Lab 1 – First Swim

You might remember that Ethernet cables ran between your control box and your E79 robot. The cables were probably cumbersome and limited the range of the robot. In E80, you will build an autonomous robot with an onboard Teensy microcontroller that controls its motors and collects data from its sensors. In this lab, you will get acquainted with the Teensy and build your first autonomous robot.

Assembling Robot

Your kit should have an undrilled box, one assembled penetrator, two PCBs and components, and PVC for the robot frame. From E79, you should already know how to assemble the robot frame. Go [here](https://github.com/jo-wong/e80-auv/tree/master/lab1_pdfs) to access the pdfs describing how to assemble the box and PCB.

Open Loop Control

You will need several software tools to run C programs written in the Arduino IDE on the Teensy. Go [here](https://www.pjrc.com/teensy/tutorial.html): follow the software download instructions and run through the blink tutorials. You will end up downloading the Teensy bootloader, Arduino (note the compatible version), Teensyduino, and possibly some OS-specific software.

After downloading the necessary software, you will go through several tutorials on writing Arduino sketches and running them on your Teensy. Complete Tutorial 2: RGB LED ([here](https://www.pjrc.com/teensy/tutorial2.html)), Tutorial 3: Serial Monitor & Input ([here](https://www.pjrc.com/teensy/tutorial3.html)) up to “Built In Pullup Resistor,” and Tutorial 4: Analog Input ([here](https://www.pjrc.com/teensy/tutorial4.html)) up to “User Controlled LED Color.” You might want to still at least read through the rest of Tutorials 3 and 4.

After completing the tutorials, go [here](https://github.com/jo-wong/e80-auv) to download the library files for this lab. Open the “libraries” folder and save the files inside in your Arduino/library working directory. Save the “lab1” folder in your Arduino working directory. The lab1 folder contains the starter code for this lab.

Using the starter code, you will write a C program for your Teensy that sends PWM signals to the robot’s left, right, and vertical motors. The program should log the PWM values sent to each motor and navigate the robot through a specific course. The starter code already has the necessary library headers, global variables, and logging code. You will implement PWM output by completing Todo #1-3. Note: Reading the MotorDriver files might help.

You might notice that an IntervalTimer object was initialized in the starter code. This object is used to switch between logging data to a memory buffer and writing that data out to the SD card. More information about IntervalTimer can be found [here](https://www.pjrc.com/teensy/td_timing_IntervalTimer.html).

Thrust Calibration

The Teensy will send PWM signals to control the motors. The PWM tutorial introduces such signals ([here](https://www.pjrc.com/teensy/td_pulse.html) is another link to the page). Ideally, the motor will begin spinning when the PWM signal is greater than 0. Realistically, it will have a deadzone and a threshold PWM value needed to spin the motors. Use a thrust stand (pictured below) to generate a plot of steady-state force vs PWM value. Update the motor deadzone values in the Params.h library. Use the calibration to write the PWM sequence in the lab1 sketch to drive your robot along the desired path. This calibration will be very similar to what you did in Practicum 1C.