix:

$$ {}^{5}_{6}T = \begin{bmatrix} 1 & 0 & 0 & 23.7 \\ 0 & \cos(\theta_6) & -\sin(\theta_6) & 0 \\ 0 & \sin(\theta_6) & \cos(\theta_6) & -5.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Once that all the transformation matrices are obtained it's possible to compute the transformation matrix that connects frame 0 (world frame) to frame 6 (end effector's frame):

$$ {}^{0}_{6}T = {}^{0}_{1}T * {}^{1}_{2}T * {}^{2}_{3}T * {}^{3}_{4}T * {}^{3}_{4}T * {}^{5}_{6}T = \begin{bmatrix} R_{3x3} & T_{3x1} \\ 0^T & 1 \end{bmatrix} $$

This matrix is needed to solve the direct kinematics problem and it can be done as follows:

- Position: The end effector's position, in respect to the world frame, is represent by the first three elements of the last column of the ${}^{0}_{6}T$ matrix.
- Orientation: Since a generic orientation in the space can be obtained in multiple ways, to get the orientation of the end effector a convention should be chosen. In this case, the convention is the *Z-Y-Z convention* using Euler angles. Hence, it's possible to do the following assumption:

$$ \begin{bmatrix} c_\alpha c_\beta c_\gamma - s_\alpha s_\gamma & -c_\alpha c_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta \\ s_\beta c_\beta c_\gamma + c_\alpha s_\gamma & -s_\alpha c_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta \\ -s_\beta c_\gamma & s_\beta s_\gamma & c_\beta \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} $$

Where the matrix on the left side represents the first three rows and columns of the ${}^{0}_{6}T$ matrix.

The following formulas describe one way to compute the angles.

$$ \begin{cases} \beta = arctan2(\sqrt{r_{13}^2 + r_{23}^2}, r_{33}) \\ \gamma = arctan2\left(\frac{r_{32}}{s_\beta}, \frac{-r_{31}}{s_\beta}\right), if\ s_\beta \neq 0 \\ \alpha = arctan2\left(\frac{r_{23}}{s_\beta}, \frac{r_{13}}{s_\beta}\right), if\ s_\beta \neq 0 \end{cases} $$

Note that, the first angle that should be computed is β, since is needed to compute all the others. Problems can occur when $\sin\beta = 0$, since the previous formulas can't be used anymore. So, if $\sin\beta = 0$ then $\cos\beta = \pm 1$ and this leads to $\alpha \pm \gamma = c_\beta \cdot atan2(\pm r_{21}, r_{22})$. Assuming $\gamma = 0$ (a different choice can be made) α can be computed as shown below:

$$ \begin{cases} \gamma = 0 \\ \alpha = c_\beta \cdot atan2(r_{21}, r_{11}) \end{cases} $$

3. Joint error

Now the aim was to study how the uncertainties propagate through the direct kinematics. The real robot will exhibit systematic errors, that in simulation will a consist in proportional differences between the observed and true values of the degrees in each joint. This error can be caused, for instance, by a miscalibration or loosening of mechanical couplings. For this analysis just the error on the positioning (and not on the orientation) has been considered.

To understand how the error in the joint's angle effects the final position, the following formula has been used:

$$dX = Jd\theta_i$$

Where dX is the general error on the final position, $d\theta_i$ are the angle errors of 6 axis (*i*=1, 2, 3,…,6) and J is the Jacobian matrix. The final Jacobian matrix $J_{6\times6}$, can be computed as follows:

$$J_{6\times6} = [J_1\, J_2\, J_3\, J_4\, J_5\, J_6]$$

where J_i is the Jacobian matrix for the i-th joint. A generic Jacobian matrix, for a revolute joint, can be calculated as follows:

$$J = \begin{bmatrix} z_{i-1} \times (t_n - t_{i-1}) \\ z_{i-1} \end{bmatrix}$$

where, z_{i-1} is the value of the top 3 elements in column of the rotation axis of each joint of the matrix T_0^i . The chosen column is different for each joint since we are using a different convention compared to the Denavit-Hartenberg convention. Then, t_n is the value of the top 3 elements in column 4 of the matrix T_0^6 and t_{i-1} is the value of the top 3 elements in column 4 of the matrix T_0^i. Since in our work we used the *RViz* simulator, characterized by the absence of error, some error was added to each joint. The error was modelled with a gaussian distribution to simulate uncertainties in the angular position.

Therefore, the angular error on each joint can be calculated as follows:

$$d\theta_i = \frac{1}{g}\sum_{k=1}^{g}(\theta_{in} - \theta_{ia})$$

θ_{in} are the nominal angles, θ_{ia} are the actual angles, meaning the nominal angle plus a random uncertainties, g is number of measurement points and *i* represents each of the robotic joints.

The position errors in the *x*, *y* and *z* axis at the end-effector are respectively:

$$dx = J_{11}d\theta_1 + J_{12}d\theta_2 + J_{13}d\theta_3 + J_{14}d\theta_4 + J_{15}d\theta_5 + J_{16}d\theta_6$$

$$dy = J_{21}d\theta_1 + J_{22}d\theta_2 + J_{23}d\theta_3 + J_{24}d\theta_4 + J_{25}d\theta_5 + J_{26}d\theta_6$$

$$dz = J_{31}d\theta_1 + J_{32}d\theta_2 + J_{33}d\theta_3 + J_{34}d\theta_4 + J_{35}d\theta_5 + J_{36}d\theta_6$$

Using these values total error on the end-effector is the following:

$$e_t = \sqrt{d_x^2 + d_y^2 + d_z^2}$$

Since the Jacobian matrix is strictly related to the configuration of the robot, this analysis was performed on two specific configurations, that are the most important for the task: the starting position of the blocks and the final one.

The results of the computation, considering a σ of value 0.01 for the normal distribution, are represented in the tables 1, 2, 3 and 4.

Each value of $d\theta_i$ in each test is the mean of 200 equispaced measurements between the maximum and minimum value of θ in each joint.

As a result, the joint that cause more error in the position of the end effector is the joint 3. However, this result may have been polarized by the experimental setup and further statistical analysis could be done to validate the result.

Stack	$d\theta_1$	$d\theta_2$	$d\theta_3$	$d\theta_4$	$d\theta_5$	$d\theta_6$
Test 1	1.3E-03	1.4E-05	7.5E-04	-8.8E-05	-2.5E-04	-2.2E-04
Test 2	-7.5E-04	7.2E-04	1.5E-03	8.0E-04	-1.1E-03	-1.2E-03
Test 3	2.2E-04	-1.5E-04	3.9E-04	-4.4E-04	7.2E-04	-4.6E-04
Mean	2.5E-04	1.9E-04	8.7E-04	9.1E-05	-2.1E-04	-6.3E-04

Table 1 – Joint error for the stack positions

Final	$d\theta_1$	$d\theta_2$	$d\theta_3$	$d\theta_4$	$d\theta_5$	$d\theta_6$
Test 1	8.4E-06	2.4E-04	-6.7E-04	2.1E-04	-3.5E-04	2.7E-04
Test 2	-6.5E-04	-6.2E-04	1.4E-03	8.4E-04	-2.0E-04	8.4E-04
Test 3	2.8E-04	-1.3E-03	6.8E-04	-7.4E-04	-4.4E-04	8.5E-05
Mean	-1.2E-04	-5.6E-04	4.7E-04	1.1E-04	-3.3E-04	4.0E-04

Table 2 – Joint error for final positions

Stack	dx	dy	dz	e_t
Test 1	0.140	-0.255	0.013	0.291
Test 2	0.402	-0.674	-0.049	0.787
Test 3	0.024	-0.044	0.041	0.065
Mean	0.189	-0.324	0.002	0.381

Table 3 – Positioning error for stack positions

Final	dx	dy	dz	e_t
Test 1	-0.184	-0.077	-0.107	0.226
Test 2	0.679	0.377	0.195	0.800
Test 3	0.045	0.170	0.113	0.209
Mean	0.180	0.157	0.067	0.412

Table 4 – Positioning error for final positions

With the help of the simulator it was also possible to determinate the positioning error of the end-effector. The average error of those measurements can be found in table 5. These results were also obtained with a sigma of 0.01 value. This results in a variation of each joint angle of up to around 0.5 degrees. Although this number doesn´t seem very high, given the fact we are dealing with a robot with millimetric precision, these variations are quite significant. Still, as seen in table 5, the error is minimal, in most cases close to 1 %, so the effect of these uncertainties does not have a major impact in the final position of the end effector. The orientation of the end-effector was also taken into consideration. Using the alpha-beta-gamma convention, it was expected that the values would be around alpha = 0, beta = 90 and gamma = 0 degrees, for all the positions. The average value of these variables for all the positions was of alpha = 2.75, beta = 87.6 and gamma = 1,87 degrees. We can conclude that the end-effector had an acceptable orientation.

	dx (%)	dy (%)	dz (%)
Stack 1	1.384	0.812	0.764
Stack 2	0.378	0.112	0.673
Stack 3	1.537	1.39	1.121
Stack 4	0.412	2.168	4.032
Final 1	0.267	1.053	2.325
Final 2	0.585	3.113	2.296
Final 3	1.141	3.545	0.833
Final 4	1.045	3.618	4.090

Table 5 – Positioning errors from the simulator

4. Strategies

Now that the small uncertainties have been tackled in the simulation, it is time to address how to deal with the errors that would occur when using the robot in the real world.

Firstly, in order to complete the task successfully, it is mandatory to have a good strategy for the movement. The end-effector must then approach the stack of blocks from a vertical position (right above the stack), in order to keep it balanced.

Other than that, after picking up the block, the robot must return to the position above the stack instead of aiming straight to the final position. This prevents the other blocks below (being in touch with the block we are picking up) to follow the movement causing the pile to fall.

The way the robot places the blocks in the target is also very important. A position above the target for the four blocks was chosen. When reached, a small adjustment is done for each one of the cases, and the robot then performs a vertical movement down towards the final position followed by the inverse movement to de-attach, similar to what happens when picking the blocks from the stack.

The order in which the blocks are placed in the target is not random as well. Because the robot is moving vertically, we want to avoid hitting other blocks already in place, as shown in Figure 3.

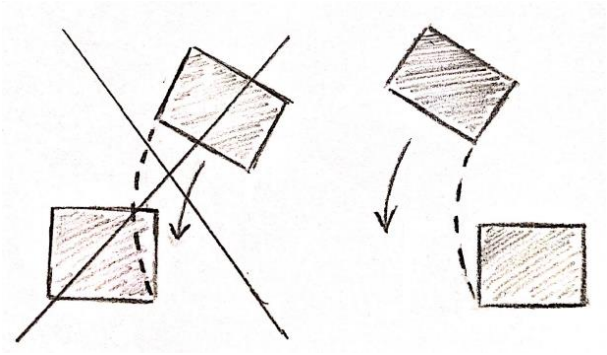

Figure 3 – Placement strategy

Therefore the configuration, shown in Figure 4 was selected. Given that the blocks are slightly smaller than the target area, a margin of around 1mm was also given to prevent collisions.

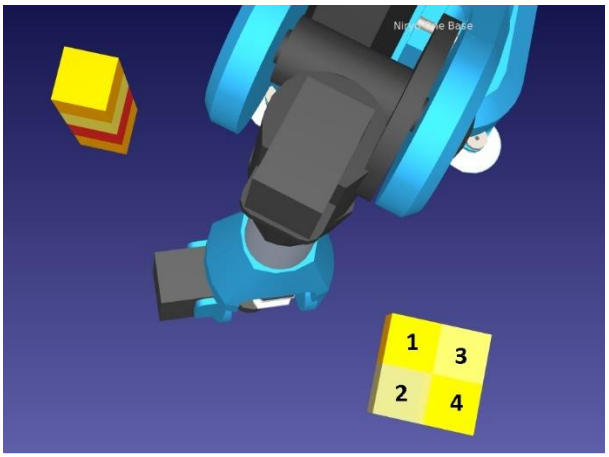

Figure 4 – Target layout

The speed in which the end-effector approaches both the stack and the final positions must also be very slow, to prevent the stack from falling and facilitate the de-attachment process. Other than that, it was noticed that the robot's movement was also improved when moving more slowly. The speed of the robot was then reduced to 25%, and to 5% when approaching the stack and target positions. In order to keep the robot from starting a new movement before finishing the previous order, there was also a 1 second interval added between each action.

It was also noticed that the robot was much more precise when doing small movements. It was decided to do 2 stops during the biggest distance that the robot had to cover, between the stack and the final position of the blocks, instead of just doing one big translation.

The first movement to approach the stack was also divided in two. One just to rotate to the stack, moving only Joint 1. This joint is then fixed for all

the movements to get on top of the stack and reaching every block.

Another observation made was that the robot was more precise when moving fewer joint angles at the same time. So, the number of joints moving between positions were minimized, being usually no more than 3 out of the 6.

Joint 6, the closest to the end-effector was found out to be the one causing the most trouble. This could be due to the fact that it's the furthest joint to the base of the robot. It was then kept unchanged for almost the entire computation. This joint had massive effect on the orientation of the block. So during the movement from the stack to the final position we found a way to keep the orientation fixed moving the other 5 joints. This also resulted in the end effector being in a lower position, closer to the objective.

The positions of the end effector in relation to the blocks, for both the stack and final positions can be visualized in Figures 5 and 6. The height of the blocks was considered to be 33mm, given that the velcro would add around 2mm in height. The overlapping of the block with the end-effector is acceptable, since it would be the result of the velcro's connecting.

The blocks are represented in blue and the end effector in red.

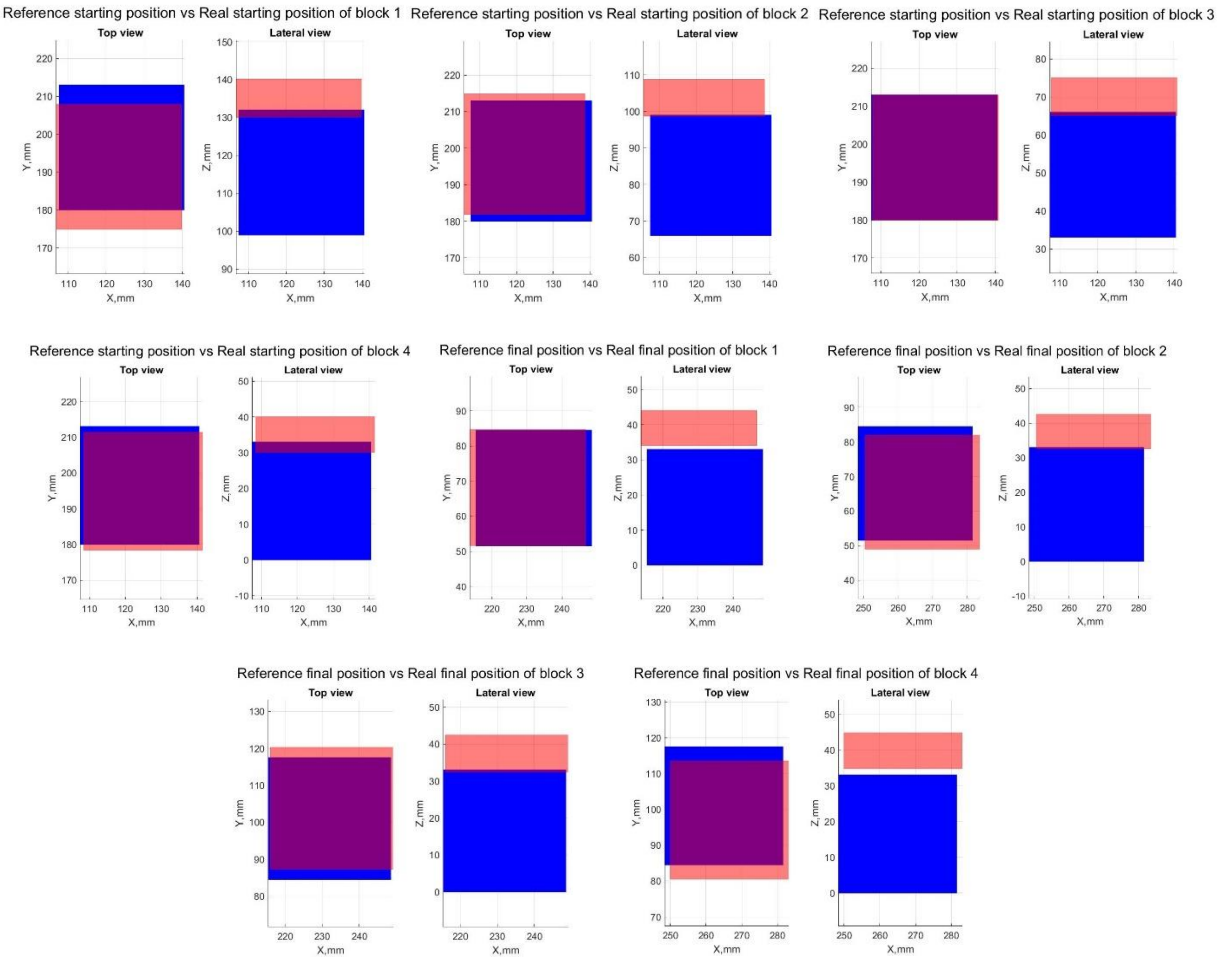

Figure 5 – Position of end-effector in relation to the blocks in the stack and final positions

For the real robot it was noticed, using the data collected in the lab, that there were fixed uncertainties, caused by the wear and loosening of the joints. These can be observed in figure 6.

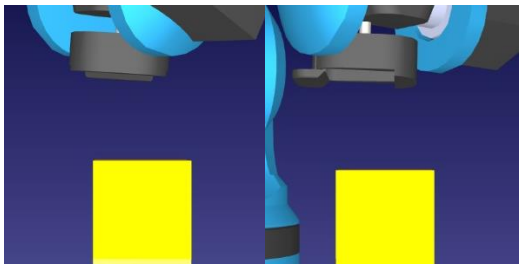

Figure 6 – Displacement in the stack for real robot

Furthermore the usage of learning mode also produced some big displacements, since the robot miscalculated the distance it was covering, probably because it was moved too fast. In the lab when moving to final position 1, the robot moved exactly one block to the left on the y-axis and was in line with the target on the x-axis These offsets can be visualized in figure 7.

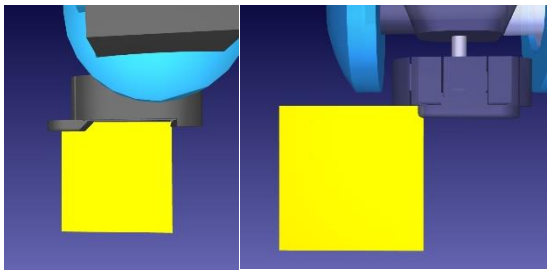

Figure 7 – Displacement in the final position for real robot

The error caused by the attachment and de-attachment of the velcro's was not taken into consideration, since the impact of that force in the robot was very unpredictable to be studied in a simulation.

5. Block diagram of the work developed

With the use of the direct kinematics, the get_joints command and the RoboDK simulator we were able to complete the strategies proposed before. A diagram of the strategy used, both for the simulation and the real robot is present in figure 8.

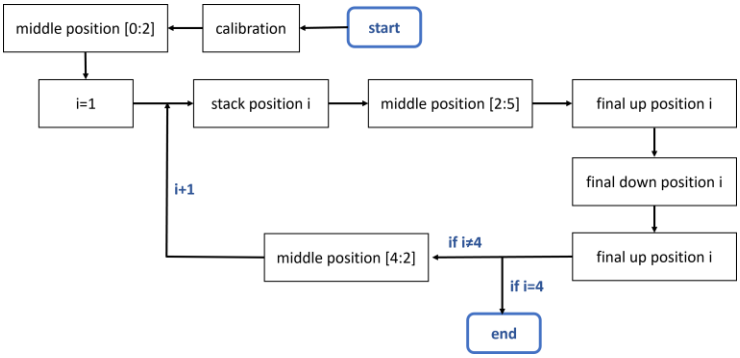

Figure 8- block diagram

6. Scalability of the work

Scalability has been defined as the capability of a system, network, or process to handle a growing amount of work, or its potential to be enlarged to accommodate that growth. The main limit of Niryo One is that there is a lack of control of the errors, with the risk that they became cumulative. Using probabilities and studying how the system behaves in a small scale, it is possible to adapt the solution to a larger problem and use it to solve problems in various areas, such as factory work, construction, medicine, etc.

It is also possible to quantify the capability of a system to be scaled up, in order to guarantee it. The universal scalability law can help us to predict how a system will behave when scaled up.

References

[1] Niryo, *Niryo One User Manual,* 03/08/2019.

[2]Niryo, *Programming the Niryo One with Blockly.*

[3]J. Galarza e E. Luis , «6 DOF anthropomorphic robot as a platform for,» *IEEE/ASME International Conference on,* 2020.

[4] J. S. Sequeira, *Robotics, slide 1,2,3,4,* 2021-2022.

[5] A. Loungthongkam e R. Chana, «The Analysis of Standard Uncertainty of Six Degree of Freedom (DOF) Robot,» *Department of Industrial Engineering, Kasetsart University, Bangkok 10900, Thailand,* 10 March 2021.

[6] A. Loungthongkam e R. Chana, «A Development of Mathematical Model for Predictive of The Standard Uncertainty,» *2020 IEEE 7th International Conference on Industrial Engineering and Applications,* 2020.

[7] MónicaPhilippart, «Chapter 12 - Human reliability analysis methods and tools,» in *Space Safety and Human Performance*, 2008, pp. 501-568.

[8] «Basic Guide-RoboDk,» [Online]. Available: https://robodk.com/doc/en/Basic-Guide.html#GetStarted.

[9] «RViz User's Guide,» [Online]. Available: http://docs.ros.org/en/indigo/api/rviz/html/user_guide/.

[10] «2022 GitHub, Inc.,» [Online]. Available: https://github.com/PickNikRobotics/rviz_visual_tools.