General outline of lab assignments

In these lab assignments you will derive the kinematics and dynamics of the robot manipulator ABB IRB 120 (see description below) to design several controllers. Throughout the three assignments, you will first develop a model library of Matlab functions describing the kinematics (Lab 1) and then dynamics (Lab 2) to finally develop a set of controllers for the manipulator (Lab 3). Each assignment consists of compulsory exercises where you need to do some implementation in Matlab as well as voluntary exercises worth in total **5 points**. The voluntary exercises are clearly marked as **OPTIONAL**.

For each assignment you can download the code from the Canvas Assignments page in a ZIP folder and fill in the necessary functions on your local computer. Once you have completed an exercise by filling in the *TODOs* in the files, you can copy paste your code to the correct location on the Canvas MATLAB Grader page to get immediate feedback. Working on your local computer is recommended, since in the ZIP folder you are also provided with a Simulink file to visualize the robot simulations. Note that for the visualization to work, you need Simulink version R2020b or higher and have the *Robotics System Toolbox* add-on installed. You can run the visualization using the Matlab script `run_XXXXX_YYYYY.m` files found in each folder.

Submission

For each assignment there will be a coding component to be finished on Canvas through the MATLAB Grader tool. The results for this component are immediately visible when you submit your solution. All MATLAB Grader exercises have an unlimited number of submissions. Additionally, a short report with the answers to the questions, the manual calculations (e.g., D-H table, robot schematics, etc.), and the discussions should be submitted in a PDF in the assignment page. You should submit your Matlab functions and scripts, and Simulink modules as a zip file to the same assignment page.

The ABB IRB 120 robot manipulator

The system used in this assignment will be the ABB robot IRB120 (see Fig.1). The ABB IRB120 is a robot with six revolute joints. However, only four of the joints will be used in this assignment. The four actuated joints **Axis 1**, **Axis 2**, **Axis 3**, and **Axis 5** are shown in Fig.1 a), and the physical parameters for the system are shown in Fig. 1 b). The data sheet with the robot's specifications provided by the manufacturer can be found as part of the zip in Canvas (**abbIRB120_specs.pdf**).

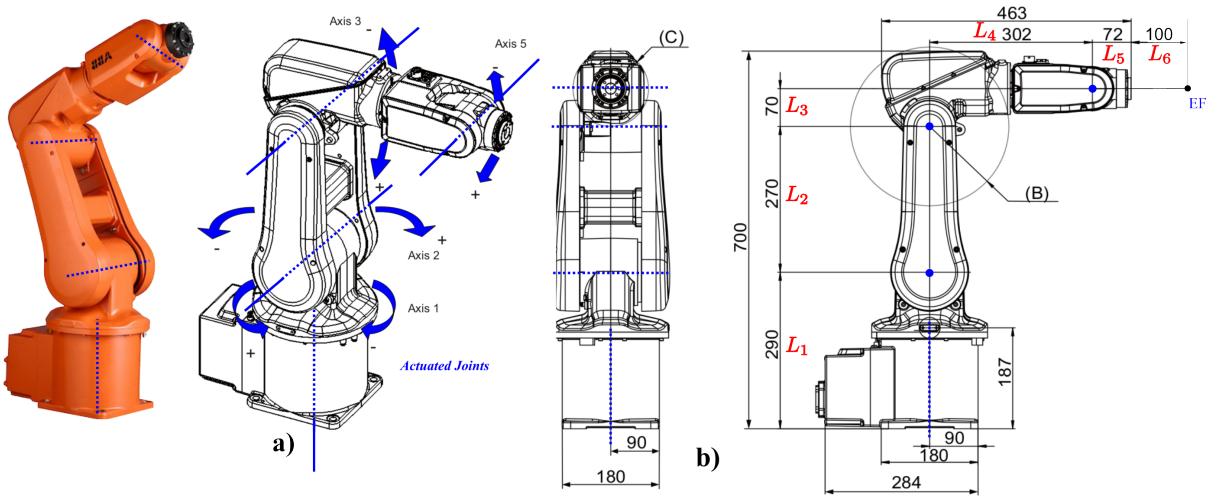

Figure 1: Robot ABB IRB 120: The robot only uses the joints 1, 2, 3, and 5.

Assignment 1 - Kinematics

This assignment consists of two parts. First, it involves the kinematic and differential kinematic modelling of robot. The main modelling task consists in deriving the kinematics using Denavit-Hartenberg parameters (Distal method). The second part includes a validation and application task, where you will validate and use the obtained models.

Modelling: Forward & Differential Kinematics

The main task is to obtain the forward kinematics of the robot in the form of Homogeneous Transformations for each of the joints and differential kinematics, i.e., Jacobian matrix of the end-effector. Within this task it is recommended that you develop generic functions that calculate both the forward kinematics and differential kinematics for any robotic manipulator given the Denavit-Hartenberg (DH) parameters. You will use these functions throughout the course. This task is divided in two parts. First, you will compute the Forward Kinematics and Differential Kinematics, and complete the functions found in the `RobotABB_IRB120_4DOFS/Model` folder. Second, you will use the obtained models and validate them in simulations by finishing the functions in `01_TestKinematics/` and `02_ManipulabilityEllipsoid/`.

Exercise 1 .....

First, compute your relative and absolute Homogeneous Transformation (HT) matrices to obtain the pose of each link and the end-effector (ef). It is recommended that you create general functions, using, for example, Matlab's Symbolic Toolbox to automate this process. Finalize the functions `Model/abbIRB4_params` and `Model/getAbsoluteHT_abbIRB4`. The first function generates an array with the kinematic parameters of the robot. Some of the parameters are already defined in Fig. 1, e.g., $L_1, L_2, \ldots, L_6$. You need to calculate the additional parameters in the parameter function. The second function receives a joint position vector, an array with the kinematic parameters, and the HT of the robot base ($link_0$) with respect to (wrt) the world coordinated frame (wcf). The output of this function is two cell arrays, one with the HTs of each link wrt the robot base ($link_0$), and the other with the HTs of each link with respect to the wcf (w). When in doubt about frame placement due to multiple solutions, choose the solution with the *x-axis* pointing towards the front of the robot or upwards. Also, x_e for the end-effector is pointing in to the screen for the last illustration from Fig. 1. Finally, recall that the *z-axis* should always be in the actuator's positive direction.

For the report: Provide the DH parameters and an illustration with the drawn frames.

Exercise 2 .....

Compute the Jacobian for the end-effector relative to the robot base ($link_0$). It is also recommended that you generate a generic function for this. Then, finalize the function `Model/J_EF_abbIRB4` and `Model/DifFK_abbIRB4`. The first function receives a vector of joint positions ($\mathbf{q}$) and the array of kinematic parameters L. The output is the Jacobian of the ef wrt robot base, $\mathbf{J}^0_{ef} \in \mathbb{R}^{m \times n}$. The second receives the joint position and velocity vectors ($\mathbf{q}, \dot{\mathbf{q}}$), and the kinematic parameters (L) to compute the linear and angular velocities of the end-effector wrt the robot base, $\dot{\mathbf{x}}^0_{ef} \in \mathbb{R}^m$.

For the report: Nothing.

Kinematic Model Validation

Now you will validate the models generated in Exercise 1 and 2. You will use two simulators to validate the different models. The first one is found in the folder `01_TestKinematics/` and the second one in `02_ManipulabilityEllipsoid/`. To evaluate the differential kinematic models (Jacobians), the first step is to validate the obtained Jacobian matrix of the end-effector ($\mathbf{J}^0_{ef}$).

By definition, the closed-form of a Jacobian is $\mathbf{J}^0_{ef} = \left[\frac{\partial \mathbf{t}^0_{ef}}{\partial \mathbf{q}} \; \frac{\partial \theta^0_{ef}}{\partial \mathbf{q}}\right]^\top$, and the linear and angular velocities of the end-effector can be obtained in two forms:

$$\dot{\mathbf{x}}^0_{ef} = \frac{d\mathbf{x}^0_{ef}}{dt} = [\mathbf{v}^0_{ef}\, \omega^0_{ef}]^\top$$
$$\dot{\mathbf{x}}^0_{ef} = \mathbf{J}^0_{ef}\mathbf{q}$$

where $\mathbf{x}^0_{ef} = [\mathbf{t}^0_{ef}\, \theta^0_{ef}]^\top$. In this evaluation, you will have to compare both velocity vectors to verify the Jacobian matrix.

Hint: Use the relation $\mathbf{S}(\omega) = \dot{\mathbf{R}}\mathbf{R}^\top$ to obtain the angular velocity of the end-effector ω_{ef}^0.

Exercise 3 .....

Finish the function `01_TestKinematics/plotRobot`. This function will use the functions generated in the folder `Model`. You can execute this function by running the script: `run_abb_irb1204_tk`. This script will add the dir `Model` to the Matlab path, and plot the robot. Each time you click on the plot the robot will change its position. You should see the robot and the coordinate frames (cf) of each link. Your task is to verify that all cf are aligned with the robot in all the different joint positions, see Fig. 2.

For the report: Provide figures showing the robot and the cf with a brief discussion in the report.

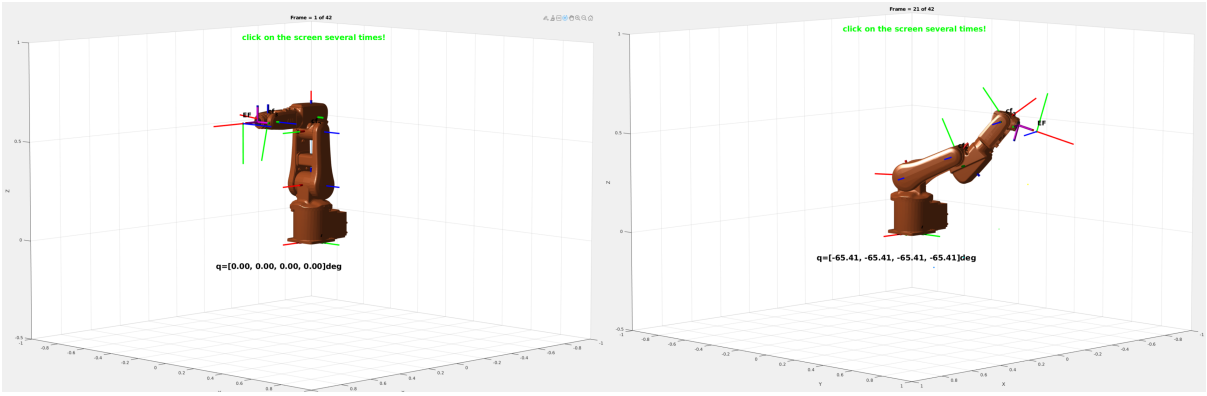

Figure 2: Expected results from the Kinematic test.

Exercise 4 .....

Complete the function `02_ManipulabilityEllipsoid/manipulability_abb_irb4`. This function receives the joint positions and velocities $(\mathbf{q}, \dot{\mathbf{q}})$, the current simulation time t, and the simulation sample time Δt. The function will generate the HTs of all the links (wrt the robot base and w), and the linear and angular velocities of the end-effector calculated with both methods (Jacobian and numeric). Use the Jacobian model to compute the manipulability index w. In this case, you will use the linear velocities Jacobian to obtain the manipulability index and ellipsoid.

For the report: Nothing.

Exercise 5 .....

Once you have finished with the function `02_ManipulabilityEllipsoid/manipulability_abb_irb4`, you can execute it by running the script `run_abb_irb1204`, which launches a Simulink model. You can then start the simulation by clicking on "Run" (see Fig.3). This will open the visualization of the manipulability ellipsoid on the end-effector and the end-effector velocity plots. In the Simulink model, you can switch between constant and time-varying joint position. The constant position is useful to analyze the singular joint configurations while the time-varying position is useful to compare the end-effector velocities calculated with both methods. The ellipsoid's shape changes with different joint configurations. In your submitted report, please provide figures showing the robot simulation and the plots that you obtained with your code. Also, provide a brief discussion, explaining your results' interpretation. The expected results for this exercise are depicted in Fig. 4 and 5.

For the report: Figures and discussion.

Exercise 6 .....

OPTIONAL: 0.5 p

The manipulability ellipsoid's shape changes when we change the joint configuration. Explain why and give some examples to support your answer.

Exercise 7 .....

OPTIONAL: 0.5 p

Include in the report three joint configurations that produce the following results:

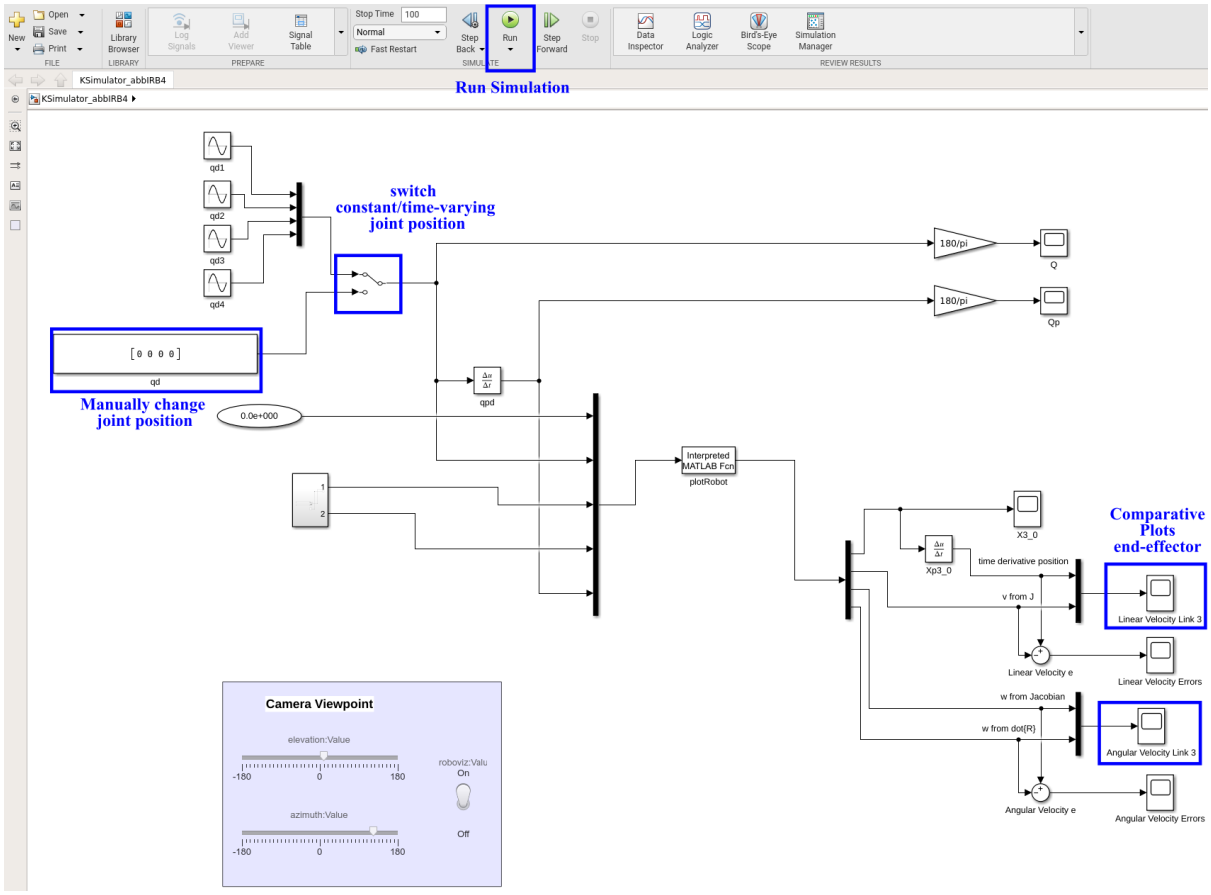

Figure 3: Simulink model for Jacobian test.

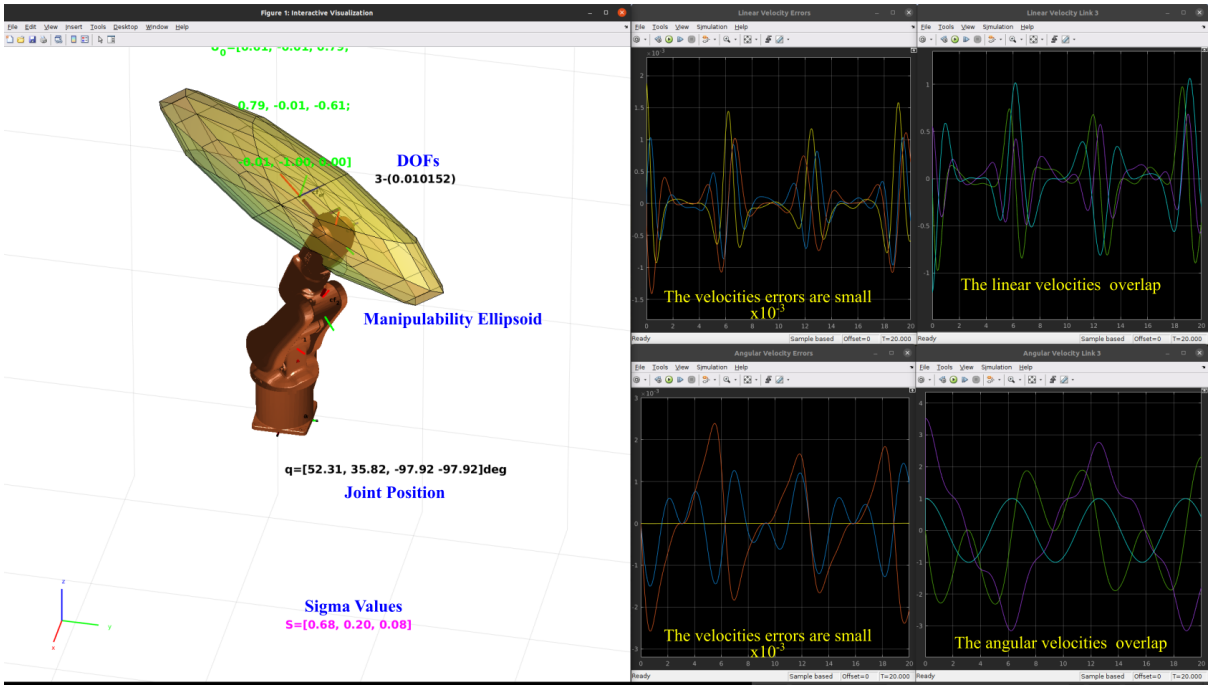

Figure 4: Expected results from Exercise 5 and 6. Left: Manipulability Ellipsoid, Right: Comparative plots of the end-effector velocities.

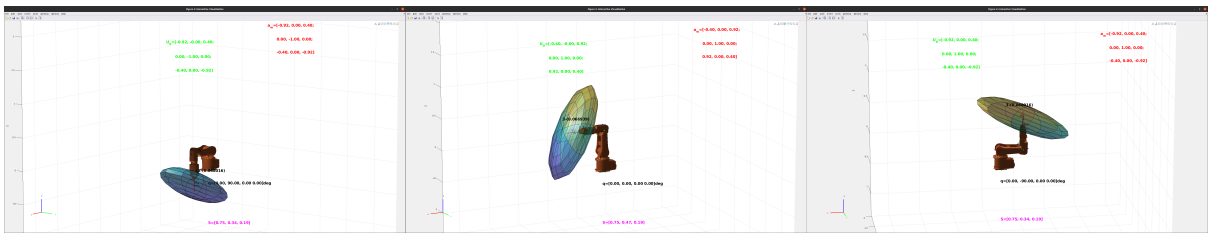

Figure 5: Manipulability Ellipsoid for different joint configurations.

$\mathbf{q}_1$: The robot is in a singular configuration, and the DOFs=2. What is the shape of the ellipsoid?
$\mathbf{q}_2$: The robot is in a singular configuration, and the DOFs=1. What is the shape of the ellipsoid?
$\mathbf{q}_3$: The robot is in a singularity-free configuration. What is the shape of the ellipsoid?

Provide the different joint configurations and figures showing the manipulability ellipsoid for each case, e.g. Fig. 5. *Hint*: You can obtain the joint configurations either by solving the singularity equation or by empirically testing different joint positions and observing the resulting ellipsoids.

Kinematic Model KUKA Robot

In our case, the ABB IRB 120 robot has 4 DOFs which means that it cannot control the full pose of the end-effector (6D). In the following exercises, you will obtain the kinematic and differential kinematic models for the KUKA IIWA 7 robot, which has 7 DOF. Fig. 6 shows the kinematic chain of the robot and its parameters.

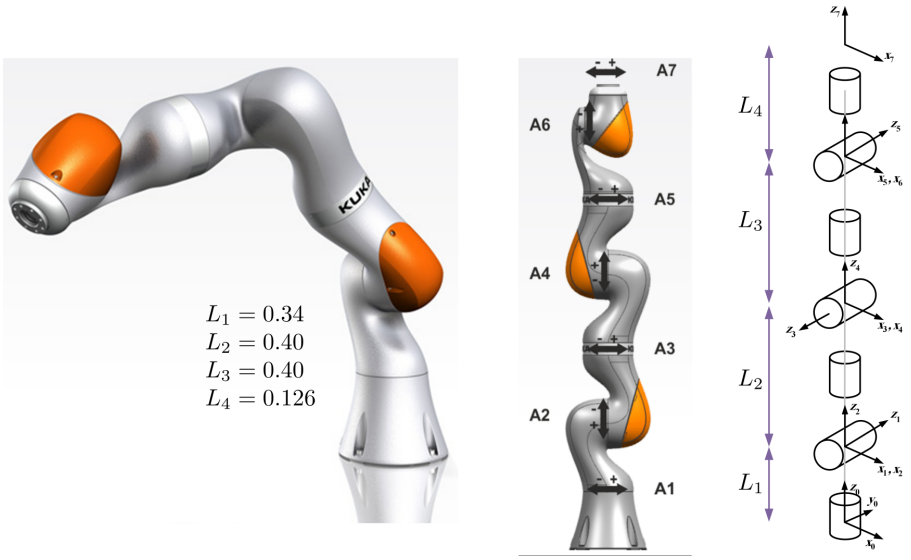

Figure 6: KUKA IIWA 7 robot. The figure shows the kinematic parameters and the kinematic chain for the robot.

For these exercises, you will use the models and the functions found in the folder `Robot_Kuka_IIWA_7DOFs`. This folder has the same structure as the folder `RobotABB_IRB120_4DOFS`. You should follow the same instructions as in the previous exercises.

Exercise 8 .....

OPTIONAL: 0.75 p

Repeat Exercise 1 but using the functions in `Robot_Kuka_IIWA_7DOFs/Model`.

Exercise 9 .....

OPTIONAL: 1 p

Repeat Exercise 2 but using the functions in `Robot_Kuka_IIWA_7DOFs/Model`.

Exercise 10 .....

OPTIONAL: 0.5 p

Repeat Exercise 3 but using the functions in `Robot_Kuka_IIWA_7DOFs/01_TestKinematics`. Provide figures showing the robot and the cf with a brief discussion in the report. Fig. 7 shows the expected results from this task.

Exercise 11 .....

OPTIONAL: 0.75 p

Repeat Exercise 4 but using the functions in `Robot_Kuka_IIWA_7DOFs/02_ManipulabilityEllipsoid`.

Exercise 12 .....

OPTIONAL: 0.5 p

Repeat Exercise 5 for the KUKA robot and provide figures showing the robot simulation, plots, and the manipulability ellipsoid with a brief discussion in the report. Fig. 8 shows the expected results.

Exercise 13 .....

OPTIONAL: 0.5 p

Fig. 8 shows the manipulability ellipsoid for the linear velocities. In this case, the KUKA robot can control both the position and orientation of the end-effector independently. How can you obtain the manipulability ellipsoid for the angular velocities? Provide a small piece of code in the report showing the difference for the linear and the angular case. Provide also a figure showing the robot with the ellipsoid for the linear velocities and the angular velocities (side-by-side) for the same joint position $\mathbf{q} = [0\,0\,0\,0\,0\,0\,0]^\top$, similar to Fig. 5. Explain the differences and why the ellipsoids have those shapes. *Hint*: What is the meaning of the ellipsoid's axes?

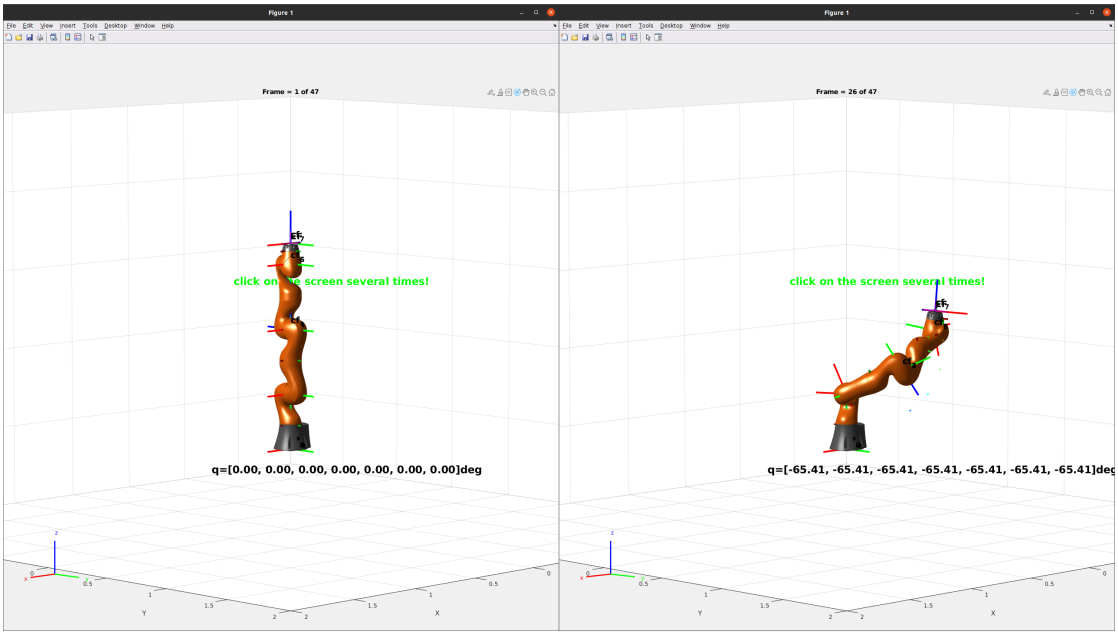

Figure 7: Expected results from Exercise 10.

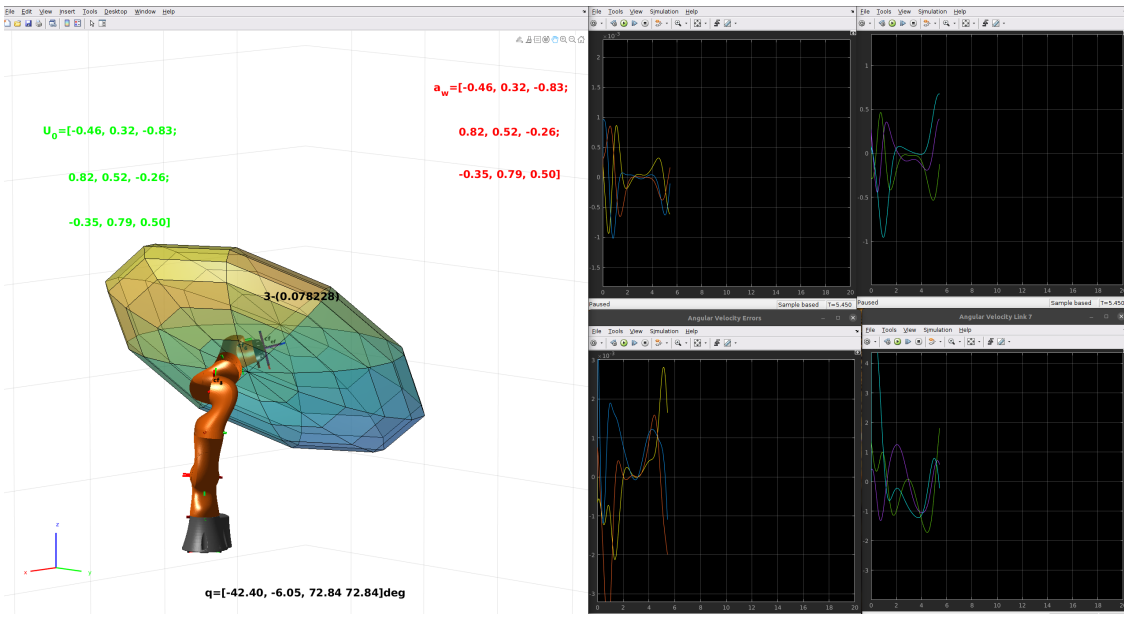

Figure 8: Expected results from Exercise 12.