Project
ME 5245, Spring 2022

Due: April 29, 2022, email ZIP file to Sipahi

LAST NAME: **FIRST NAME:**

Instructions

Please email your project to Prof. Rifat Sipahi and Noah Ossanna as a single ZIP file (MAIN TASKS) with an organized folder structure including a PDF of the report. If you have solved any of the Bonus tasks, please make sure you clearly mark your PDF report with headings so we don't miss grading them. Bonus tasks are optional.

Please incorporate all relevant plots, equations, as well as source codes. Solutions without proper justification will not earn credits.

This is an **individual-based project**. Each student will work **on their own** to complete their tasks. Students are permitted to discuss the project with each other however each student must produce their own, original work. Any cheating detected in student submissions will be handled based on NU policies.

Project Description

In this project, students will develop mathematical models and simulations for a DC motor control system, and integrate hardware and software to realize this control system. In the modeling and simulation part, the student should produce simulations in Simscape/Simulink corresponding to the physical system at hand. In parallel, the student should build the hardware system by wiring an encoder, the DC motor, an H-bridge together, and create the software that will interface with this hardware to perform various control actions, specifically, controlling the position and speed of the motor.

Provided Hardware

Each student is provided the following hardware components:

- 1 Arduino Uno R3 and USB cable (`https://docs.arduino.cc/hardware/uno-rev3`)
- 1 Pololu 172:1 Metal Gearmotor with 48 CPR Encoder (`https://www.pololu.com/product/4828`)
- 1 Dual H-Bridge Motor Driver - L293D (`https://www.ti.com/lit/ds/symlink/l293.pdf`)
- 5 AA batteries
- 1 DC jack adapter
- 1 Battery holder

- 1 Potentiometer
- 1 Breadboard
- Wires

If possible please do not solder any components. If soldering is a must please consult with the instructor and Noah before soldering components.

After the project is completed and documented, these components should be un-assembled and returned to Noah in a Ziplock. Students not returning their equipment on time will receive an Incomplete grade.

Necessary Software

The student will develop software in both Arduino and in MATLAB. Specifically:

(a) Arduino coding. Please find examples at this link `https://www.arduino.cc/en/Tutorial/HomePage`. If you participated in the Arduino lab, you already have a head start. Further information about Arduino coding will be provided in an upcoming lab and also can be found here `https://www.arduino.cc/reference/en/`. However students are encouraged to get started on their own, as soon as possible.

(b) Simulink with real time target. You will build the control algorithm in Simulink and target it to Arduino. This will by-pass the need to write an Arduino code. In Canvas Lecture 19, we have a detailed demo that presents how real time targeting can be done. In an upcoming Lab 4, Noah will demonstrate real time targetting as well. More information about Simulink real time Arduino targeting is available here `https://www.mathworks.com/help/supportpkg/arduino/run-on-target-hardware.html`.

Control Objectives

Students will demonstrate motor speed control with Arduino coding and motor position control in Simulink using Real Time Target. In addition, the students will perform Simulink/Simscape simulations to predict the motor speed/position control problems, and compare the simulations against the experiments. Students whose Simulink/Simscape simulations well predict the experimental results will receive BONUS points.

(a) **Position control in Simulink using Real Time Target:** In this case, potentiometer position will set the reference position of the rotor shaft. When the potentiometer is moved from one position to another, the rotor shaft should move from one position to another (forward and backward). In this setting, potentiometer position must be read by the Arduino and is set as a reference signal. The motor encoder should be wired to the Arduino to help calculate rotor position in real time. Reference signal should then be compared to encoder readings, and the difference between them (position tracking) error must be multiplied by a proportional controller gain K_p. The arising signal is the command to the motor, but must be first mapped to an appropriate PWM value, which should then be sent to the H-bridge to drive the motor.

(b) **Speed control with Arduino coding:** In essence this experiment is similar to that of position control except that now the potentiometer position determines a reference speed for the rotor and the encoder position readings must be utilized to interpret the speed of the rotor. Then one can compute in the software the speed error as the difference between the reference speed and rotor speed, and feed the error information to a proportional controller gain K_p to compute the motor commands.

Main Tasks - Submit via email to Prof. Sipahi and Noah

Successful completion of the following tasks will equate to 100% of project grade. Except the first task, which does not require software, each task should be performed using software as described above. Items 1 - 4 below involve basic setup tasks that are required for subsequent speed and position control. Items 5 - 6 represent the control objectives of this project which build upon 1 - 4. Additional tasks for BONUS points are provided in the next section.

1. Provide the complete circuit diagram with the Arduino hardware, encoder, potentiometer, H-bridge and the DC motor. Encoder should be wired in quadrature mode.
2. Acquire the encoder readings to interpret rotor displacement in Simulink and rotor speed in Arduino software. Based on encoder resolution, calculate how much position and speed error you expect.
3. Show in Simulink software that, for different PWM duty cycles, motor position grows faster in time. Based on rotor speed data obtained in Arduino software, provide motor speed curves and using these curves, estimate motor transfer function, specifically motor DC-gain and time constant.
4. Show that you can acquire the potentiometer signal into Arduino software and Simulink.
5. Develop your Simulink to create the position control system described as Control Objective (a). Show that the motor does position tracking, and compare the experimental results against Simulink/Simscape simulations.
6. Develop your Arduino software to create the speed control system described as Control Objective (b). Show that the motor does speed tracking, and compare the experimental results against Simulink/Simscape simulations.

If your simulations predict well the experimental results, you will earn BONUS points under the Main Tasks section. Convincing evidence of this needs to be provided in the report. Please clearly report this in your PDF document.

Bonus Tasks - Submit along with Main Tasks ZIP file

The below tasks are optional and if completed will earn bonus points. The Student student should use either software option (Arduino or real time target) for each of the tasks depending on which they used for the main deliverables.

- Please obtain the FFT of the rotor speed data in a Matlab code and show which frequency components are dominant in this data.
- In the speed control task, please design a low-pass filter in the Arduino code and use this filter to filter the speed data. Use then the filtered speed data for comparison with reference speed. Compare the controlled speed of the motor with that obtained under Main tasks under the same conditions (but without the filter).
- In your Simulink/Simscape motor model, incorporate noise into rotor shaft position and use this noisy signal to perform position control. Using the PID tuner available in Simulink next, design a PID controller for the position control problem.
- In Simulink real-time target mode, please insert the PID designed from previous step into the position control system and test the performance of this PID controller in regulating the motor position.

If your time simulations predict well the experimental results, you will earn ADDITIONAL BONUS points. Convincing evidence of this needs to be provided in the report.

Important Points to Consider

Please meet Noah as soon as possible to pick up your supplies. If you are a part time student and would prefer that we mail you the supplies, please let us know.

You will realize that you will need to run a number of tests on the experimental system to identify certain parameters and/or figure out the proper scaling between various physical parameters. Please make sure you run several simple tests, diagnosis to make sure your system operates as it should. Please keep in mind that Arduino will have a certain sampling rate and this rate will influence how well you can control your system. Your sample time will also influence how quickly you acquire data and act on it. Please make sure you record all critical parameters of your system on a table, chart. If you need access to an oscilloscope or a multimeter please contact Noah.

As a starter, students should create a list of detailed tasks. Several tasks can be performed in parallel, and at certain milestones, these tasks can be merged into each other. For example, while figuring out the wiring diagram, one can also do simple testing with input/output signals (e.g., using a potentiometer) with Arduino, and build the motor model, the controller, etc. in Simulink/Simscape. Once the wiring diagram is confirmed, the student will wire the hardware and this can be integrated with basic Arduino coding. Majority of the Simulink motor model can be worked in parallel and can receive some information from the experimental work, so as to align the simulations with the experiments. These simulations can also put light on how to control the actual motor, e.g., what controller gain is needed for the experiment.

To stay on track, I advise each student to work on their wiring diagram and aim to complete it with Noah's guidance within a WEEK. In parallel the student should run the Arduino code with simple signals (labs cover some aspects), and also start building the Simulink/Simscape motor model (some of this model was built in class). From Lecture 19, with real time target demo, the student will quickly adopt that knowledge and work on integrating software with hardware.

Notice that this is a learning experience and you can think of it like a mini Master's Project. Such projects require continuous communication with other engineers (in this case, Sipahi and Noah), and Sipahi and Noah will in some sense collaborate with each student. On the other hand, it is important to note that it is the student's responsibility to be proactive to ask for help and guidance. It is also student's responsibility to be resourceful, look for solutions, and bring ideas, suggestions to meetings with Sipahi and Yijie. Students are allowed to discuss with each other as to how they are approaching their projects.

Remember that project management is very important so make sure you create a schedule, a list of tasks, and keep updating this document as you go. Make sure you do careful data logging, labeling, etc. It is better to develop your final report as you go. If you do not keep good track of your data and results, it may be overwhelming to put the final report together in the last minute. Make sure you provide experimental data, plots with proper labels, and any evidence to support your results.

To avoid any unpleasant situation, please do NOT share any source files or your report with others. It is OK to discuss and brainstorm with others, but every student must produce their own software and results. If a student is found to submit another student's work, BOTH students will be responsible.