ECE 380 Analog Control Systems
Gennaro Notomista

PROJECT DESCRIPTION

1 Introduction

Mobile robotic platforms are becoming increasingly important in a variety of applications, including warehouse automation, medical assistance, cleaning services, agricultural robotics, and environmental monitoring, to cite a few.

1.1 Mobile robot control

Prototypical mobile robot tasks consist in visiting a sequence of locations in space. Once a path has been planned which passes through these locations, a controller has to be designed to make the robot follow this path. In this project we consider the problem of designing a controller to drive a ground mobile robot to a desired location.

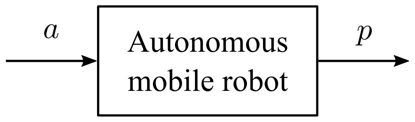

Figure 1: Block diagram of the considered ground mobile robot. The control input of the robot is its acceleration vector, $a \in \mathbb{R}^2$, and we have access to a measurement of its position $p \in \mathbb{R}^2$.

The block-diagram model of the considered mobile robotic system is given in Fig. 1. The control input to the system is the robot acceleration vector $u = [u_1, u_2]^T = [a_x, a_y]^T = a \in \mathbb{R}^2$, where a_x and a_y are the accelerations of the robot along the directions of the axes of the reference frame defined on the plane where the robot moves. The output of the system is the robot position vector $y = [y_1, y_2]^T = [p_x, p_y]^T = p \in \mathbb{R}^2$, where p_x and p_y are the components of the position of the robot along the axes of the reference frame. *Driving the robot to a desired fixed location corresponds to regulating the output of the system to a given constant reference value.*

The mathematical model of the system is not known but it is possible to probe the system by supplying an acceleration input signal and measuring the corresponding position output signal. This way, one can build a mathematical model of the system to be used to design a controller that fulfills the desired specifications.

1.2 Code

The code provided in the file `system_model.py` contains the Python class `SystemModel` that implements the dynamics of the mobile robot. This class allows us to faithfully simulate the motion of the robot and obtain the output response to a given input signal. The latter can be done by calling the class method `sim` as follows:

```
output_signal = sm.sim(input_signal),
```

where `sm` is an object of the class `SystemModel`, and `input_signal` and `output_signal` are the input and output signals, respectively, represented as `numpy` arrays whose shape is `(L, 2)`. The variable `L` is the length of the signal measured in number of time steps, where one time step lasts $\Delta t = 0.001$s. This way, if a signal lasts for 1s, the length of the `numpy` array representing the signal will be 1000.

The class `SystemModel` has one another method, `step`, which can be used as follows to obtain the measured output vector $\bar{y}$ at time $t + \Delta t$ in response to an input signal $u(\tau) = \bar{u}, \forall \tau \in [t, t + \Delta t]$:

```
y_bar = sm.step(u_bar),
```

where `u_bar` and `y_bar` are the input and measured output vectors, represented as `numpy` arrays whose shape is `(2,)`.

2 Project tasks

Task 1 consists of collecting and processing input-output data of the robot to control.

2.1 Task 1

Complete the items described below:

- **Item 1** Define constant input signals u_1, u_2 of amplitude 1 over a time horizon of 15s. Generate the corresponding output signals y_1, y_2.
- **Item 2** Plot the signals u_1, u_2, y_1, y_2.
- **Item 3** Choose a cut-off frequency $\omega_c = 10$ to filter out noise from the signals y_1, y_2 and design a third order low-pass filter with real and coincident poles and cut-off frequency ω_c.
- **Item 4** Filter the signals y_1, y_2 using the designed low-pass filter. Plot the filtered signals.

2.2 Task 2

Assume the robot can be modeled using the following transfer functions:

$$\begin{aligned} G_{11}(s) = G_{22}(s) &= \frac{0.53}{s^2 + 1.05s + 0.005} \\ G_{12}(s) = G_{21}(s) &= 0, \end{aligned} \tag{1}$$

where $G_{ij}(s)$ is defined by the following relation:

$$Y_i(s) = G_{ij}(s)U_j(s), \quad U_k(s) = 0 \ \forall k \neq j.$$

Complete the items described below:

- **Item 1** Choose $x_1 = [y_1, \dot{y}_1]^T \in \mathbb{R}^2$ and find a state space representation of $G_{11}(s)$:

$$\begin{cases} \dot{x}_1 = A_1 x_1 + B_1 u_1 \\ y_1 = C_1 x_1 + D_1 u_1. \end{cases} \tag{2}$$

  Analogously, choose $x_2 = [y_2, \dot{y}_2]^T \in \mathbb{R}^2$ and obtain a state space representation of $G_{22}(s)$.

- **Item 2** Using the state space models found in **Item 1**, compute state feedback controllers $u_1 = -K_1 x_1$ and $u_2 = -K_2 x_2$ such that the closed-loop systems satisfy the following specifications:
  - (i) Overshoot is less than 2% [1]
  - (ii) The 2% settling time is less than 4s [2]
- **Item 3** Implement the state feedback robot controller in the method `calculate_acceleration` of the `Controller` class in the provided `controller.py` file. *Hint: $\dot{y}_1$ and $\dot{y}_2$ required to evaluate the controller may be approximated using finite differences.*
- **Item 4** Simulate the controlled autonomous mobile robot for a time horizon of 30 seconds and plot the signals y_1, y_2, u_1, u_2 over time. The provided `main.py` file can serve as a starting point.

[1] $O.S.\% = 100e^{-\zeta\pi/\sqrt{1-\zeta^2}}$.

[2] $T_s^{2\%} \approx \frac{4}{\zeta\omega_n}$.

3 Deliverables and grading

The deliverables for each task consist of:

- A report containing the required plots and explaining the details of the developed solution; the report should be in the form of a `.pdf` file, of maximum 2 pages, and written using the LaTeX or Microsoft Word IEEE conference proceedings template available here
- A Python script (`.py` file) containing the code to solve the task

Important notes

- If the format of a project task submission is not as above, the project task will not be graded.
- The report should contain the explanation of the steps followed to solve each item of a project task, as well as the reference to the part of the code that solves each item, otherwise the corresponding item will not be graded.

4 Logistic details

- The project work is carried out in groups, same groups of the lab.
- Project submissions, one per group, will be handled via LEARN.
- Multiple submissions will be accepted via LEARN before the deadline, but only the most recent one will be available for grading.