Robot Dynamics Midterm 2

Prof. Marco Hutter
Teaching Assistants: Takahiro Miki, Joonho Lee,
Nikita Rudin, Victor Klemm, Clemens Schwarke

November 9, 2022

Duration: 1h 15min
Permitted Aids: Everything; no communication among students during the test

1 Instructions

1. Download the ZIP file `RobotDynamics_Quiz2_2022.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. When the time is up, zip the entire folder and name it
   `ETHStudentID_StudentName.zip`
   Submit this zip-file through Moodle under **Midterm Exam 2 Submission**. You should receive a confirmation email.

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`

7. **Important:**
   (a) Implementations outside the provided templates will not be graded and receive 0 points. No external libraries are allowed, except for MATLAB's Symbolic Math Toolbox.
   (b) Helper functions included in the `solutions/pcode` directory are used for simulating the robot. Using these functions in the solutions for other questions is prohibited and will receive 0 points.

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

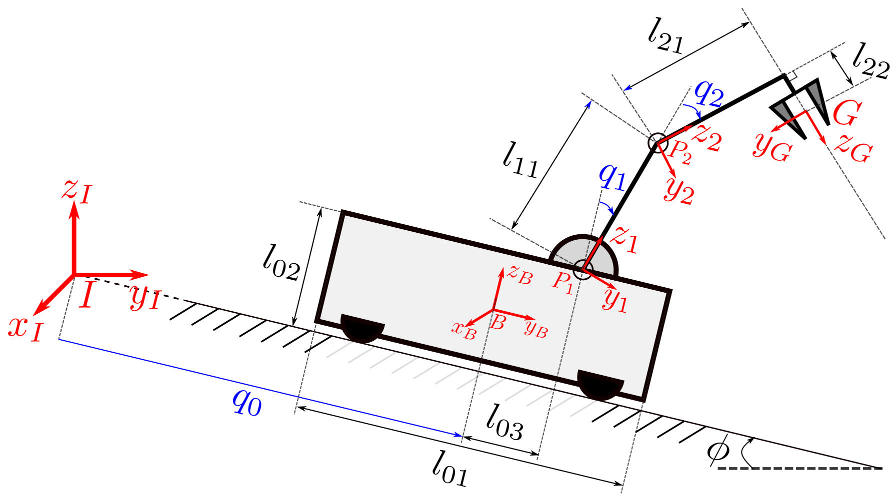

Figure 1: Schematic of a robot mobile manipulator with a two degrees of freedom robotic arm attached to a mobile base. The base can only move along the slope direction, while all joints on the arm can rotate around the positive x_I axis. The x axis of the frames $\{P_1\}, \{P_2\}$ is parallel to the x_I axis.

2 Questions

In this quiz, you will model the dynamics of the robotic manipulator shown in Fig. 1. It is a 2 Degrees-of-Freedom (DoF) arm connected to a mobile base.

The base can only move linearly along the slope with inclination angle ϕ. The reference frame attached to the base is denoted as $\{B\}$. The arm is composed of two links. The reference frames attached to each link are denoted as $\{P_1\}, \{P_2\}$. The frame attached to the gripper is denoted as $\{G\}$. *Note:* The transformation between $\{P_2\}$ and $\{G\}$ is fixed, i.e. there is a fixed 90 deg rotation between the gripper and the second link of the arm.

The generalized coordinates are defined as

$$\boldsymbol{q} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \end{bmatrix} \in \mathbb{R}^3 . \tag{1}$$

Clarification 1: The angles q_1, q_2, and ϕ are measured using right-hand thumb rule. For the state of the scene shown in Fig. 1, q_1, q_2, and ϕ would have negative values.

In the following questions, all required parameters are passed to your functions in a structure called `params`. You can access it as follows:

```
l01 = params.l01;
l02 = params.l02;
l03 = params.l03;
l11 = params.l11;
l21 = params.l21;
l22 = params.l22;
phi = params.slope_angle;
```

In the following questions, we have already provided the kinematics (transforms, Jacobians) and controller gains (k_P, k_D) for you. The variables stored as MATLAB *cells* may be accessed as follows:

```
m{k};        % mass of link k
k_r_ks{k};   % Position of com of link k in frame k
k_I_s{k};    % Rotational inertia of link k in frame k
R_Ik{k};     % Rotation matrix from frame k to I
I_J_rot{k};  % Rotational Jacobian of link k in I frame
etc..
```

Question 1. 3 P.

Calculate the mass matrix $\boldsymbol{M}(\boldsymbol{q})$, nonlinear terms $\boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ (Coriolis and centrifugal), and gravitational terms $\boldsymbol{g}(\boldsymbol{q})$. A helper function, `dAdt.m`, is available in the `utils` folder, to compute the time derivative of a matrix.

Implement your solution in `Q1_generate_eom.m`.

Hint: Do you need to take the slope angle explicitly into account when formulating the dynamics, or is it already included in the provided quantities?

Question 2. 2 P.

Implement a forward dynamics simulator that computes the joint accelerations $\ddot{\boldsymbol{q}}$ and integrates them to get $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$. You should implement the calculation of $\ddot{\boldsymbol{q}}$, given $\boldsymbol{\tau}$, the input torque for each joint.

Use the mass matrix, non-linear terms and gravitational terms obtained from `M_fun_solution(q)`, `b_fun_solution(q, dq)` and `g_fun_solution(q)`.

You should implement your solution in `Q2_forward_simulator.m`.
You can check your implementation by running `simulate_forward_dynamics.m`

Question 3. 2 P.

Implement a joint-level PD controller that compensates for the gravitational terms and tracks desired joint positions and velocities. Calculate $\boldsymbol{\tau}$, the control torque for each joint.

Current joint positions $\boldsymbol{q}$ and joint velocities $\dot{\boldsymbol{q}}$, as well as desired joint positions $\boldsymbol{q}^d$ and desired joint velocities $\dot{\boldsymbol{q}}^d$ are given to the controller as input arguments to the MATLAB file. You should obtain the gravitational terms in this question through the provided `g_fun_solution(q)`. Use the PD gains provided in the parameters (`params`).

Implement your solution in `Q3_gravity_compensation.m`. A simulation of your implemented controller (or the solution) is available in `simulate_gravity_compensation.m`.

Question 4. 3 P.

In this question, we assume that there's a vertical wall in front of the robot and we want to wipe the wall with the gripper. Implement a controller that uses a task-space inverse dynamics algorithm, i.e. a controller which compensates for the entire dynamics and tracks a desired motion in the task-space. Calculate $\boldsymbol{\tau}$, the control torque for each joint.

The inputs to this controller are the desired pose of the gripper, desired force applied on the wall as well as the current joint position $\boldsymbol{q}$ and joint velocities $\dot{\boldsymbol{q}}$. The desired pose and force have the following components (all expressed in the inertial frame):

- desired gripper position ${}_I r_{IG}^d \in \mathbb{R}^3$
- desired gripper velocity ${}_I v_G^d \in \mathbb{R}^3$
- desired gripper orientation $C_{IG}^d \in \mathbb{R}^{3\times 3}$
- desired y axis force on the wall ${}_I F_{G_y}^d \in \mathbb{R}$

Implement a controller that computes the torques necessary for following the desired motion of the end-effector in task-space as well as a force applied on the wall. The PD gains are provided in the parameters (`params`). Use the mass matrix, non-linear terms and gravitational terms obtained from `M_fun_solution(q)`, `b_fun_solution(q, dq)` and `g_fun_solution(q)`.

Implement your solution in `Q4_task_space_control.m`. A simulation of your implemented controller (or the solution) is available in `simulate_wall.m`.

Hint: Define and use corresponding selection matrices to control the force in the y direction and control the pose of the gripper. The gripper can move in all directions except the y direction, where it applies a force.