Lab 5 part 1

Preliminary: Matlab installed, Robotics Toolbox(RTB file) from Peter Corke's website is installed (double click from inside Matlab).

A 3DOF RRR manipulator has the kinematic structure represented in Fig.1. Use the Robot Toolbox for Matlab/Simulink (http://www.petercorke.com) to model the robot and to control it in the joint space. If some specifications are missing in this text make reasonable assumptions.

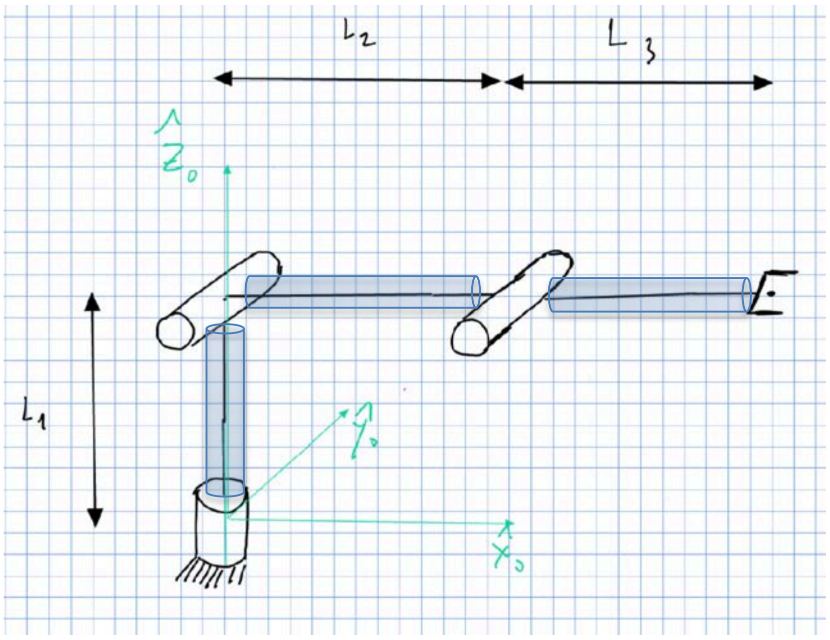


Fig. 1 – robot model for Lab 5

Parameter	Link1	Link2	Link3
MASS [Kg]	0	10	10
LENGH [m]	1	1	1
Parameter	**Joint1**	**Joint2**	**Joint3**
Motor Inertia [Kgm²]	1*10^-4	1*10^-4	1*10^-4
Gear Ratio	500	500	500
Friction [Nms]	10*10^-4	10*10^-4	10*10^-4
Range Limit [deg]	[-180 180]	[-90 90]	[-90 90]

Step 1: Affix the frames to each link (frame zero with origin in the robot base, see Fig.1).

Step 2: Find the DH parameters.

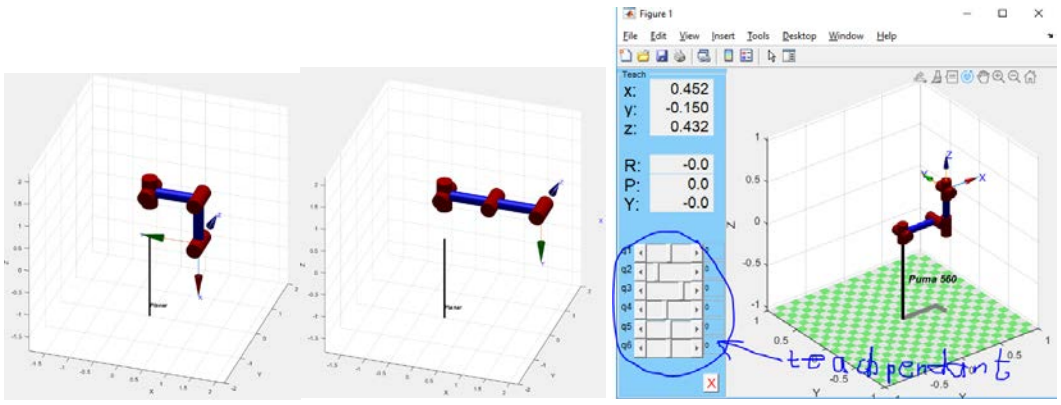
Step 3: Consider each of the three links as a cylinder of homogeneous density having radius of the cross section **R_i=0.05m** and calculate by using the following formula the associate inertia matrix (relative to a frame having as origin the position of the center of the mass (**C_i**) and as orientation the same as the link's frame).

$$
{}^{c_i}I_i = \begin{bmatrix} \frac{1}{2}m_iR_i^2 & 0 & 0 \\ 0 & \frac{1}{12}m_il_i^2+\frac{1}{4}m_iR_i^2 & 0 \\ 0 & 0 & \frac{1}{12}m_il_i^2+\frac{1}{4}m_iR_i^2 \end{bmatrix}
$$

Step 4: Model the robot by using the toolbox in Matlab (when you create the links use the modified DH convention, this is the one we used in our course). You may use the Puma560 model as template (you find it attached).
Hint:
In order to create one link of a robot (in the Peter Corke's book: link object) use
YOUR LINK NAME (link number) = Revolute('d', YOUR JOINT EXTENSION , 'a' , YOUR JOINT OFFSET, … , 'alpha', YOUR JOINT TWIST, ….) that is a revolute link object with the kinematic and dynamic parameters specified by the key/value pairs using the standard Denavit-Hartenberg conventions. All options are described in the Peter Corke's book on page 430.

Step 5: Plot the robot model and test it by using the virtual teach-pendant (verify that is correctly represented as shown in Fig. 2c). **80%**
Hint: Functions *YOUR ROBOT NAME.plot* and *YOUR ROBOT NAME.teach*

Step 6: Find the static torques required at the joints to keep the manipulator in the home (Fig.2a) position (the base joint1 = 0 deg, joint2 = 90 deg, joint3 = 0) with a load at the end-effector of 1Kg (remember to set the gravitational vector constant correctly). Find the static torques required at the joints to keep the manipulator in the hold position (Fig.2b), in which joint2 = 0 deg. If the torques for Fig.2b are higher than in Fig.2a, then your computations in Matlab are most probably correct. **99%**
Hint: Function *jacob0* outputs a manipulator Jacobian matrix, with respect to the world frame, based on the input joint coordinate vector. outputs the Jacobian matrix. The robot object is a parameter.
There is also Function *gravload.* It is also described in the book.



x: 0.452
y: -0.150
z: 0.432
R: -0.0
P: 0.0
Y: -0.0

(a) (b) (c)
Fig.2 – Home position and hold position

***** Extra task:** Do the same of the PLANAR ROBOT that we used in ROS labs 1 - 4 (the links and joints are defined in URDF file). **100%**

Files to submit:

1) Link to Matlab code to Github repository
2) Screen record of the moving robot model (scroll the control joint bar and move the joints) to youtube or your own drive link.
3) In your text with the link to the github answer: Why the torques are different for different configurations?