Robot Dynamics Quiz 1

Prof. Marco Hutter
Teaching Assistants: Jan Carius, Ruben Grandia,
Jean Pierre Sleiman, Maria Vittoria Minniti

October 14, 2020

Duration: 1h 15min
Permitted Aids: The exam is open book, which means you can use the script, slides, exercises, etc; The use of internet (beside for licenses) is forbidden; no communication among students during the test.

1 Instructions

1. Download the ZIP file for quiz 1 from Piazza. Extract all contents of this file into a new folder and set MATLAB's[1] current path to this folder.

2. Run `init_workspace` in the Matlab command line

3. All problem files that you need to complete are located in the `problems` folder

4. Run `evaluate_problems` to check if your functions run. This script does not test for correctness. You will get 0 points if a function does not run (e.g., for syntax errors).

5. When the time is up, zip the entire folder and name it
   `ETHStudentID_StudentName.zip`
   Upload this zip-file through the following link
   `https://www.dropbox.com/request/JGp7ImPmEZzRDdrssfyy`.
   You will receive a confirmation of receipt.

6. If the previous step did not succeed, you can email your file to
   `robotdynamics@leggedrobotics.com`
   from your ETH email address with the subject line
   `[RobotDynamics] ETHStudentID - StudentName`

[1] Online version of MATLAB at `https://matlab.mathworks.com/`

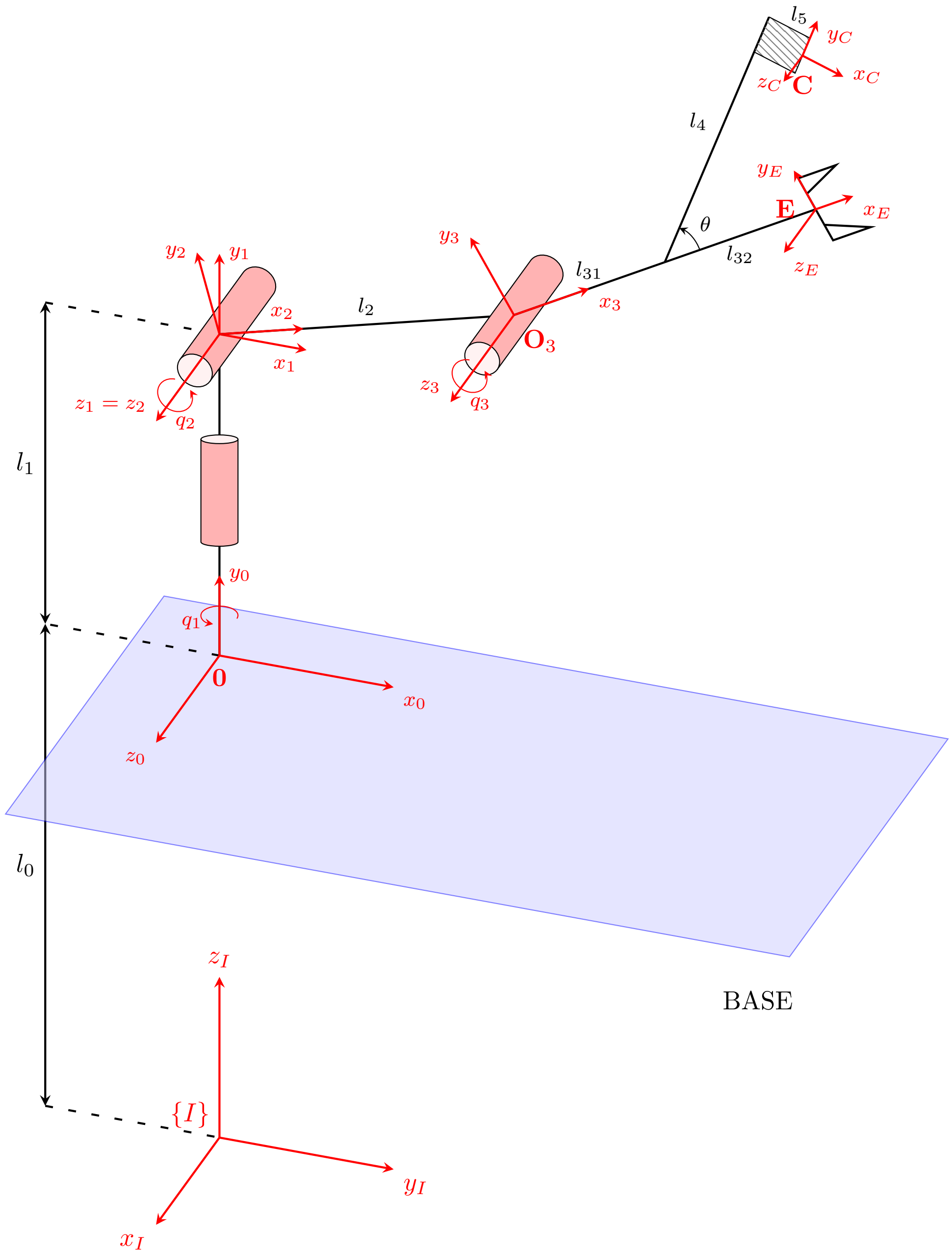

Figure 1: Schematic of a 3 degrees of freedom robotic arm attached to a fixed base. A camera is rigidly mounted on the last link of the arm.

2 Questions

In this quiz, you will model the forward, differential, and inverse kinematics of the robotic arm shown in Fig. 1. It is a 3 degrees of freedom arm connected to a **fixed** base.

Let $\{0\}$ be the base frame, which is displaced by l_0 from the inertial frame $\{I\}$ along the ${}_I z$ axis.
The arm is composed of three links. The reference frames attached to each link are denoted as $\{1\}, \{2\}, \{3\}$. The links' segments have lengths l_1, l_2, $l_{31} + l_{32}$.
Additionally, a camera is mounted on the last link of the arm. As shown in figure 1, the camera link is mounted at a constant angle θ around the axis z_3.

A visualization of the robot in the plane $x_1 y_1$ is also provided in figure 2.

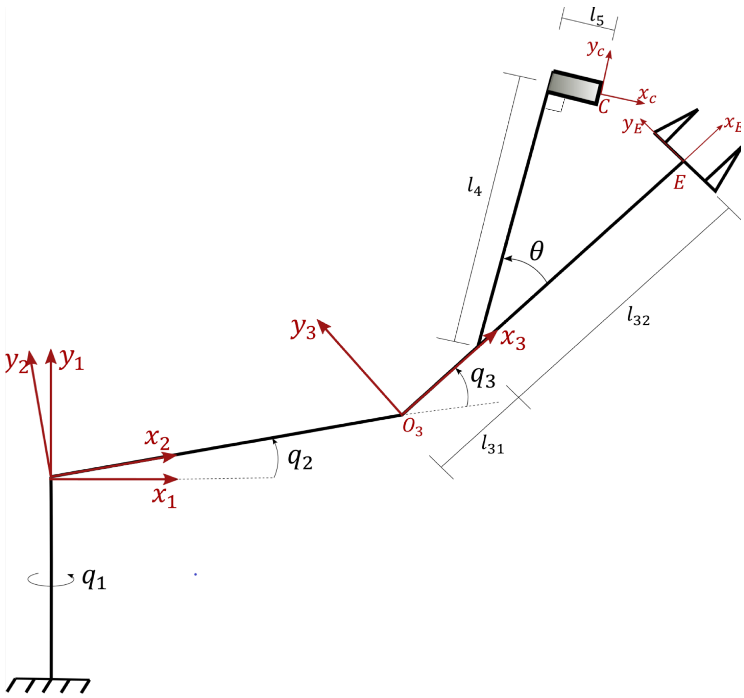

Figure 2: Planar visualization of the 3-DOF robotic arm

The generalized coordinates are defined as

$$\boldsymbol{q} = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^\top . \tag{1}$$

NOTE: the joint angles q_2 and q_3 are assumed to be zero when the axis x_2 and x_3 are parallel to the axis x_0 of the base frame.
In the following questions, all required parameters are passed to your functions in a structure called `params`. You can access it as follows:

```
l0 = params.l0;
l1 = params.l1;
l2 = params.l2;
l31 = params.l31;
l32 = params.l32;
l4 = params.l4;
l5 = params.l5;
theta = params.theta;
max_it = params.max_it;
lambda = params.lambda;
alpha = params.alpha;
```

Question 1. 6 P.

Let $\{E\}$ be the end-effector frame. Find the homogeneous transform between the inertial frame $\{I\}$ and the end-effector frame $\{E\}$, i.e., the matrix $\mathbf{T}_{IE}$ as a function of the generalized coordinates $\boldsymbol{q}$.
Hint: Try to find the transforms of subsequent frames first.

You should implement your solution in the function `jointToEndeffectorPose.m`

Question 2. 6 P.

Consider the difference between the geometric Jacobian $\mathbf{J}_{IC} \in \mathbb{R}^{6\times 3}$ of point $\mathbf{C}$, and the geometric Jacobian $\mathbf{J}_{IO_3} \in \mathbb{R}^{6\times 3}$ of point $\mathbf{O_3}$:

$$ {}_3\mathbf{J}_{O_3C} = {}_3\mathbf{J}_{IC} - {}_3\mathbf{J}_{IO_3}, \tag{2} $$

where all the terms are expressed in the same frame $\{3\}$. The Jacobian ${}_3\mathbf{J}_{O_3C} \in \mathbb{R}^{6\times 3}$ defines the following mapping:

$$ \begin{bmatrix} {}_3\boldsymbol{v}_{IC} - {}_3\boldsymbol{v}_{IO_3} \\ {}_3\boldsymbol{\omega}_{IC} - {}_3\boldsymbol{\omega}_{I3} \end{bmatrix} = {}_3\mathbf{J}_{O_3C}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3} $$

where ${}_3\boldsymbol{v}_{IC}, {}_3\boldsymbol{v}_{IO_3}, {}_3\boldsymbol{\omega}_{IC}, {}_3\boldsymbol{\omega}_{I3} \in \mathbb{R}^3$ are the linear and angular velocities of the frames $\{C\}$ and $\{3\}$, respectively. Compute the Jacobian ${}_3\mathbf{J}_{O_3C}$ in reference system $\{3\}$, using the following hints:

1. Note that

$$ \boldsymbol{v}_{IC} - \boldsymbol{v}_{IO_3} = \begin{bmatrix} \mathbf{n}_1 \times \mathbf{r}_{O_3C} & \mathbf{n}_2 \times \mathbf{r}_{O_3C} & \mathbf{n}_3 \times \mathbf{r}_{O_3C} \end{bmatrix} \dot{\mathbf{q}} \tag{4} $$

   where $\mathbf{n_1}, \mathbf{n_2}, \mathbf{n_3} \in \mathbb{R}^3$ are the rotational axis of the three joints, and $\mathbf{r}_{O_3C} \in \mathbb{R}^3$ is the position vector from $\mathbf{O_3}$ to $\mathbf{C}$.

2. The points $\mathbf{C}$ and $\mathbf{O_3}$ belong to the same rigid body.

3. The MATLAB function for cross product $\boldsymbol{a} \times \boldsymbol{b}$ is `cross(a,b)`.

You should implement your solution in the function `point3ToCameraGeometricJacobian.m`

Question 3. 3 P.

Given a camera position $\mathbf{r}^*_{IC} \in \mathbb{R}^3$ and a starting joint configuration $\mathbf{q_0}$, implement a MATLAB function which computes the joint angles $\mathbf{q}$ corresponding to the given camera position, using an iterative inverse kinematics algorithm.
For this question, we provide:

- the position vector ${}_I\mathbf{p}_{IC}$.

  You can call it with `jointToCameraPosition_solution(q, params)`.

- the position Jacobian ${}_I\mathbf{J}_{IC} \in \mathbb{R}^{3\times 3}$ of point $\mathbf{C}$.

  You can call it with `jointToPositionJacobian_solution(q, params)`.

- a function for calculating damped pseudo-inverses, as you have seen in the exercise: `pseudoInverseMat_solution(J, lambda)`.

You should implement your solution in the function `inverseKinematics.m`

Question 4. 3 P.

Assume now that the base frame $\{0\}$ can freely rotate and its rotation with respect to the inertial frame $\{I\}$ is described by a given quaternion $\mathbf{Q}_{I0}$. Write a MATLAB function to compute the rotation matrix $\mathbf{C}_{IC}$, that represents the orientation of the camera frame $\{C\}$ with respect to the inertial frame $\{I\}$.

For this question, we provide the transform $\mathbf{T}_{0C}$, which you can access with `T_0C_solution(q, params)`.

You should implement your solution in the function `cameraFrameOrientationWithBaseRotation.m`