KINOVA

Iterative Inverse Kinematics

2021-04-21

Gen3 lite Cursus

KINOVA

Table of content

Objectives

In this lab, we will study the inverse kinematics of Gen3 lite using an iterative method. Although iterative methods can be used to solve very complex systems of equations, their performance depends highly on the initial guess used to start iterating. In some cases, calculations may converge very quickly, in others it may take a while, and in some it is possible that it never happens . To make sure that we have at least a decent initial guess for our algorithms, we will use the output of the function we implemented during the lab *Inverse Kinematics Simplified Model*. We will implement our iterative method, and then we will experimentally compare our results with data obtained from the real robot.

About Gen3 lite

As usual, all the information we need for this lab is in the User Guide. For convenience, a copy of the figure presenting the robot reference frames and dimensions is illustrated in Figure 1.

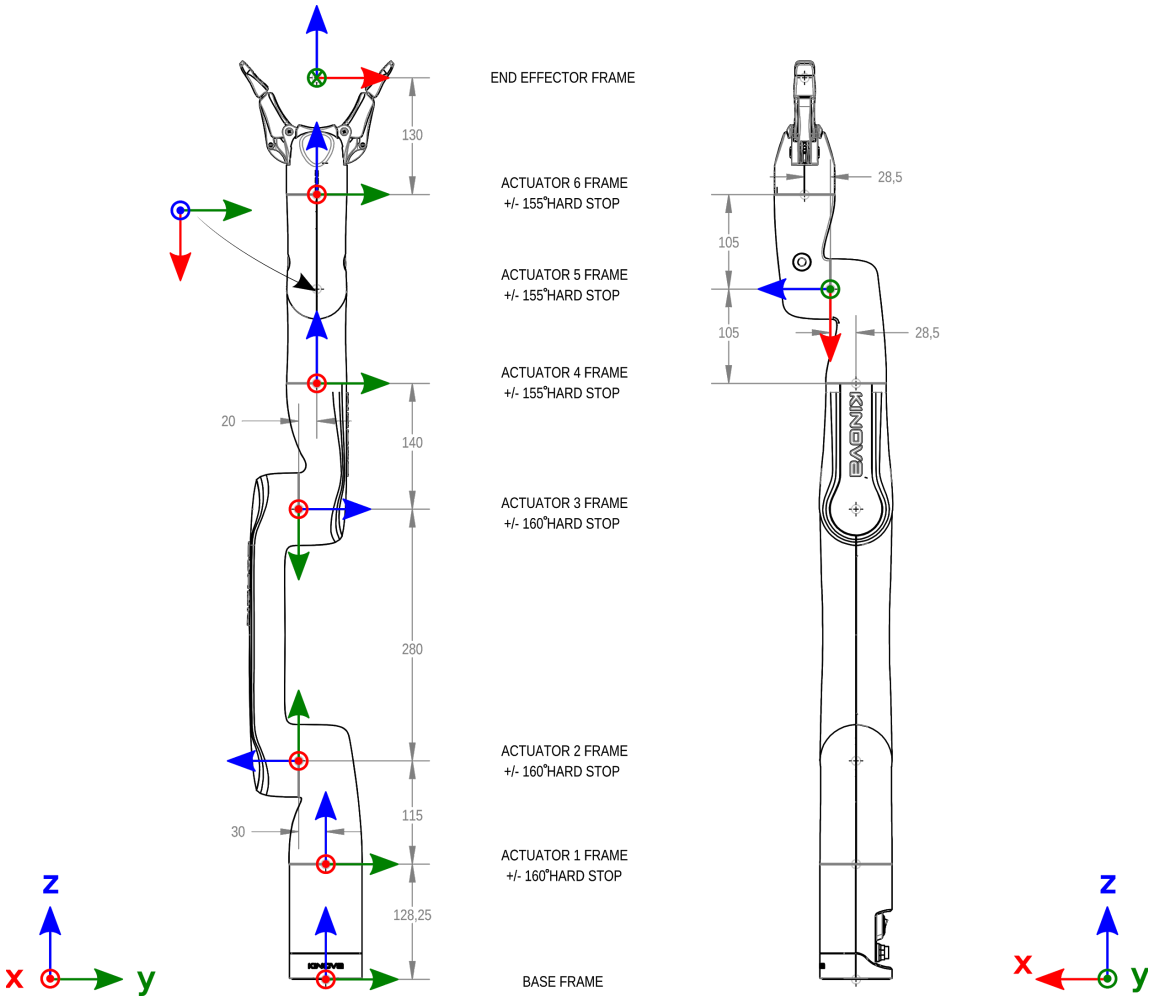

FIGURE 1: Actuator frames and dimensions

In addition to the dimensions of the robot, we will also need the joint position limits, which are indicated in Table 1.

Joint #	1	2	3	4	5	6
Position limit (deg)	± 154.1	± 150.1	± 150.1	± 148.98	-144.97, +145.0	± 148.98

TABLE 1: Gen3 lite Joint position limit

Finally, we will also reuse the simplified geometric model we have used in the previous lab.

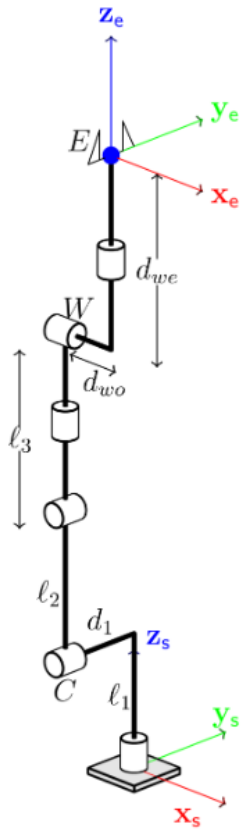

FIGURE 2: Simplified geometric model

Theory

We will implement the following algorithm, which takes our initial simplified model and tries to correct the error caused by our spherical wrist approximation:

0 - Assume an initial guess for $\boldsymbol{\theta}_6^k$
1 - Based on $\boldsymbol{\theta}_6^k$, compute the real position of point *W*.
2 - Use the simplified model to compute the inverse kinematics solution $\boldsymbol{\theta}^{k+1}$ using this value of W.
3 - If abs($\boldsymbol{\theta}_6^k$ - $\boldsymbol{\theta}_6^{k+1}$) < tolerance, we have a solution, otherwise $k = k + 1$and return to #1.

Manipulations

Part 1: Implementation of the inverse kinematics

In this section, we will implement the above algorithm in MATLAB.

1.1 Determine the equations describing the exact coordinates of *W* in the base reference frame when the position of the end-effector *E* in the same frame and $\boldsymbol{\theta}_6$ are fully known.

1.2 Write a function called **ik_gen3_lite_iterative**(T, type, sign), which uses the equations obtained in **1.1**, the above algorithm and your functions from the previous lab to compute the inverse kinematics. The inputs of the function are the same as **ik_gen3_lite_simplified**(T, type, sign), the function we implemented in the previous lab to compute the inverse kinematics. Use a tolerance of 10^{-3} and a maximum number of iterations of 200. Use the simplified solution as well to compute your initial guess for $\boldsymbol{\theta}_6^k$.

Part 2: Validation on the robot

In this section, we will write a script to send cartesian commands to the robot and compare the angular pose calculated by the robot to the one obtained from our own calculation. You should reuse the code that we developed during the *Introduction to Gen3 lite* lab. As a reminder, connection to the robot is established via the **CreateRobotApisWrapper**() function, angular commands are sent using **ReachCartesianPose**() and the angular readings are obtained via **RefreshFeedback**(). For more detail on the MATLAB API, consult the documentation.

2.1 Using either the Web App or your script, use joint commands to identify poses where the robot is in each of the possible configurations. Fill the second and third columns Table 2. You may also reuse the points you had in the previous lab.

Configuration	Measured Joint positions	Cartesian pose	Calculated Joint positions
‘rd+’			
‘rd-’			
‘ru+’			
‘ru-’			
‘ld+’			
‘ld-’			
‘lu+’			
‘lu-’			

TABLE 2: Experimental kinematics results

2.2 Input the cartesian pose you noted in Table 2 in your **ik_gen3_lite_iterative** function and fill the last column of Table 2. You should reuse the **cart2pose** function we implemented during the lab on *Forward Kinematics*.

2.3 How does our new algorithm compare to our simplified model?

2.4 Try running your set of points again, but this time always use $\boldsymbol{\theta}_6^k = 0$ as an initial guess. What do you notice?

2.5 What would be a limitation of using this iterative method to compute the inverse kinematics?

Acknowledgements

This document was produced in collaboration with the following establishments

Including direct implication from the following people:

- Prof. David Saussié, Eng., M.A.Sc., Ph.D., Department of Electrical Engineering.
- Alexis Tang, M.Eng., Department of Electrical Engineering
- Prof. Jérôme Le Ny, Eng., M.A.Sc., Ph.D., Department of Electrical Engineering.
- Prof. Richard Gourdeau , B.A.Sc., M.A.Sc., Ph.D., Department of Electrical Engineering.
