Lab 5: DC Motors, H-Bridges, and Encoders

Overview

In this lab, you will learn to control a DC motor using an Arduino. You will start with open-loop control and eventually move on to feedback control. You will use an H-bridge to allow an Arduino to control the speed and direction of the motor while power is being supplied from an external source.

Video Overview

Here is a video summarizing the expectations of the lab and showing what successful Arduino and Python code will do:

- video: https://gvsu.ensemblevideo.com/Watch/Ht6k3Z4X

Pre-Lab Reading

In addition to watching the video overview, you must read up on encoders and H-bridges before the lab:

- Encoders: https://en.wikipedia.org/wiki/Rotary_encoder
- H-Bridges: https://tronixlabs.com.au/news/tutorial-l298n-dual-motor-controller-module-2a-and-arduino/

Motor Details

Here is a link to the description of the motor including the pinout of the wire connection:
https://www.pololu.com/product/2823

Open-Loop Testing Requirements

For the open-loop portion of this lab, you must do the following:

- use a timer interrupt to ensure control actions are taken at precise intervals
  - prove that you timer interrupt is working correctly and that you can run it at 100, 200, 300, or 500 Hz
  - your timer interrupt should help control how long your pulse input stays "on" and how long the test runs
- use a pin interrupt to respond to the encoder signals
- write Arduino code to decode the encoder signals
  - verify that your code is correctly finding the rotational position
- write a function in your Arduino code that takes an integer input that can be positive or negative and commands the motor to move at the appropriate speed and in the appropriate direction based on the sign and magnitude of the input
  - i.e. a positive input should mean positive rotation and a negative input should mean negative rotation
  - the sign of your function should correspond to the sign of your encoder code so that positive voltage leads to positive change in θ
- write Arduino code that requests two inputs from the user (pulse amplitude and pulse width) and then runs a test for a specified number of time steps
  - your code must print real-time data to the serial monitor
  - your code must stop printing after a specified number of time steps
    * constantly printing data to the serial monitor will cause problems and confusion
- use the `pyserial` module to run tests and capture data using Python
- write a Python function that takes pulse amplitude and width as inputs and then runs a test and plots the results
- your Python code must also be able save data to .csv files
- run many open-loop tests and come up with a relationship between pulse amplitude and width and the corresponding change in rotational position

Recommended Steps

- install `pyserial` using the command `pip install pyserial`
  - use either `powershell` (windows) or the terminal (mac/linux)

- this lab can be broken into different pieces and each piece can be worked on independently:
  - commanding the motor to move using the H-bridge and writing the corresponding Arduino function
  - getting timer interrupts working correctly
  - decoding the encoder signals and finding $\theta(t)$
  - getting serial working via python
    - set up your Arduino to ask for the two inputs and then stream fake or simulated data for now

Open-Loop Report Specifications

You will turn in one report that covers both the open-loop and closed-loop portions of the lab. Here are the expectations for the open-loop portion:

- Prove that you are using timer interrupts correctly and that you can control the motor and update frequencies of 100, 200, 300, and 500 Hz.
  - include the relevant code in your report along with your supporting proof
- Include your Arduino code for handling the encoder signals and prove that it is working correctly.
- Prove that you can command the motor with correct sign/direction and speed.
- Demonstrate that your Arduino code requests pulse amplitude and width from the user and runs a test for a specified number of time steps with the correct pulse width corresponding to the time the pulse is "on".
- Show that you can use python to run tests via the serial module
- Show graphs of some open-loop pulse tests
- Shows graphs or other summary information relating pulse width and amplitude to change in θ and discuss whatever relationships you have found.