LAB 5-MEC830
PID Control
(Individual Lab)

1. Purpose
The purpose of this lab is to get-hands-on experience with PID control on microcontrollers.

2. Scope
You will learn about PID control with Arduino. You will work with 1) ***Servo Motor***, 2) ***Ultrasonic Sensor***, and 3) ***PID control algorithm***.

3. Documents
The following documents will help you through the Lab:

- Lecture notes and the ELEGOO programing examples

4. Procedure
Familiarize with the ELEGOO Starter Kit.

- Try loading the examples from class and the kit.
- Use the serial monitor to help debug your program.

You will create a simple crank slider mechanism with a limited range of motion (90 degrees on the servo). This mechanism will control the linear distance from a block to an ultrasonic sensor using a PID control algorithm. See Figures 1 and 2.

Figure 1: PID control

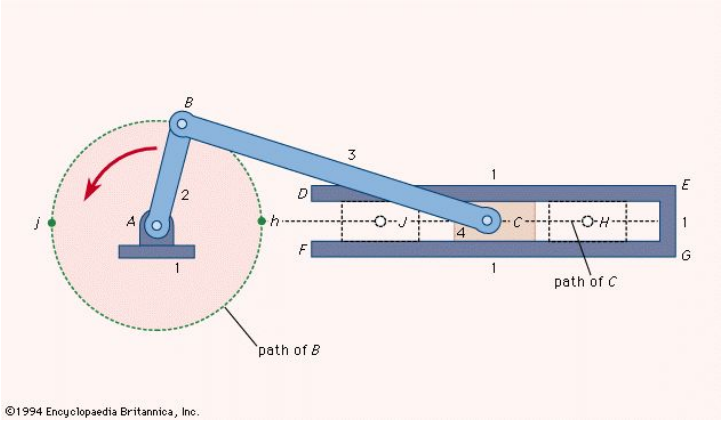

Figure 2: Crank slider mechanism

(Imege: https://www.britannica.com/technology/slider-crank-mechanism)

Using household items (ruler, cardboard, clips, etc.), create a slider crank mechanism. Make sure that the fully extended position corresponds to zero degrees on the servo. The angular range of motion is 90 degrees.

Refer to Figures 3 and 4 for a simple build. Link 1 is about 4cm, and link 2 is about 6.2cm (joint to joint length). This system gives about a range of 5.5cm of movement, corresponding to a range of 90 degrees on the servo.

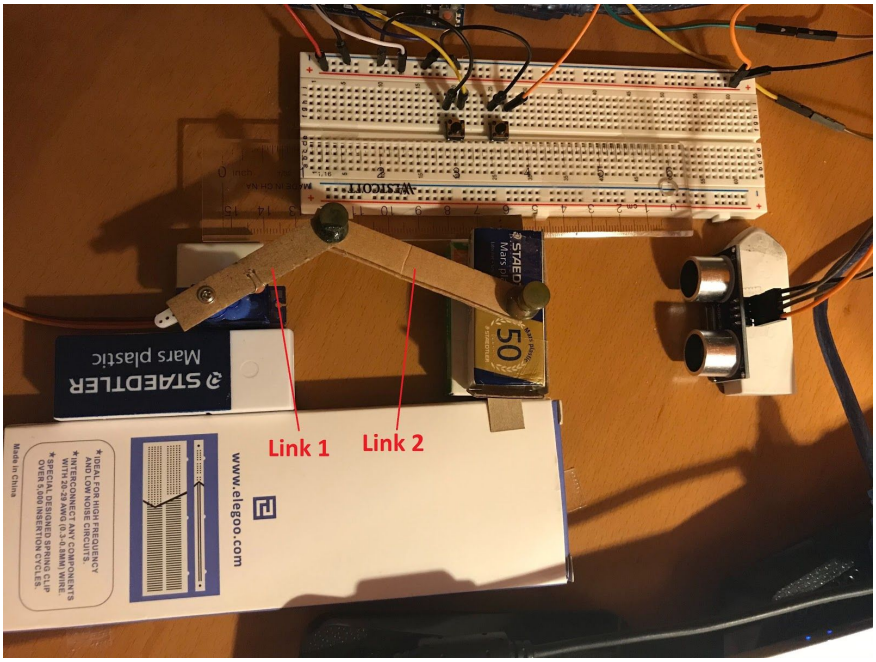

Figure 3: Top view

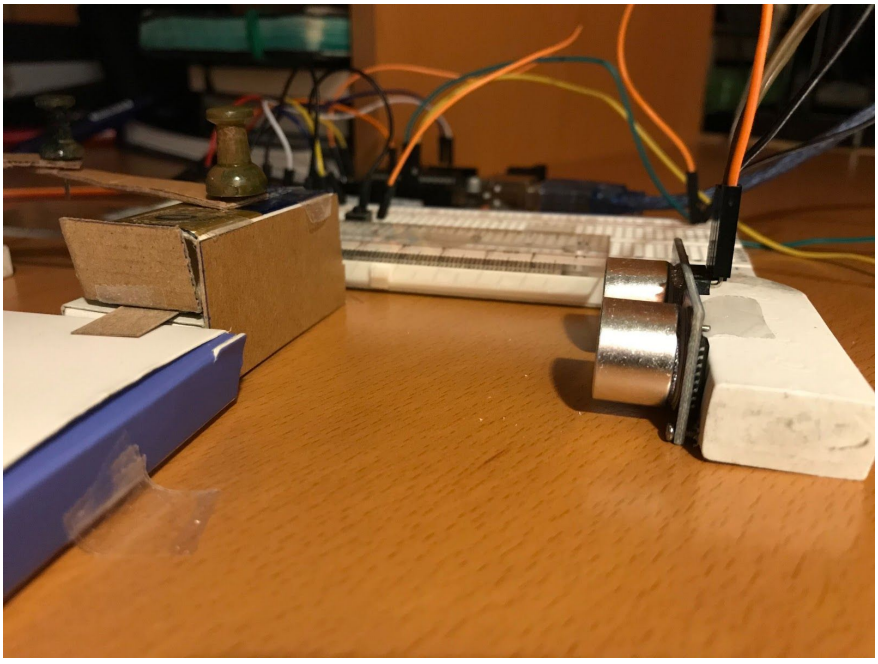

Figure 4: Side view

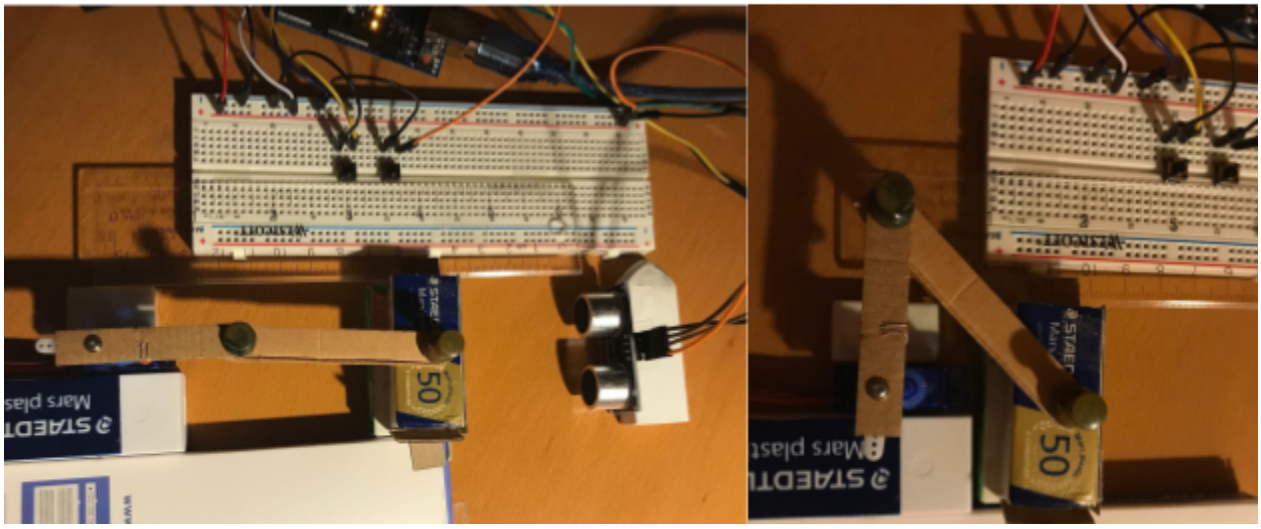

Figure 5: Fully extended and fully retracted

Task:

(a) Build a simple crank slider system as shown above. Reference the dimensions and run the servo to make sure the system moves smoothly. When the arm is fully extended, this corresponds to 0 degrees on the servo, and when it is fully retracted (furthest from the ultrasonic sensor), this corresponds to 90 degrees on the servo. Get a measurement of how far the box can travel linearly. Report the range in report. Set the halfpoint mark to be the setpoint from the sensor. Refer to figure 5.

(b) Write a program that reads the values of the ultrasonic sensor. Compute the error between the read values and the setpoint. Apply PID control to the error values. You may need to adjust the gains of the PID control (K_p, K_i, and K_d) by trial and error.
As shown in the code template, use the following PID library to apply the PID control: Go Tools/Manage Libraries, Filter your search for PID, add the library written by Brett Beauregard. https://github.com/br3ttb/Arduino-PID-Library

(c) Deliverables:
1) Code and report
2) Plots of how the system is following the setpoint, while you try to apply some disturbance. Apply the disturbance by moving the sensor suddenly forward/backward, e.g by ±1cm.
3) Use two push buttons to increase/decrease the setpoint by 0.1cm. Report the plots.
3) Report the gain values (PID) and range of motion.
4) You should also report the steady state error plot, which is the difference between the setpoint and input.

Watch the videos to see how this lab should work, and what you need to present.
https://youtu.be/MEEJR0UQQHs
Once the task is working properly, demonstrate to your TA and ask him/her to sign off.

Quantity	Component
1	Arduino Uno R3
2	Pushbutton
1	Micro Servo
1	Ultrasonic Distance Sensor

Table 1: Bill of Material

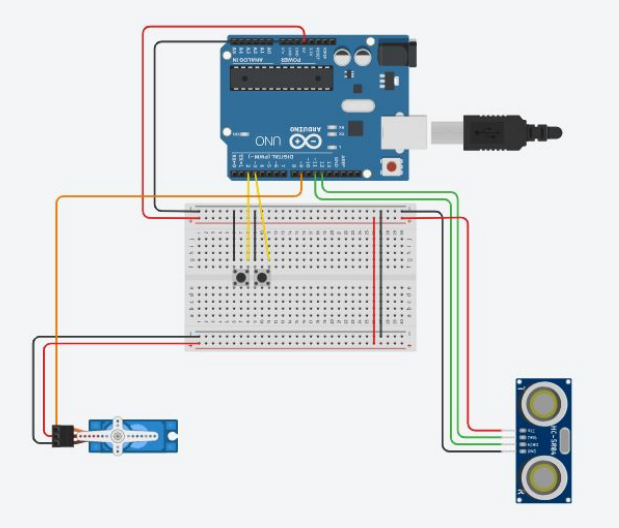

Figure 6: Schematic of the system

Notes:

- For calculating the ultrasonic sensor distance, refer to the Lab 4, where a function is used to calculate the distance to the decimal point, rather than just an integer like in the ELEGOO example.
- Test your servo to see the range of motion. See what values to write to your servo to get 0 to 90 degrees on it, as it may not be perfectly accurate.
- To relate the linear distance read from the ultrasonic sensor to the desired angle on the servo, it is recommended to use trigonometric relations to get the most accurate results. Measure all the important parts of your mechanical system to incorporate into equations in the code. Use the sonar reading as the input to the PID control, and convert the output into an angle for the servo to move.

5. Report

Your lab report should include:

- Signed lab report cover page: http://www.ryerson.ca/mie/documents/
  (make sure you put your lab group number on the front)
- Abstract
- Introduction
- Experimental Equipment (ie. what was used)
- Description of the Program with Flowchart.
- Plot of how the System Responds to Disturbances and Change in Setpoint.
- Conclusions & Recommendations
- Appendix: Program Listing
- Report file name convention:
- Report_[Section#]_[Student_ID]_ [Last_Name}_[First_Name]_LAB1.pdf, e.g. Report_09_00099887766_Smith_John_LAB1.pdf
- Your code also should be submitted in a zip file, if more than one file needs to be submitted. Otherwise submit the code unzipped.
- Code file name convention:
  Code_[Section#]_[Student_ID]_[Last_Name}_[First_Name]_LAB1.pdf, e.g. Code_09_00099887766_Smith_John_LAB1.[c, zip]

- Lab reports are due in 1 week since your lab session starts.
- Submit to D2L→ Assessment → Assignment → Lab5
- Late submissions will be penalized at a rate of 10% per day, where weekends count as two days for online submission.
- Each student should submit his/her own individual report/work. This is not group work.
- Lab attendance is mandatory. If you submit a report without attending the lab, you will get zero marks.
- Weight:
  - 50%: TA confirms that you did the lab during the lab hour
  - 50%: Lab report and the code