Introduction to Robotics (CS223A) Homework #3

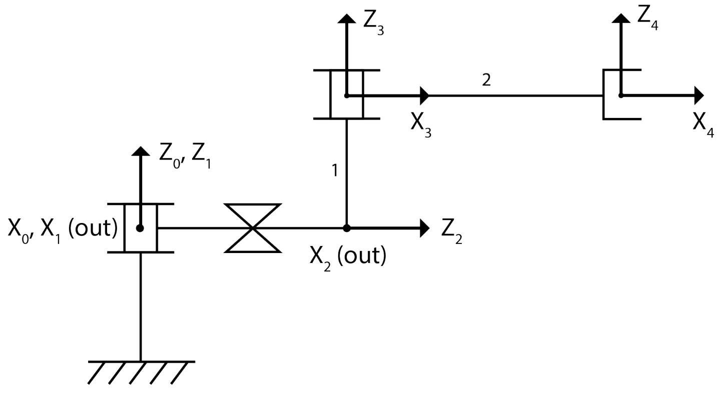
(Winter 2017/2018) Due: **Wednesday, February 7**

Name ________________________ **SUNet ID** ________________________

Some tips for doing CS223A problem sets:

- Use abbreviations for trigonometric functions (e.g. $c\theta$ for $\cos(\theta)$, s_1 or $s\theta_1$ for $\sin(\theta_1)$) in situations where it would be tedious to repeatedly write sin,cos, etc.
- Unless instructed otherwise, leave square roots in symbolic form rather than writing out their decimal values.
- Use common sense for decimals – – if the question states a = 1.34, then don't give answers like 2*a = 2.680001245735.
- If you give a vector as an answer, make sure that you specify what frame it is given in (if it is not clear from context). The same rule applies to rotation and transformation matrices.

1. Let us consider the RPR manipulator with 3 links represented in the schematic below. The schematic is drawn in the configuration $\theta_1 = 0, \theta_3 = 90°$



Luckily, you do not need to compute the forward kinematics, because they are given to you here (note that $c_{13} = \cos(\theta_1 + \theta_3)$):

$$
{}^0_1T = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^0_2T = \begin{bmatrix} c_1 & 0 & -s_1 & -d_2s_1 \\ s_1 & 0 & c_1 & d_2c_1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^0_3T = \begin{bmatrix} c_{13} & -s_{13} & 0 & -d_2s_1 \\ s_{13} & c_{13} & 0 & d_2c_1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0_ET = \begin{bmatrix} c_{13} & -s_{13} & 0 & 2c_{13} - d_2s_1 \\ s_{13} & c_{13} & 0 & 2s_{13} + d_2c_1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

(a) Find the linear velocity and angular velocity of the end-effector in frame $\{0\}$ as a function of the joint variables. Obtain the linear velocity by differentiation.

Hint: Once you've implemented `jacobian.py` in Q2, you can cross check your analytical differentiation solution with numerical velocity propagation for specific joint configurations. The links for Q1 are already defined for you in `jacobian.py`.

(b) For this manipulator, find the joint velocities that will achieve an end-effector angular velocity of $\mathbf{w} = [0\ 0\ 1]^T$ in frame $\{0\}$ while keeping the end-effector's position stationary (that is, linear velocities are zero!), in function of the configuration of the manipulator. Give an equation of the form $\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q})$. Is this equation always valid?

(c) If the robot is stationary (i.e. $\dot{\mathbf{q}} = \mathbf{0}$ and $\ddot{\mathbf{q}} = \mathbf{0}$), and we apply a force (measured in frame $\{0\}$) of ${}^0F = [F_x\ F_y\ F_z]^T$ on the end-effector, what are the resulting joint torques?

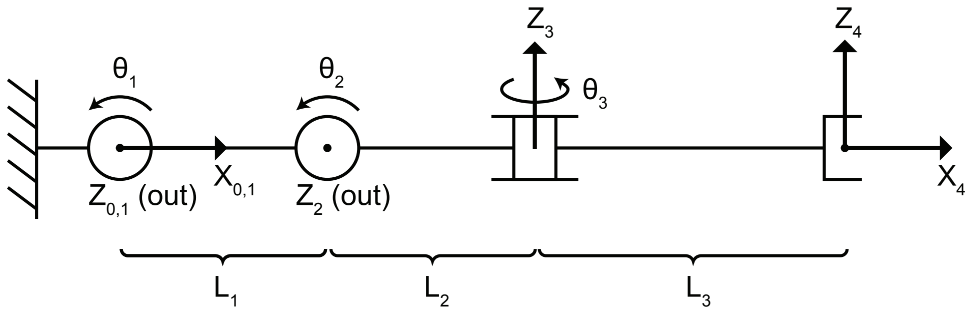
2. Use velocity propagation (Eqs. (4.71) and (4.74) in the course reader) to implement `linear_jacobian()`, `angular_jacobian()`, and `basic_jacobian()` in `jacobians.py` for a general serial manipulator. The Jacobians should be expressed in frame $\{0\}$. This part will be evaluated by the autograder. Please read the function comments carefully.

   There are four arguments to compute the Jacobian: the kinematic structure of the robot, the joint configurations q, the position of the point on the robot whose velocity you want to compute (often the end-effector), and the link that the point is attached to (often the last link). You could compute the Jacobian for a point attached to link 1 of a 3-dof robot, for example. In this case, the second and third joints would have no effect on the velocity at this point.

   Hint: The `Link` class is defined in `links.py`. Take a look at all the functions in `links.py` as they may be useful.

   Hint: Compare your implementation with your answer from Q1a for the given RPR manipulator. Also try using the Jacobian visualizer in `visualizer.py` to make sure your Jacobian makes intuitive sense.

3. You are presented with the RRR manipulator below. L_1, L_2, and L_3 are strictly positive.



(a) Find the Denavit-Hartenberg parameters for this manipulator. Assign the frames such that all your a_i are positive.

i	a_{i-1}	α_{i-1}	d_i	θ_i
1				
2				
3				
4				

(b) The position of the end-effector is:

$$^0P_4 = \begin{bmatrix} L_1c_1 + L_2c_{12} + L_3c_{12}c_3 \\ L_1s_1 + L_2s_{12} + L_3s_{12}c_3 \\ -L_3s_3 \end{bmatrix},$$

where $c_{12} = cos(\theta_1 + \theta_2)$.

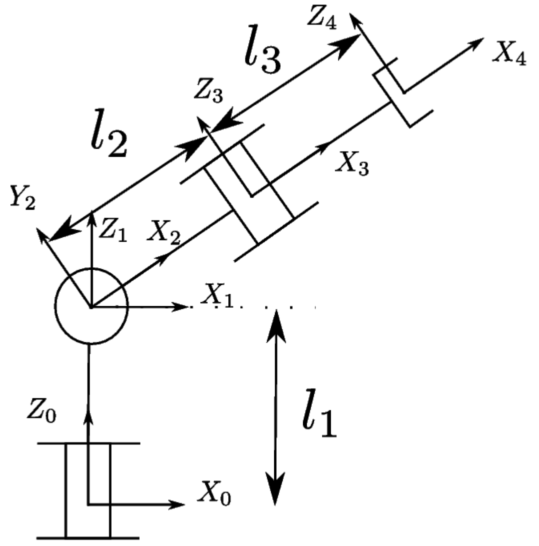
Derive the linear Jacobian ^0J_v.

(c) Find the singular configurations of this manipulator. For each singularity, draw the robot configuration and clearly state how the movement is restricted (in terms of frame axes). Remember that the link lengths of this robot are strictly positive. For some singularities, it may be easier to draw the robot from a different angle.

Hint: The linear Jacobian in frame $\{2\}$ is given to you here:

$$^{2}J_v = \begin{bmatrix} -L_1 s_2 & 0 & -L_3 s_3 \\ L_1 c_2 + L_2 + L_3 c_3 & L_2 + L_3 c_3 & 0 \\ 0 & 0 & -L_3 c_3 \end{bmatrix}$$

4. (a) For the manipulator below, find the linear jacobian ${}^0J_v$ and the angular jacobian ${}^0J_\omega$ for the end effector point (origin of frame $\{4\}$), expressed in frame $\{0\}$. The joint angles are $\theta_1, \theta_2, \theta_3$.



(b) **[extra credit]** Find a manipulator that has no singularities for the task of positioning its end-effector in 3D space. Draw a schematic and prove it has no linear singularities.