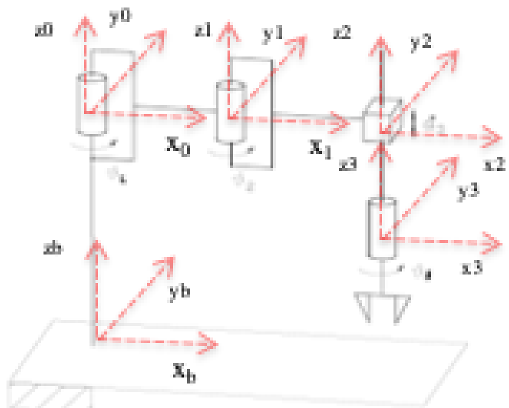
1 Direct and Inverse Kinematics

Consider the SCARA manipulator depicted below. For this project only the kinematic parameters are needed. You have received a trajectory for the the manipulator end effector. The trajectory is provided in a file named kinematic_traj.mat and can be read using init.m.



The manipulator parameters are

$$d_0 = 1\ m,\ a_1 = a_2 = 0.5\ m$$

$$\theta_{1_{min}} = -\pi/2\ rad,\ \theta_{1_{max}} = \pi/2\ rad,\ \theta_{2_{min}} = -\pi/2\ rad,\ \theta_{2_{max}} = \pi/4\ rad$$

$$d_{3_{min}} = 0.25\ m,\ d_{3_{max}} = 1\ m,\ \theta_{4_{min}} = -2\pi\ rad,\ \theta_{4_{max}} = 2\pi\ rad$$

The frames are depicted into the figure and the DH parameters are

	d_i	α_i	θ_i	a_i
Link 1	0	0	θ_1	a_1
Link 2	0	0	θ_2	a_2
Link 3	d_3	0	0	0

Table 1: Table with DH parameters.

Please not that the 0 frame is not coincident with the b frame. There is a translation from the ground plane denoted with $d_0 = 1$. The frame 4 is coincident with the frame 3 so can be discarded from your analysis. Be careful on the d_3 component. The range of values is always positive. When the arm is fully extended (down towards the floor) the value is $1m$ whereas 0.25 when retracted (away from the floor). However, when you build your matrix note that d_3 moves along $-z_2$ axis and for this reason you translation in A_3^2 should be negative as $-d_3$.

Questions:

1. Implement in Matlab/Simulink the algorithms for kinematic inversion with inverse and jacobian transpose along the given trajectory. Adopt the Euler integration rule with integration time 1 ms. Implement a final function visualize_results.m for each part including everything in init.m and all the 2d-plots(joint value and error). A sample function called plot_outputs.m is provided for the joint errors.

2. Suppose to relax one component in the operational space, implement in Matlab/Simulink the algorithm for kinematic inversion with Jacobian pseudo-inverse along the given trajectory maximizing in two separate cases the distance from the mechanical joint limits (relax the orientation component ϕ). Just implement the one considering the jacobian inverse.

Instructions:

- Make your code as a combination of matlab and simulink. The structure is already provided in the folders and you should listen the end of the recorded lecture for additonal instructions. You should call your initialization in a function named init.m. This function should load the trajectory and all the manipulator variables that have been previously listed. You will then define your Jacobians in another function named Jacobian.m, which will be loaded in simulink as shown during the class. The file init.m contains as well a call to a 3D visualization of the manipulator behavior. The file direct_kin.m can be used if you like to write the direct kinematics in matlab in case you do not want to write that using simulink blocks.
- Make a different folder of each question. Once init.m is ran in each folder, we should be able to automatically play the simulink environment and obtain results. Show the joint trajectories in the joint space and the errors operational space. The final report should be made according to the template.

2 Report

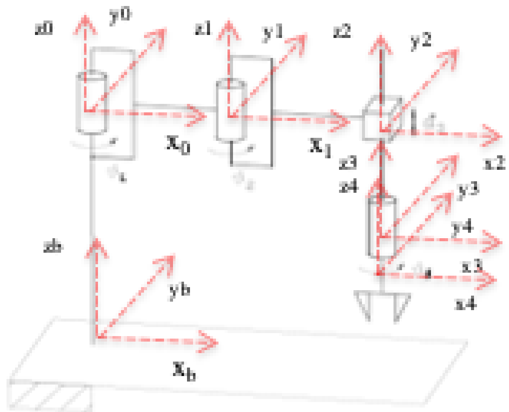
You need to summarize your results in a report submitted in pdf format and generated with latex or word. Please add on top of your manuscript your name and NYU ID. The report should not be more than 8 pages including plots. In addition to the results, please include your models and any explanation you think is appropriate. Do not just write equation, but try to add your logic process and explain why and how you used the equations or models you have in your code.

3 Grade Policy and Submission

The overall score will be 100 and will be subdivided in the following way, part 1 (50 points), part 2 (40 points), and report quality and readability (10 points). Do not modify any part of the code as specified above. Any other type of modification will result in 0 points. All the files, including code and report, should be submitted in an unique zip file.

1 Direct and Inverse Kinematics

Consider the SCARA manipulator depicted below. For this project only the kinematic parameters are needed. You have received a trajectory for the the manipulator end effector. The trajectory is provided in a file named kinematic_traj.mat and can be read using init.m.



The manipulator parameters are

$$d_0 = 1\ m,\ a_1 = a_2 = 0.5\ m$$

$$\theta_{1_{min}} = -\pi/2\ rad,\ \theta_{1_{max}} = \pi/2\ rad,\ \theta_{2_{min}} = -\pi/2\ rad,\ \theta_{2_{max}} = \pi/4\ rad$$

$$d_{3_{min}} = 0.25\ m,\ d_{3_{max}} = 1\ m,\ \theta_{4_{min}} = -2\pi\ rad,\ \theta_{4_{max}} = 2\pi\ rad$$

The frames are depicted into the figure and the DH parameters are

	d_i	α_i	θ_i	a_i
Link 1	0	0	θ_1	a_1
Link 2	0	0	θ_2	a_2
Link 3	d_3	0	0	0
Link 4	0	0	θ_4	0

Table 1: Table with DH parameters.

Please not that the 0 frame is not coincident with the b frame. There is a translation from the ground plane denoted with $d_0 = 1$. The frame 4 is coincident with the frame 3 at the starting. Be careful on the d_3 component. The range of values is always positive. When the arm is fully extended (down towards the floor) the value is $1m$ whereas 0.25 when retracted (away from the floor). However, when you build your matrix note that d_3 moves along $-z_2$ axis and for this reason you translation in A_3^2 should be negative as $-d_3$.

Questions:

1. Implement in Matlab/Simulink a second order algorithm for kinematic inversion with jacobian inverse along the given trajectory. Adopt the Euler integration rule with integration time 1 ms. Implement a final function visualize_results.m for each part including for all the 2d-plots (joint value and operational space errors). A sample function called plot_outputs.m is provided for the joint errors.

2. Suppose to relax one component in the operational space, implement in Matlab/Simulink the second order algorithm for kinematic inversion with Jacobian pseudo-inverse along the given trajectory maximizing the distance from an obstacle along the path (relax the z component). Suppose that the obstacle is a sphere centered in $p = \begin{bmatrix} 0.4 & -0.7 & 0.5 \end{bmatrix}^\top$ with radius 0.2 m.

Instructions:

- Make your code as a combination of matlab and simulink. The structure is already provided in the folders. You should call your initialization in a function named init.m. This function should load the trajectory and all the manipulator variables that have been previously listed. You will then define your Jacobians in another function named Jacobian.m, which will be loaded in simulink as shown during the class. The derivative of the Jacbian is Jacobian_dot.m. The file init.m contains as well a call to a 3D visualization of the manipulator behavior in case you want to use it. The file direct_kin.m can be used if you like to write the direct kinematics in matlab in case you do not want to write that using simulink blocks.
- Make a different folder of each question. Once init.m runs in each folder, we should be able to automatically play the simulink environment and obtain results. Show the joint trajectories in the joint space and the errors operational space.

2 Report

You need to summarize your results in a report submitted in pdf format and generated with latex or word. Please add on top of your manuscript your name and NYU ID. The report should not be more than 8 pages including plots. In addition to the results, please include your models and any explanation you think is appropriate. Do not just write equation, but try to add your logic process and explain why and how you used the equations or models you have in your code.

3 Grade Policy and Submission

The overall score will be 100 and will be subdivided in the following way, part 1 (50 points), part 2 (40 points), and report quality and readability (10 points). Do not modify any part of the code as specified above. Any other type of modification will result in 0 points. All the files, including code and report, should be submitted in an unique zip file.

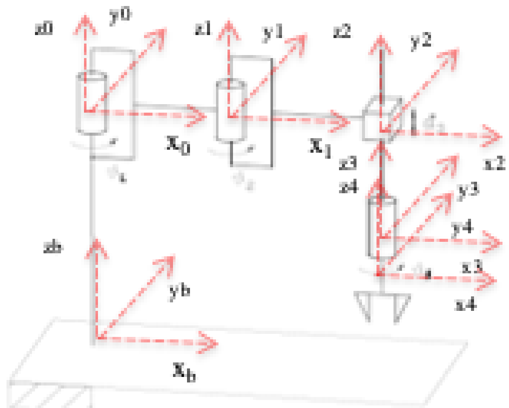
ROB-6003 Foudations of Robotics — **Project 3**

New York University, Fall 2020 — updated November 16, 2020

Due on November 30, 11:59 PM

1 Direct and Inverse Kinematics

Consider the SCARA manipulator depicted below. For this project only the kinematic parameters are needed. You have received a trajectory for the the manipulator end effector. The trajectory is provided in a file named kinematic_traj.mat and can be read using init.m.



The manipulator parameters are

$$d_0 = 1\ m,\ a_1 = a_2 = 0.5\ m,\ l_1 = l_2 = 0.25\ m$$

$$\theta_{1_{min}} = -\pi/2\ rad,\ \theta_{1_{max}} = \pi/2\ rad,\ \theta_{2_{min}} = -\pi/2\ rad,\ \theta_{2_{max}} = \pi/4\ rad$$

$$m_{l1} = m_{l2} = 25\ kg,\ m_{l3} = 10\ kg, Il1 = I_{l2} = 5\ kgm^2,\ I_{l4} = 1\ kgm^2$$

$$k_{r1} = k_{r2} = 1,\ k_{r3} = 50\ rad/m,\ k_{r4} = 20,$$

$$I_{m1} = I_{m2} = 0.0001\ kgm^2,\ I_{m3} = 0.01\ kgm^2, I_{m4} = 0.005\ kgm^2$$

$$F_{m1} = F_{m2} = 0.0001\ N \cdot m \cdot s/rad, F_{m3} = 0.01\ N \cdot m \cdot s/rad, F_{m4} = \ 0.005\ N \cdot m \cdot s/rad$$

$$d_{3_{min}} = 0.25\ m,\ d_{3_{max}} = 1\ m,\ \theta_{4_{min}} = -2\pi\ rad,\ \theta_{4_{max}} = 2\pi\ rad$$

As done for projects 1 and 2, the frames are depicted into the figure and the DH parameters are

	d_i	α_i	θ_i	a_i
Link 1	0	0	θ_1	a_1
Link 2	0	0	θ_2	a_2
Link 3	d_3	0	0	0
Link 4	0	0	θ_4	0

Table 1: Table with DH parameters.

Please not that the 0 frame is not coincident with the b frame. There is a translation from the ground plane denoted with $d_0 = 1$. The frame 4 is coincident with the frame 3 at the starting. Be careful on the d_3 component. The range of values is always positive. When the arm is fully extended (down towards the floor) the value is $1m$ whereas 0.25 when retracted (away from the floor). However, when you build your matrix note that d_3 moves along $-z_2$ axis and for this reason you translation in A_3^2 should be negative as $-d_3$.

Questions:

1. Generate a trajectory in the robot operational space of 4 s with trapezoidal velocity profile for each segment passing trough the following waypoints $p_0 = \begin{bmatrix} 0 & -0.80 & 0 \end{bmatrix}$ at time $t_0 = 0.0$, $p_1 = \begin{bmatrix} 0 & -0.80 & 0.5 \end{bmatrix}$ at time $t_1 = 0.6$, $p_2 = \begin{bmatrix} 0.5 & -0.6 & 0.5 \end{bmatrix}$ at time $t_2 = 2.0$, $p_3 = \begin{bmatrix} 0.8 & 0.0 & 0.5 \end{bmatrix}$ at time $t_3 = 3.4$, $p_4 = \begin{bmatrix} 0.8 & 0.0 & 0.0 \end{bmatrix}$ at time $t_4 = 4.0$. The trajectory should be generated such that the robot should not stop at each waypoint so that the waypoints are via points. The anticipation time for each segment should be 0.2 s. You need show the position, velocity and acceleration of your path. The sampling time will be $T_s = 0.001$ s.

2. Consider a 5 kg load placed at the end effector. Generate an inverse dynamic control approach. The setpoints for each joint have to be generated starting from the trajectory generated in the operational space considering a second order inversion kinematic algorithm developed during project 2.

Instructions:

- Make your code as a combination of matlab and simulink. The structure is already provided in the folders. For part 1, you can create all functions you need, but your program should run by playing the init_traj.m function available in the folder. Once your trajectory has been generated save it in a mat file generated_traj.mat.

- For part 2 as usual, you should call your initialization in a function named init.m. For the second order inverse kinematic algorithm same rules of project 2 apply, so you will find the same files. You will have as well another couple of .m files related to control. You will notice in the simulink file that there is a specific subsystem where you should put your second order inverse kinematic algorithm and will give you as output the desired joint values. The trajectory you generated is loaded in the init.m file as generated_traj.mat. Copy it from the previous part in the part 2 folder once you finished part 1. You will have to show your position and velocity errors in the joint space as well as the joint trajectories. You can use the same function for projects 1 and 2 as reference.

2 Report

You need to summarize your results in a report submitted in pdf format and generated with latex or word. Please add on top of your manuscript your name and NYU ID. The report should not be more than 8 pages including plots. In addition to the results, please include your models and any explanation you think is appropriate. Do not just write equation, but try to add your logic process and explain why and how you used the equations or models you have in your code.

3 Grade Policy and Submission

The overall score will be 100 and will be subdivided in the following way, part 1 (30 points), part 2 (60 points), and report quality and readability (10 points). Do not modify any part of the code as specified above. Any other type of modification will result in 0 points. All the files, including code and report, should be submitted in an unique zip file.