ME 221, Fall 2022
University of California, Riverside
Department of Mechanical Engineering

HW 3
Due: 11/3/2022

1. Consider the planar robot in Fig. 1. The forward kinematics can be found from the following transformation matrices:

$$
{}^{0}T_1 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & l_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & 0 & l_1\sin\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{1}T_2 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & l_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & l_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{2}T_3 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & l_3\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & l_3\sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

Solve the inverse kinematics, e.g., find joint angles given the end-effector pose:

$$
\begin{bmatrix} X \\ Y \\ \varphi \end{bmatrix} \rightarrow \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}.
$$

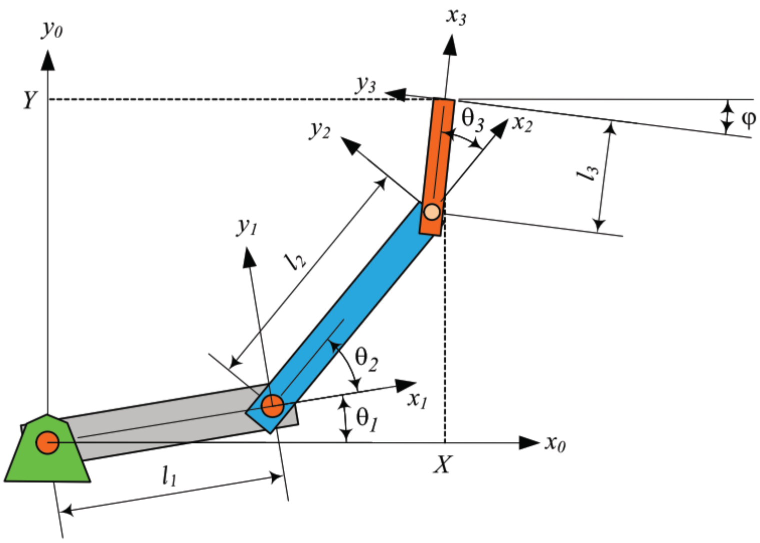

Figure 1: Schematic for Problem 1.

2. Consider a rotation matrix parameterized using z-y-x Euler angles:

$$
{}^{G}R_B = R_z(\phi_1)R_y(\phi_2)R_x(\phi_3)
$$

a. Find the angular velocity vector ${}_{G}\boldsymbol{\omega}_B$

b. Find a matrix M such that:

$$
\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_x \end{bmatrix} = M \begin{bmatrix} \dot\phi_1 \\ \dot\phi_2 \\ \dot\phi_3 \end{bmatrix}
$$

3. Show that for a homogeneous transformation matrix T:

$$\dot{T}T^{-1} = \begin{bmatrix} \tilde{\omega} & \dot{\boldsymbol{d}} - \tilde{\omega}\boldsymbol{d} \\ 0 & 0 \end{bmatrix}$$

4. Find the Jacobian matrix $J(\boldsymbol{q})$ of the 2R planar robot in Fig. 4 through direct differentiation.

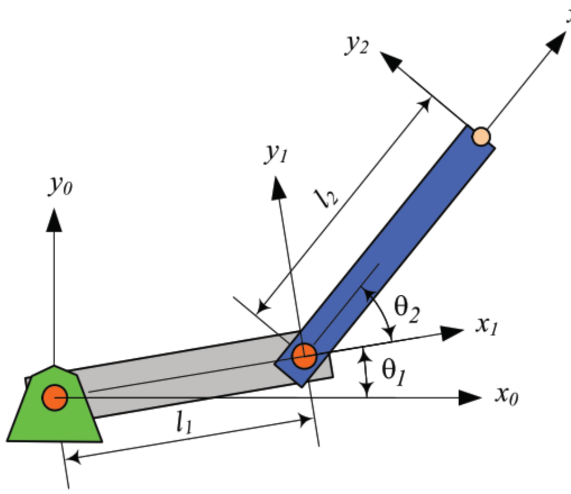

Figure 2: Schematic for Problem 4.

5. In this exercise you will explore inverse kinematics with the MATLAB simulation we have been developing. The idea is to simulate a task where the orientation of the end-effector is important. Make sure you can render the IRB 120 robot in the virtual environment. The goal of this problem is to use inverse kinematics to following a semi-circle trajectory. The challenge will be to ensure the orientation of the end-effector remains **normal** to the trajectory. This simulation is motivated by industrial applications such as sanding or welding operations where the tool must remain normal to the surface.

   a. The end-effector trajectory is the top half of a circle located in the $Y - Z$ plane. The circle has radius 0.25 and is centered at $x = 0.25$, $y = 0$, and $z = 0.25$ (see Fig. 3). First you will need to express the desired end-effector trajectory as a time varying vector:

   $$\boldsymbol{r}(t) = x(t)\hat{I} + y(t)\hat{J} + z(t)\hat{K}$$

   To find the desired orientation note that in the MATLAB simulation the tool frame is parallel with the Global axis when the robot is at the home position (e.g., $\boldsymbol{q} = 0$), as shown in Fig. 3. You want the x-axis of the tool frame to be normal to $\boldsymbol{r}(t)$, or put another way, the z-axis of the tool frame should be tangent to $\boldsymbol{r}(t)$. To find the proper orientation, consider what rotations are need to move the end effector from is home position to the proper normal orientation. (*Hint: The desired orientation can be expressed as x-y-z Euler angles with $\phi_x = \alpha$, $\phi_y = 0$ and $\phi_z = \frac{-\pi}{2}$, where α is the direction cosine of the vector tangent to the curve with the unit z-axis.*)

   b. Use one of the MATLAB inverse kinematic method we discussed in class (see example script `ik_ex.m`) to create an animation of the robot following the desired trajectory. Use `getTransform` to store the achieved end-effector the pose at each time step. Example animation `hw3-ik-analytic.mp4`.

   c. Use the *resolved rates method* to find the inverse kinematics solution. Recall that we can derive a simple update rule from the Jacobian:

   $$\dot{\boldsymbol{q}} = J(\boldsymbol{q})^{-1}\dot{\boldsymbol{x}}$$
   $$\Delta\boldsymbol{q} = J(\boldsymbol{q})^{-1}\boldsymbol{v}$$
   $$\boldsymbol{q}_{k+1} = J(\boldsymbol{q})^{-1}\boldsymbol{v}_k + \boldsymbol{q}_k$$

Implement the above, using `geometricJacobian` and starting with $\boldsymbol{q}_0$ from one of the IK solvers. Generate a subplot to compare the end-effector pose from the actual desired trajectory, and the two IK methods implemented by calculated the error for each component as shown in Fig. 4. You can decompose the end-effector orientation using `rotm2eul` to compare. Jacobian animation: `hw3-ik-jacobian.mp4`.

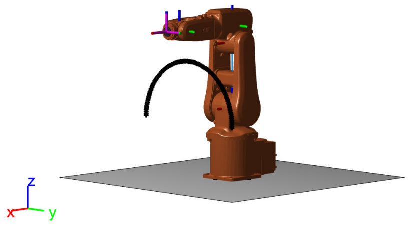

Figure 3: Desired semi-circle trajectory with robot in the home position.

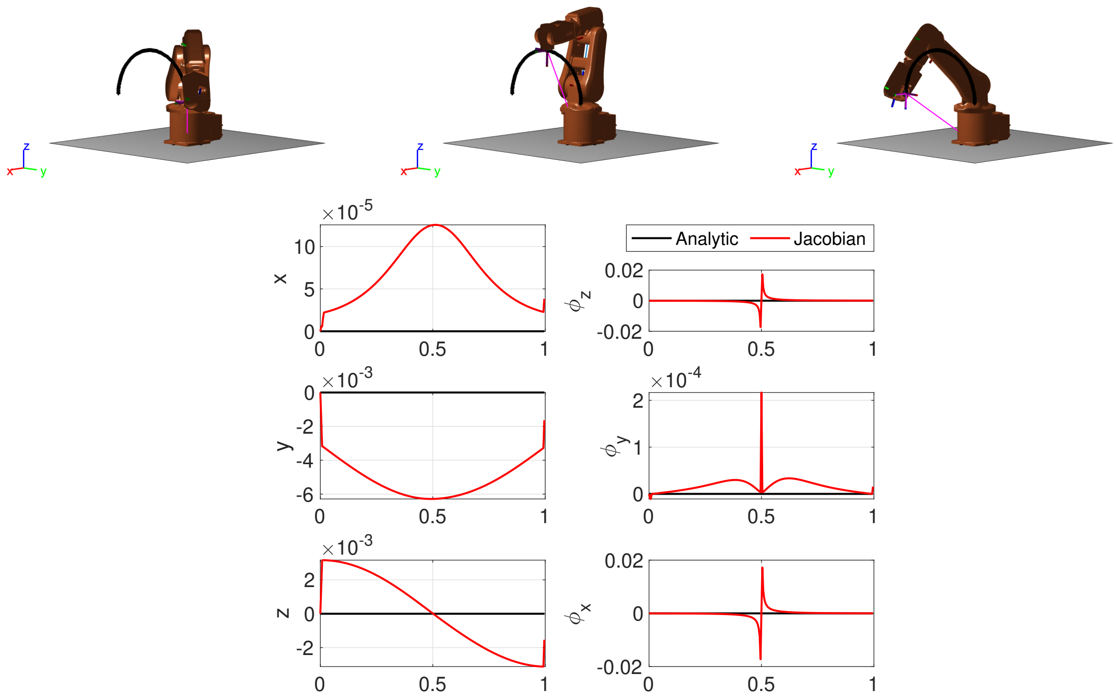

Figure 4: Snapshots from the IK sequence and error for each component for both methods.