# ILR #1
# Sensors and Motors Lab

Michael Beck, Team E
October 14th, 2016

## Individual Progress

My primary responsibilities for this lab we're to wire and code the force sensor, and to code the PID control for the DC motor with Matt. I also worked some on troubleshooting part integration with the other sensors and motors.

### *Force Sensor*

I was chosen to wire and code the force sensor for this lab. My primary responsibility for our team project is supervision of grasping mechanisms for our robotic arm, and we anticipate that our final design for our gripper will likely involve one or more force or pressure sensors as a method of gripping verification. We may also use force sensors for localization of components of our system, so understanding of this sensor is highly relevant for our project.

The force sensor has a variable resistance that drops as pressure is applied to its pad. This makes the sensor readings easy to interpret through the use of a simple voltage divider. Accordingly I supplied a 5 volt source from our system's Arduino to the force sensor in series with a resistor, and then wired an input pin from the Arduino between the sensor and resistor. This setup allows the Arduino to see a varying voltage as pressure is applied to the force sensor, as a function of the change in the sensor's resistance and that change's effect on the voltage divider setup.

Code for this exact sensor and application was readily available online. The code was altered to reflect the resistance value of the series resistor chosen for this lab. The code was further altered to output a more relevant range of resistance and force values to our GUI, and to provide data to the system even when no force was applied (which the original code did not do). The chosen series resistance came to 3.2k ohms, which was the lowest value that fit nicely with a provided voltage to force curve in the sensor spec sheet (as seen in Figure 1). The sensor operated as expected reporting voltages, forces, and changes in resistance that were approximate to the specification sheet.

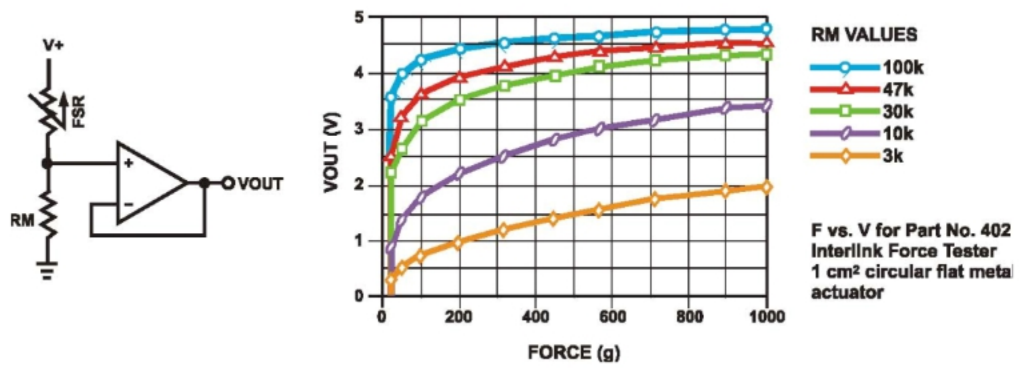

Figure 1: Force to voltage relationship for the force sensor (1).

### *DC Motor*

I worked with Matt in order to wire and code the DC motor for velocity and position control. Both Matt and I were rusty with PID control applications, so I found a template for our code to work from online. The code had to be modified heavily in order to operate with our particular motor and board, which was primarily handled by Matt. Initial operation of the motor showed some signs of position control capability but was unable to execute velocity control, and would instead always run the motor at the same velocity regardless of input.

I first worked to debug the code so that the velocity control was operating correctly by using a series of serial print functions in order to check where the algorithm was failing. It was discovered that our PID error function was growing larger and larger with the altered code, as a result of a bad tick function, which resulted in the motor always running at its top speed in order to make up for the interpreted error. I was able to fix this bug by altering the tick count values which correlated to one full revolution of the motor, and by largely replacing the error check arithmetic, which was originally resulting in an unwanted re-initialization of variables during its runtime. After these fixes the motor ran at desired RPM's very accurately as verified through empirical testing.

Position control turned out to be difficult to make accurate, due to issues with motor drift/momentum which always resulted in an overshoot of the desired position. Attempting to control these problems with PID seemed non-feasible for low angles, as these would require minute movements and very low speeds which the motor would be unable to perform due to stalling issues at low power consumption. Instead of PID a linear fit was created to observed angles ranging from 45 degrees to 360 degrees mapped to given inputs. This fit turned out to be acceptably accurate for desired angles under 360 degrees, usually falling within 5-10 degrees of the given input.

## <u>Challenges</u>

### *Code Revisions and Integration*

The major roadblock for our team with this lab was lack of communication applying to code integration and revision, as well as not having the forethought to make copies of working revisions of code.

A debilitating communication error occurred between myself and Matt in reference to the DC motor code. Matt and I were on the same page as far as code alteration up to the point of having the motor responding to inputs, but I finished up velocity and position control in Matt's absence. I communicated this point in a message board for our group, but Matt did not see this message. The result was that Matt began working on the DC motor code assuming that it was still not functioning, and he ended up completely overhauling the work I had done. His work slightly improved the accuracy of the positioning for the motor, but unfortunately it completely broke the velocity control. By the time we touched base again we were unable to integrate my old code with his code before our lab deadline (his code had already been integrated with the rest of the system, so simply replacing his code with mine was not feasible in such a short time).

A second large hurdle was our groups lack of forethought to make multiple saves files of code revisions. After some of my other teammates had integrated the system the position control for the DC motor was malfunctioning and causing a hang. I was able to troubleshoot this issue and get the positioning working correctly, however we did not make a copy of the code at this time. Afterward we altered the code heavily when trying to get the velocity control for the DC motor working, and these alterations in turn broke the position control functions. When we attempted to revert back to old copies of the code so that we would have the position control functioning we realized our mistake of not making a copy, which resulted in the position control being unusable during our demonstration.

These challenges served as a strong lesson for the importance of clear communication and revision control, even with smaller systems such as the one in this lab.

## <u>**Teamwork**</u>

The work for this lab was split among our five team members, with each teammate being responsible for an individual sensor and a motor, with the exception of Leo who instead worked on developing our system GUI. Matt was responsible for the potentiometer and for the DC motor, Jin was responsible for the ultrasound sensor and the servo motor, and Akshay was responsible for the IR sensor and stepper motor. Akshay also took primary responsibility for integration of the code for each sensor and motor, but this task was shared among all five of us to some degree.

Communication for the lab revolved around making sure that everyone had assigned pins for the Arduino board that did not overlap with other member's, and that everyone was using variable inputs in their code in a way that would be ready to be integrated into the GUI developed by Leo. The team also planned ahead for power requirements of the motors so that the entire system could run on one voltage input from our power supply.

## <u>**Future Plans**</u>

Moving forward with our project we have subdivided project responsibilities our among teammates. Matt is primarily responsible for path planning of our robotic arm. In the coming weeks he will be working to integrate ROS planning libraries with our UR5.

This will involve setting up the system environment and integrating the arm with a workstation. He will also be working to generate mapping data for our resident expert Andrew, who will help Matt in the creation and integration of more advanced planning strategies with our system.

Leo is our software architect. He has created rough drafts of the system architecture, and will be refining the architecture over the next month. Primary focuses for the architecture are reduction in complexity through code division of subtasks, modularity of each subsystem (path planning, perception, and grasping) that will allow for faster subsystem testing in isolated conditions, and the outline for major ROS nodes/topics/messages that will be vital to the overall system controls.

Akshay and Jin our jointly responsible for the vision aspect of the project. In the coming weeks they will be working to implement a image catalog pipeline that was created by MIT for last year's picking challenge. In addition they are also developing trade studies for various potential sensors such as the Kinect and Realsense in order to come to a final decision as to our system sensor.

I am responsible for the grasping portion of our project. I have been working to design a 1 DOF suction end effector for our system so that we may begin full system as soon as possible. In the coming weeks I will be refining CAD designs for the end effector, and passing along potential design considerations to my team for review. I will also be researching vacuum pumps and working to draft the PCB layout for the power system our project will require in order to actuate our end effector.

## References

1. FSR 400 Data Sheet. Http://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/2010-10-26-DataSheet-FSR400-Layout2.pdf. Accessed Oct 14th, 2016.