**SY202 – Cyber Systems Engineering**

**Intro**

**CSE**

**Due Date: 01 April 2019**

**LABORATORY INVESTIGATION #07: Simulation of Elevator Control System**

**Objectives**

* To identify and understand the hardware necessary to control and sense within a simple cyber-physical system, in this case, an elevator system
* To graphically model the relationships between electrical, mechanical, sensing, actuating, and software components of a system
* To simulate a closed-loop (feedback) control system using MATLAB and Simulink
* To explore via simulation the effect of hardware limitations (such as sensor noise, actuator’s nonlinear behaviors, and calibration errors) on the closed-loop system

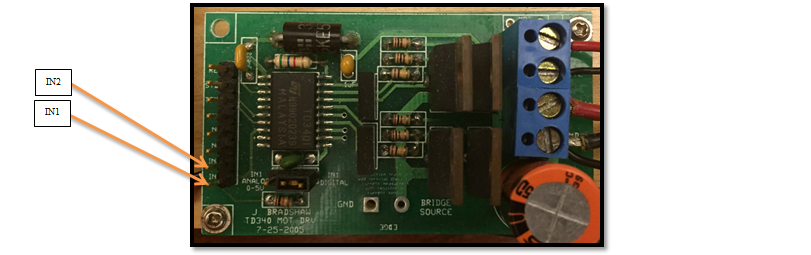
**Introduction**

In this course, you have learned about many areas of technology including, but not limited to, sensors and measurement systems, drive and actuation systems, electronic components, and microcontrollers. As a future cyber-physical systems engineer, it is imperative to understand how the components of the system operate and how each component contributes to the behavior of the system as a whole. Furthermore, as a control designer, it is important to test your design in simulation before implementing it in the real system. Later in the class, you will be tasked to design a series of closed-loop controllers for a real elevator prototype as part of your course’s project. Herein, you will simulate the response of the elevator to proportional and proportional-integral controllers ahead of Project I.

**EQUIPMENT REVIEW**:

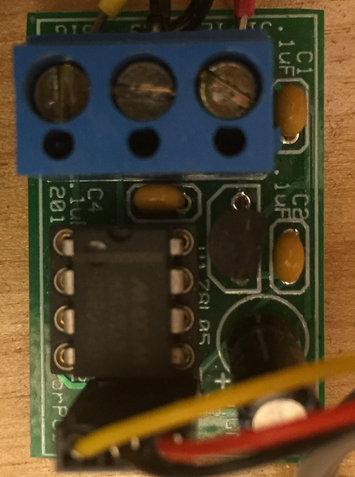
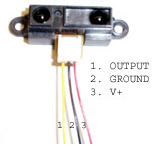
The following information comes mostly from equipment manufacturer specification sheets. Please DO NOT skip this information, as it may help you understand how the closed-loop system works and the practical limitations of your system.

*H-Bridge Quad Power MOSFET Driver for DC Motor Control*



* Allows N-channel power MOSFETS driving in a full H-bridge configuration and is best suited for DC Motor Control Applications. The four driver’s outputs are designed to allow 25kHz MOSFET switching.
* The speed and direction of the motor are to be set by three pins.
* The H-bridge is capable of driving two motors, however, in this lab only one motor is required.
* The H-bridge is powered by a 12-volt source, denoted by reference voltages V+ and V-.
* The H-bridge takes a 3.3V PWM signal as input and produces a V output PWM signal.
* Voltage output is controlled by low side Pulse Width Modulation (PWM). This PWM feature can be made internally when the input pin is connected to an analog signal, or it can be given directly from a digital source.

*Sharp GP2D12 optoelectronic distance measuring sensor*



* The sensor is a low current device that relies on pulsed infrared light for ranging measurements.
* The output is an analog voltage that corresponds to a range. Unfortunately, the device will not function properly if it is loaded. To be loaded means there is significant current draw. The A/D converters on your mbed interfere with the sensor by loading it down during the read phase. As such, the sensor is *buffered* using an operational amplifier.
* The buffer printed circuit board (PCB) contains an operational amplifier. The device is optimized for single supply operation. This buffer protects the sensor from the loading effects of the mbed by providing sufficient current from the main power supply to make the A/D work properly.
* Ignore power considerations of the distance measuring sensor in your functional block diagram.
* The sensor calibration curve can be approximated with a polynomial of degree 5 for a range between 6 to 24 inches. Outside of this range the relationship between voltage and height becomes more complicated.

*TETRIX® MAX DC Motor*

* The DC motor produces angular velocity proportional to the voltage difference applied at the input terminals
* 6-12 Volt operating range
* No-load speed of 152 rpm (at 12 V)
* Stall torque of 320 oz-in or 2.26 N-m (at 12 V)

*T85 Limit switch*



* The T85 limit switch mechanically terminates power from the DC motor when the car reaches a threshold height
* The switch state is naturally closed and opens when the contact lever is pressed

*Elevator pulley mechanism*

* The elevator pulley mechanism converts rotational motion of the DC motor to translational motion of the elevator car.
* The average radius of the pulley (when the car is at 16 inches) is 0.2015 inches.

*mbed microcontroller*

* In this application the mbed microcontroller produces a PWM signal whose duty cycle is user-specified. The duty cycle corresponds to an average applied voltage to the DC motor.
* The mbed in this lab is assumed to receive an analog voltage from the proximity sensor via pin #20. The PWM output signal is connected to pin #21 and the measured height is connected to pin #22. (These might be different pin number from those to be used in the final project). Pins to controlled the direction of DC motor are ignored.

**Exercise**

The functional block diagram (or blue print) in the Appendix is a graphical model of how the closed-loop system for the elevator is built and how each component within the system interacts with each other. For the purpose of implementing control algorithms for any cyber-physical system, it is extremely useful to develop a mathematical computer simulation model that *approximates* the actual behavior of the system. Simulink is a powerful simulation tool whose programming structure follows similarly to functional block diagrams. In this exercise you will create and simulate operation of the elevator system using proportional (P) and proportional-integral (PI) controllers implemented by an mbed emulator.

**Procedure**

**PART 1: Creating the Elevator Simulink Model**

1. Create a working directory folder in your laptop to use for this lab, e.g., “Lab 07”.

Open your Google Drive Folder 🡪 Lab 07 – Elevator Simulation 🡪 MATLAB Mids Files. Download the contents of the “MATLAB Mids Files” folder (5 files in total) to your working directory folder [DO NOT SKIP THIS PART]:

* 1. SY202\_elevator\_library.slx
  2. SY202\_elevator\_template.slx
  3. plot\_fcn.m
  4. mbed.jpeg
  5. StepInfoSimData.m

1. Open MATLAB and specify the directory to your Lab 07 Files folder location. In order for this lab to work, your MATLAB working directory should be your newly created folder with the five provided files placed inside the folder.
2. Open SY202\_elevator\_library.slx. The library contains blocks that represent the components comprising the elevator setup. Each block has specified inputs and outputs.
3. Click the SY202\_elevator\_template.slx file. This file is a blank template upon which you will build your model. This template has been customized to create an animation after each simulation. Any other blank template will not work.
4. Using the Functional Block Diagram in the Appendix, re-create the graphical representation of the system in the SY202\_elevator\_template.slx file using the blocks in the provided library.
   1. First, drag the DC Motor block from the SY202\_elevator\_library.slx into your model. This block contains a mathematical approximation of how the DC motor operates. Double-click the DC Motor block. A window should appear that describes the functionality of the block. Its inputs are the positive and negative reference voltages and its output is the gear shaft speed. The constants represent internal characteristics of the motor that can be modified to make the model better reflect reality. Leave the constant parameters at their default values.
   2. Drag the remaining blocks into your model and connect each block according to the functional block diagram in the Appendix. Double-click any block to learn more information about its inputs, outputs, operation, and internal parameters.
   3. To understand the “internal works” of each block, you can (1) click on the arrow at the lower-left corner of the block or (2) right-click the block and select Mask 🡪 Look Under.
   4. For now, keep the defaults parameters that come with each block.
5. To help you visualize the system behavior, we will use scopes. The scope allows you to see the data after running your simulation by double-clicking on it. Place one scope at each of the following outputs:
   1. The PWM signal from mbed to the H-bridge
   2. The voltage output of limit switch
   3. The speed of DC motor
   4. The speed of elevator
   5. The “real” height of the elevator
   6. The measured sensor’s voltage
   7. The measured height (mbed’s p22)
6. For animation and plotting of your data, we will use the To Workspace from SY202\_elevator\_library.slx. The To Workspace block saves the data of your simulation. DO NOT change the default variable name and format. In addition, we will use “tags” and a multiplex (or mux) to route the signals in a neat, organized way. The tags work as “wireless connections”. A Tag (Goto) connects to at least one Tag (From). The Mux combines multiple signals into a single wire.
   1. In some corner of your template, place the To Workspace block. Add a Mux to your template and connect the output of your Mux to the input of the To Workspace block.
   2. Place a Tag (Goto) at the each of the following outputs: the sensor’s voltage reading, the “real” elevator’s height, and the measured elevator’s height. Double click on the tags and provide reasonable names (e.g., SensorVoltage, RealHeigth, etc.).
   3. Place three Tag (From) blocks and connect them to the Mux block. Double click on the Tag (From) blocks, click Update Tags, and select tags in the following order: top tag = sensor’s voltage, middle tag = real height, and bottom tag = measured height. That’s is, the top input of your mux should receive the voltage reading from sensor, the middle input should receive the actual or real height of the elevator and the last (bottom) input of your mux should receive the measured height or pin #22. You should have something similar to the figure below, except for the tag names:



1. Show your model to your instructor before proceeding with the next part.

**PART 2: Simulation of P and PI control of Elevator**

1. Double-click the Integrator block. Set the initial elevator’s height (Initial Condition) to 10 inches.
2. Double-click the mbed block and set the desired output (height) to 20 inches. Note that this is the “input” or “reference signal” of your closed-loop system (the desired elevator height). For the moment, set the gains of your controller to Kp = 0.1 and Ki = 0.0. This corresponds to a Proportional controller. You may right click the mbed, select “Mask 🡪 Look under the Mask” to see how the controller is being implemented/simulated.
3. Run your Simulink model for 30 seconds by changing the simulation time to 30 seconds and clicking the green play button  located at the top of your model window.
4. After the simulation finishes, an animation should appear showing the height of the elevator vs. time. You should also see a second figure plotting only the height data.
5. Double-click the scopes in your model to inspect the data. Does it match the data shown in the animation? What happens to the PWM signal and the Voltage to the DC Motor during the first 5 seconds of your simulation?
6. Approximate (measure) the settling time, rise time, percentage of overshoot (if any), and steady-state error using the data from the sensor (red line). In an actual physical system, you will be measuring your performance using the output of your sensors. Therefore, we will be making most of our performance computations using the sensor output. You will notice that the sensor signal is noisy. Simply approximate your computations as best as you can. For comparison, approximate the “true” steady state error using the actual/real height of the elevator (use blue line). Additionally, you can run the file StepInfoSimData.m after each simulation. Open the file to see instructions in comments. Annotate those values and save the second figure (assign an appropriate, intuitive name).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | %OS = |  |  |

1. Increase the proportional gain to 0.7 and run your simulation. Repeat step 15 and compare your results. Look into the PWM signal and voltage to DC motor. You should notice that both outputs saturate at their limits during the first 5 seconds. Remember to save Figure 2 with a distinctive name.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | %OS = |  |  |

1. Using the same mbed’s parameters as in step 16 (desired height = 20 inches, Kp = 0.7, and Ki = 0), change the elevator’s mass. In the DC Motor block, change the Load (Elevator) Mass from 0.025 to 0.075 kg. You may need to increase the time of your simulation. Measure (approximate) the settling time, rise time, % of overshoot, and steady state errors. Look into the PWM’s scope signal and describe the steady-state behavior (observe the variation in PWM (minimum and maximum values) near steady-state). Compare to your result in step 16.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | %OS = |  |  |

1. The increase in weight can be seen as a “disturbance” to your system. To compensate for the larger steady-state error, we can add an integral gain to your controller. In the mbed’s block, change Ki = 0.0 to Ki = 0.005. Run the simulation. You may need to adjust the time of your simulation. Approximate settling time, rise time, % OS (if any), and steady-state errors. Compare your results with those in step 17.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | %OS = |  |  |

1. Increase Ki = 0.04, run your simulation and compare with step 18. Annotate the differences. You may need to adjust the simulation time.
2. Set Kp back to 0.1, set Ki to 0.005 and the elevator mass back to 0.025 kg. Run your simulation and compare your results with those from steps 16 and 17. You may need to adjust the simulation time.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | %OS = |  |  |

**PART 3: Effect of sensor’s noise**

Keep the same parameters as in step 20. Add a bias noise to the sensor’s reading. Double-click the IR Range Sensor block and change the bias error from 0.0 to 0.2. Run the simulation. Describe how the system respond to such noise. Compare with the results from step 20.

**PART 4: Optional**

The above guidelines are meant to guide you into understanding:

* 1. The mechanical and electrical components of your system, their functions, and their limitations.
  2. How your control gains (Kp and Ki) affect your closed-loop system?
  3. How changes in your system and/or disturbances (e.g., increase in mass) affect the response of your closed-loop system?
  4. How sensor limitations and noise affect your system?

In order to write a good discussion in your report, I recommend you exploring the system by your own, using other gains and scenarios than those listed above.

**Deliverables**

Follow the lab report template and the general lab guidelines for SY202 lab reports. Refer to the lab rubric for the grading of the lab report.