Haptic Human-Robot Interfaces: Lab 2

Assistants : *Giulia Ramella (giulia.ramella@epfl.ch)*
Mouhamed Zorkot (mouhamed.zorkot@epfl.ch)
Jonathan Louis Muheim (jonathan.muheim@epfl.ch)
Aiden Xu (xiangyu.xu@epfl.ch)

Lab goals

- Simulate the dynamics of the haptic paddle using MATLAB/Simulink.
- Compare the simulation results with the measurements from the actual haptic paddle.
- Tune a PID controller to reach a specified performance.
- Explore how the three PID control parameters influence the step response.

1 Grading

This lab will be graded. A report should be submitted on Moodle, no later than one week after the last session of this lab.
The report should contain at least the following elements:

- The equation of motion of the paddle based on which you built the Simulink model (also specify how the equivalent damping and inertia terms are calculated)
- A screenshot of the Simulink paddle model (and the contents of the subsystems).
- Output plots of the simulations (the nonlinear effect of gravity, the drifting of the position out of the normal range of motion of the paddle when apply a large motor torque).
- Comparative plots between simulation and implementation on the paddle. Comment.
- An explanation of the strategy you used to tune the regulator (starting point, incrementation of the gains, etc.).
- Using the Hall Sensor, plot 3 cases of PID implementation. Comment.
- Using the Incremental Encoder Sensor, plot 3 cases of PID implementation. Comment.
- Comment the difference between the results obtained with the two position sensors.
- Is filtering useful? Would you rather filter the position or the velocity? How does it affect the limits of the gains?
- For the case of the Haptic Paddle, does the Ziegler-Nichols tuning method result in a performant regulator?

2 Lab instructions

2.1 Basic identification

- Estimate roughly the dry friction of your paddle, by increasing gradually the motor torque, until the paddle moves. Try this in both directions.

2.2 Simulation with Simulink

The goal of this part is to build a Simulink model of the paddle, and use it to safely tune a position regulator.

- Download the Simulink project archive on Moodle, and unzip it.
- Start MATLAB, and set the extracted folder as the current MATLAB directory. This archive contains the following files:
  - main.m: this MATLAB script should set the simulation parameters, then start the Simulink simulation, and finally plot the results. Calculation of some parameters is not complete, and the plotting part is not implemented.
  - HapticPaddle.mdl: this is a placeholder for Simulink model of the paddle.
- Start Simulink by typing the "simulink" command, then click "Open…" and select the file "HapticPaddle.mdl". If you are using an old version of Simulink, you can also rename the HapticPaddle.mdl.r2016b to HapticPaddle.mdl and use that one instead.
- Make a Simulink model of the paddle. From the motor torque, it should allow to obtain the paddle angle, speed and acceleration over time.
- Open the script main.m which includes the definitions of constant parameters (refer to Figure 2 at the end of this document for a mechanical schematic showing the parameters). Complete the lines at the end of this script marked as "TO BE COMPLETED".
- Run the script to load the parameters into the workspace and start the simulation. After the constants have been loaded into the MATLAB workspace, you can also directly use the "Run" button in Simulink to restart the simulation.
- Inspect the Simulink scope plots, and check that they are consistent.
- Extend the Simulink model by adding a PID regulator, in order to control the position. To make the block diagram tidy, you can create one subsystem block for the PID, and one for the paddle model.
- Use the variables generated in your MATLAB workspace to generate a plot of the target position and the actual paddle position, over time.

2.3 Position controller implementation on the board firmware

On the board firmware, implement a position controller using a PID regulator. From the measured position of the paddle, its role is to compute the motor torque to reach the given set point. It should have the following features:

- An input filter to smooth the actual position. This is necessary to avoid having too much noise amplified by the derivative part of the PID.
- The regulator should be disabled by default when the board is starting, and it should be possible to enable/disable it from a variable of the remote-control interface.
- The ability to set the target position from the remote-control interface.
- The possibility to use the encoder position, or the Hall-effect sensor position.

2.4 PID tuning with the actual paddle

The goal of this part is to optimize the PID coefficients, in order to meet the required dynamic performance.

- Tune Kp, Ki, Kd such that the requirements for a 10° step response are met:
  - Maximum overshoot: 25% (2.5°).
  - Maximum settling time to reach an error band of ±0.2°: 350 ms.
  - Maximum rise time: 100 ms.
- Record the target and actual (measured) positions, to create a MATLAB plot, and check the performance.
- Compare the obtained performances using the encoder and the Hall-effect sensor.
- Does the behavior of the paddle correspond the one obtained by the simulation?
- Apply the Ziegler-Nichols method to compute new gains for the PID. Comment on the performance of these new gains, with respect to the previous hand-tuned gains.

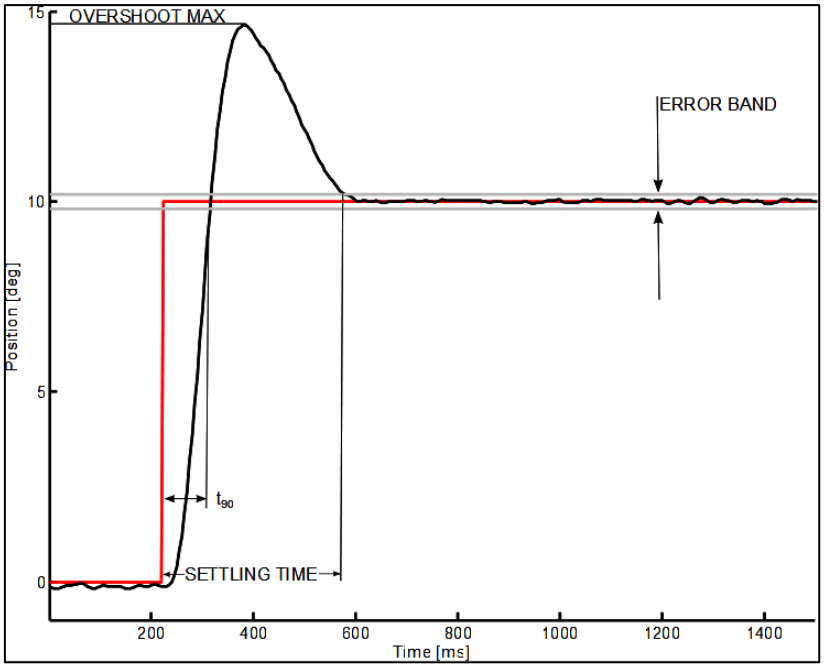

Figure 1: definition of the performance metrics for a step test

3 Appendix

3.1 Paddle model

This is a simplified mechanical schematic of the paddle, with the different parameters marked on it.

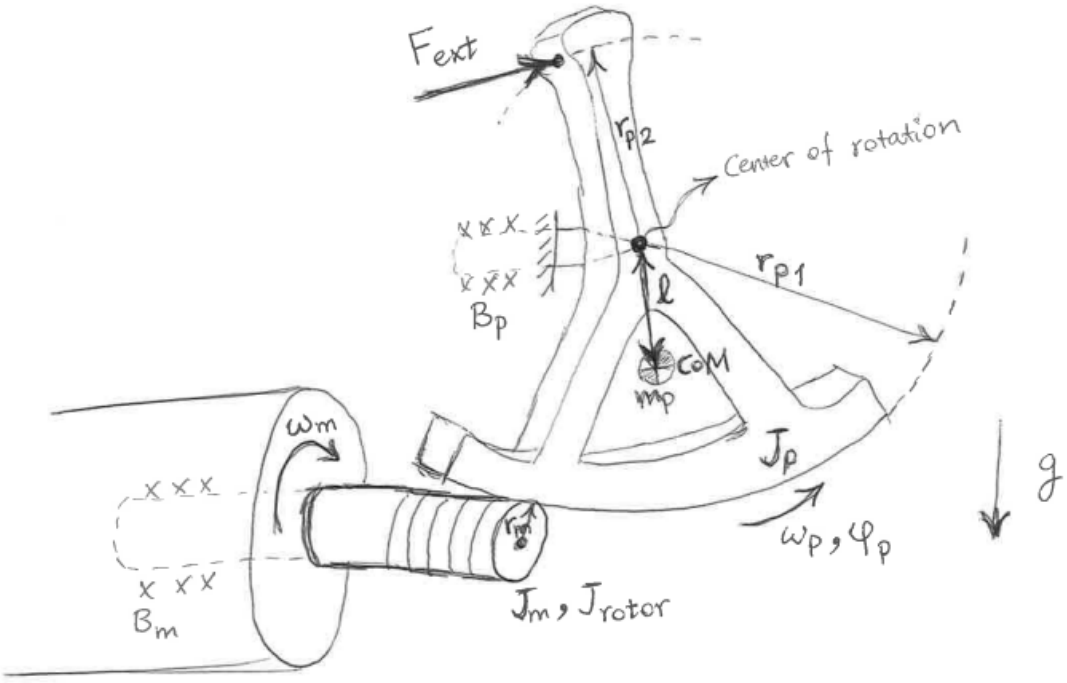

Figure 2: definition of the symbols used in the main.m file