# Homework 9

EECS/BioE C106A/206A
Introduction to Robotics

Due: November 10, 2020

*Note: This problem set includes a programming component that will require you to have scipy installed. You will be best off completing this assignment in a Python 3 environment.*

**Problem 1. Control of a Fully-Actuated Cart-Pole System**

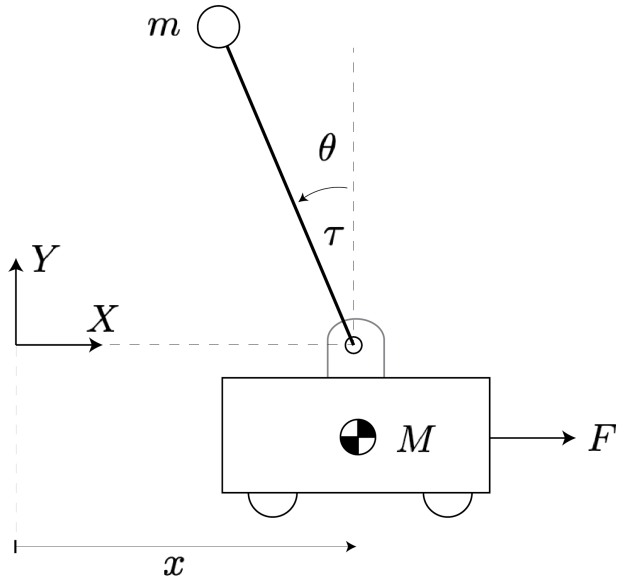


Figure 1: A fully actuated cart-pole system.

Figure 1 shows a cart of mass $M$ with an inverted pendulum (rod) balancing on top of it. The pendulum is a massless rod of length $L$ with a point mass $m$ on top of it. The displacement of the center of the cart from the spatial origin in the horizontal direction is $x$, as notated, and the angle formed between the rod and the vertical is $\theta$. The cart has mounted in its drive-train a motor that can apply an external force $F$ in the horizontal direction. Additionally, a motor is mounted in the pivot of the rod that applies a torque $\tau$.

In this problem, you will design and implement a computed-torque control law to regulate the cart-pole system so that the cart stabilizes at the point $x = 0$ and the rod is balanced perfectly vertically. We will do this by controlling the force $F$ and torque $\tau$. In other words, our control input will be $u(t) = (F(t), \tau(t))^\top$.

(a) Use the method of Lagrange to show that the dynamics of the cart-pole system can be written in the form

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + N(q) = u \tag{1}$$

where $q = (x, \theta)^\top$ and

$$M(q) = \begin{bmatrix} M + m & -mL\cos\theta \\ -mL\cos\theta & mL^2 \end{bmatrix}$$

$$C(q, \dot{q}) = \begin{bmatrix} 0 & mL\dot{\theta}\sin\theta \\ 0 & 0 \end{bmatrix}$$

$$N(q) = \begin{bmatrix} 0 \\ -mgL\sin\theta \end{bmatrix}$$

(b) Complete the implementation of the inertia matrix, Coriolis matrix and gravity vectors in the provided python file.

(c) In lecture, we developed a way to make the generalized coordinates $q$ track a desired trajectory $q_d(t)$. Given that we wish to stabilize the cart so that it is at $x = 0$ with the rod balancing vertically, write down what the desired trajectory $q_d(t)$ should be.

(d) Now, let's define our control law. We will use a computed torque controller, as described in section 4.5.2 of MLS. Write down a computed torque control law for $u$ as a function of $(q, \dot{q})$. You may use the tuneable positive gain matrices $K_p = \text{diag}(\alpha_1, \alpha_2)$ and $K_v = \text{diag}(\beta_1, \beta_2)$.

(e) Implement your answer to the previous part in the python file.

(f) Now let's examine the performance of our controller by simulating it. Read the provided hw9.py file for instructions on running it. The file initializes the system with default constants of $m = M = L = 1$. Use the initial condition $q(0) = (x(0), \theta(0)) = (1.0, 0.2)$. By default we will set $\dot{q}(0) = 0$ (i.e. the system starts off at rest in the initial state). Simulate the response of the system for the following sets of controller gains:

| $\alpha_1$ | $\alpha_2$ | $\beta_1$ | $\beta_2$ |
|---|---|---|---|
| 0.1 | 0.1 | 0.0 | 0.0 |
| 0.1 | 0.1 | 0.1 | 0.1 |
| 0.5 | 0.5 | 0.1 | 0.1 |
| 0.1 | 0.1 | 0.5 | 0.5 |
| 0.5 | 0.5 | 0.5 | 0.5 |

In answering each of the following questions, you should use plots to support your answer wherever appropriate.

(i) Are each of the 5 sets of gains above able to successfully stabilize the system to the desired point in the 100 second time-period?

(ii) What is the effect of introducing a nonzero $K_v$?

(iii) Comment on the effect of increasing or decreasing $K_p$ and the effect of increasing or decreasing $K_v$.

(iv) Using the above gains as a starting point, find a set of gains that provide good performance. Here, "good" means that the state is stabilized to the desired point quickly with minimal overshoot or oscillations. Provide a plot to demonstrate the performance of your chosen gains.