Robot Dynamics Quiz 1

Prof. Marco Hutter
Teaching Assistants: Maria Vittoria Minniti, Mayank Mittal,
Koen Kramer, Farbod Farshidian, Jean Pierre Sleiman

October 20, 2021

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

1 Instructions

1. Download the ZIP file `RobotDynamics_Quiz1_2021.zip` from Moodle. 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. You can use helper functions. However, only the provided function-template files are used for grading. Implementations outside the provided templates will not be graded and receive 0 points.
6. When the time is up, zip the entire folder and name it
   `ETHStudentID_StudentName.zip`
   Submit this zip-file through Moodle under **Midterm Exam 1 Submission**. You should receive a confirmation email.
7. 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/`

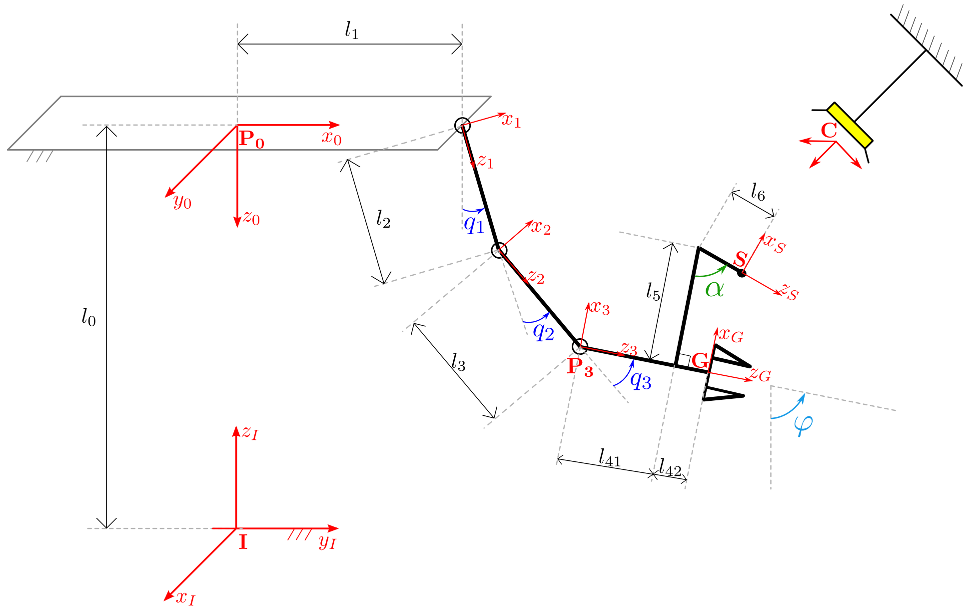

Figure 1: Schematic of a 3 degrees of freedom robotic arm attached to a fixed base. All joints rotate around the positive y_0 axis. The y axis of the frames $\{1\}, \{2\}, \{3\}$ is parallel to the y_0 axis.

2 Questions

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

The arm is composed of three links. The reference frames attached to each link are denoted as $\{1\}, \{2\}, \{3\}$. The links have lengths l_2, l_3, $l_{41} + l_{42}$.
As shown in Fig. 1, a sensor is rigidly mounted at point S on the last link of the arm, rotated of a constant angle α.
Additionally, an external camera is located at point C and fixed to the ceiling.
The generalized coordinates are defined as

$$\boldsymbol{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \in \mathbb{R}^3 \, . \tag{1}$$

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;
l3 = params.l3;
l41 = params.l41;
l42 = params.l42;
l5 = params.l5;
l6 = params.l6;
alpha = params.alpha;
```

Question 1. 6 P.

Let $\{S\}$ be the sensor frame, as indicated in Fig.1. Find the homogeneous transformation between the inertial frame $\{I\}$ and the sensor frame $\{S\}$, i.e., the matrix $\mathbf{T}_{IS}$ as a function of the generalized coordinates $\boldsymbol{q}$.
Hint: Note that a constant translation **and** a constant rotation are present between the inertial frame $\{I\}$ and the base frame $\{0\}$.

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

Question 2. 4 P.

Compute the position Jacobian ${}_0\mathbf{J}_P \in \mathbb{R}^{3\times3}$, that fulfills:

$$ {}_0\boldsymbol{v}_{03} = {}_0\mathbf{J}_P(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{2} $$

where ${}_0\boldsymbol{v}_{03} \in \mathbb{R}^3$ is the linear velocity of point P_3 with respect to the fixed point P_0, expressed in frame $\{0\}$.
You should implement your solution in the function `jointToPositionJacobian.m`

Question 3. 2 P.

Compute the rotation Jacobian ${}_I\mathbf{J}_R \in \mathbb{R}^{3\times3}$, that fulfills:

$$ {}_I\boldsymbol{\omega}_{I2} = {}_I\mathbf{J}_R(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{3} $$

where ${}_I\boldsymbol{\omega}_{I2} \in \mathbb{R}^3$ is the angular velocity of frame $\{2\}$ with respect to the inertial frame $\{I\}$, expressed in frame $\{I\}$.
You should implement your solution in the function `jointToRotationJacobian.m`

Question 4. 3 P.

Implement a kinematic trajectory controller for the robot arm. Your implementation should return the desired joint velocities $\dot{\boldsymbol{q}}^*$, that the robot needs to track to make the **gripper**, located at point G, follow a pre-defined reference trajectory.

We indicate with ${}_I\boldsymbol{p} \in \mathbb{R}^3$ and ${}_I\boldsymbol{w} \in \mathbb{R}^3$ the following vectors:

$$ {}_I\boldsymbol{p} = \begin{bmatrix} {}_Iy_G \\ {}_Iz_G \\ \varphi \end{bmatrix}, \qquad {}_I\boldsymbol{w} = \begin{bmatrix} {}_I\dot{y}_G \\ {}_I\dot{z}_G \\ \dot{\varphi} \end{bmatrix}, \tag{4} $$

where the angle φ is indicated in Fig. 1.

The desired vectors ${}_I\boldsymbol{p}_{des}$ and ${}_I\boldsymbol{w}_{des}$, as well as the current joint angles $\boldsymbol{q}$, are passed as inputs to your matlab function.
For this question, we provide:

- a function to calculate the current gripper position in the plane ${}_Iyz$:

$$ {}_I\boldsymbol{p}_{yz} = \begin{bmatrix} {}_Iy_G \\ {}_Iz_G \end{bmatrix} \in \mathbb{R}^2. \tag{5} $$

  You can call it with `jointTo2DGripperPosition_solution(q, params);`

- the analytical Jacobian $\mathbf{J}_A \in \mathbb{R}^{3\times3}$, that fulfills:

$$ {}_I\boldsymbol{w} = \mathbf{J}_A\dot{\boldsymbol{q}} \tag{6} $$

  You can call it with `jointToGripperAnalyticalJacobian_solution(q, params);`

- A function to compute the damped pseudo-inverse of a matrix `A`. You can call it with `pseudoInverseMat_solution(A, lambda)`.

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

Your implementation is judged based on how well the robot tracks a predefined trajectory.
A kinematics-simulator is implemented for you in the function `main_control_loop.m`, where the variable `use_solution` is set. If `use_solution` is set to 1, executing this function will show you how the solution should look like. If `use_solution` is set to 0, at each time-step the simulator will use the velocities returned by your own implementation.

Question 5. 3 P.

Let $\{C\}$ be the reference frame describing the pose of an external, fixed camera, as shown in Fig. 1. The orientation of $\{C\}$ with respect to the inertial frame $\{I\}$ is expressed via a ZYX Euler angles parametrization.

The inputs to your matlab function are the following:

- the Euler angles θ_z, θ_y, θ_x;
- the camera position ${}_I\boldsymbol{p}_{IC} \in \mathbb{R}^3$, expressed in the inertial frame;
- the homogeneous transformation $\mathbf{T}_{IG} \in \mathbb{R}^{4\times 4}$.

Compute the homogeneous transformation $\mathbf{T}_{CG}$.
You should implement your solution in the function `gripperToCameraPose.m`