Sensors and Motors Lab

Rick Shanor

Team D: Human Assistive Robotic Picker

Teammates: Alex Brinkman, Feroze Naina, Abhishek Bhatia, Lekha Mohan

ILR01

October 16, 2015

I. Individual Progress

For this lab, I was responsible for position control of the DC motor and for incorporating the IR sensor. In addition, I worked on filtering the noise from the sonar and IR sensor. Finally, I worked with Abhishek to compile everyone's Arduino code into a single sketch.

Figure 1: DC Gearmotor with Encoder, Solarbotics L298 Motor Driver, Sharp IR Range Finder

After reading the datasheets for these individual components, I wired them all to the Arduino. The power leads of the DC motor connected to the Solarbotics motor driver. The encoder required a 5V power line, which came from the Arduino. The two encoder signal pins connected to pins 2 and 3 on the Arduino, since they required interrupts due to the high signal frequency. On the software side, the PJRC encoder library was used to read the quadrature encoder signal.

I used the Solarbotics L298 Motor Driver to control the DC motor. The logic table, provided by Solarbotics and shown in figure 2, explains that digital output pins from the Arduino connected to L1 and L2 on the motor driver control whether the motor spins forward or reverse. Finally, a PWM signal from the Arduino controls the motor speed.

ENABLE	L1	L2	Result
L	L	L	OFF
L	L	H	OFF
L	H	L	OFF
L	H	H	OFF
H	L	L	BRAKE
H	L	H	FORWARD
H	H	L	BACKWARD
H	H	H	BRAKE
PWM	L	L	PULSE-BRK
PWM	L	H	FWD-SPD
PWM	H	L	BCK-SPD
PWM	H	H	PULSE-BRK

Figure 2: Solarbotics Motor Control Logic Table

I was responsible for controlling the motor position. To accomplish this, I used a PID control algorithm. PID stands for "Proportional-Integral-Derivative". The equation below was used to calculate the gain.

$$u(t) = K_p * error + K_i * \int_0^t error * dt + K_d * \frac{de}{dt}$$

The error, error sum, and change in error were calculated based on encoder readings. This gain was passed to the Solarbotics L298 Motor Driver as a PWM signal. As the motor approaches a desired position, the input changes. PID control is used to control system performance metrics such as rise time, overshoot, and stability. I had to work to tune the Kp, Ki, and Kd gains such the system performed adequately.

I used the output of the IR sensor to control the position of the DC motor. Specifically, the sensor output a voltage between 0V and 3V depending on the distance between the sensor and an object. The Arduino received this voltage through analog input. SHARP provided a transfer function such that the voltage output could be converted to a distance. Finally, the distance was converted to a degree. The degree was then passed into the PID controller to drive the position of the motor.

The IR sensor, as well as the sonar sensor, were very noisy. In order to reduce the noise, I implemented a rolling average filter. The equation for this filter is shown below. In addition, I plotted the sensor output for several different parameters to show the filters ability to reduce the noise of the system at the cost of a delayed response time.

$$Sensor\ Value = OldValue * A + NewValue * B \quad s.t. \quad A + B = 1$$

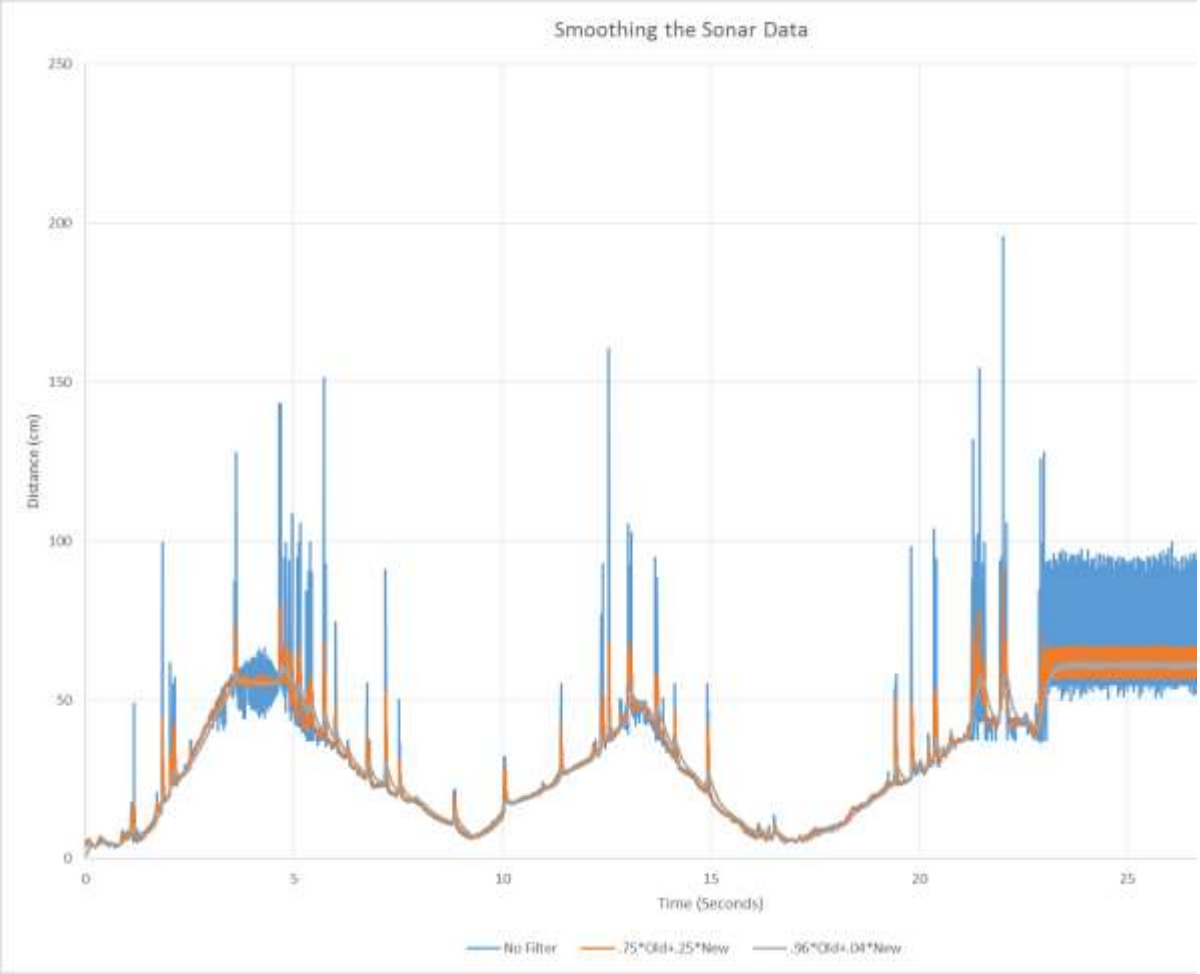

Figure 3: Rolling Average Filter Results

II. Challenges

The biggest challenge I faced was achieving the desired performance on the PID controller. Due to high friction and 'stiction' in the motor, a large input was required to make the shaft start spinning. However, once it started, there was a significant acceleration and sometime overshoot past the desired position. I was able to improve performance by incorporating a PID controller and tuning the gains. I was never able to totally eliminate some of the overshoot and jitter in the system.

Abhishek and I also faced challenges trying to control the servo. First, we checked pins with the multimeter and reworked the servo code. Finally, after exhausting all other options, we swapped out the driver and the system worked perfectly. When purchasing our own components, I plan to research parts carefully and make sure to check reliability.

III. Teamwork

The team effectively used GitHub to share code with each other. As a team, we generated an Arduino pinout to help with integration. Alex was responsible for DC speed control and the pressure sensitive resistor. Abhishek worked on control of the stepper motor and set up the sonar. He also did button debouncing. Feroze set up the servo and a potentiometer. Finally, Lekha, Feroze, and Alex all worked together to build the GUI and handle communication between the GUI and the Arduino over serial. The entire team was involved in integrating code to get the system functioning as a whole.

IV. Future Plans

I am currently working to develop a suction system for grasping objects. I have already made a CAD model (figure 4) and ordered parts to prototype. This week, I will get the prototype suction system picking up small objects. In addition, I plan to redo the CAD such that the mechanism can be held by PR2, the robot platform the team now has access to through Maxim Likachev's lab. In addition, I plan to prototype a vision system. I will use SIFT feature detectors to predict an items position in a frame. These algorithms will be tested on an image database from last year's Amazon Picking Challenge.

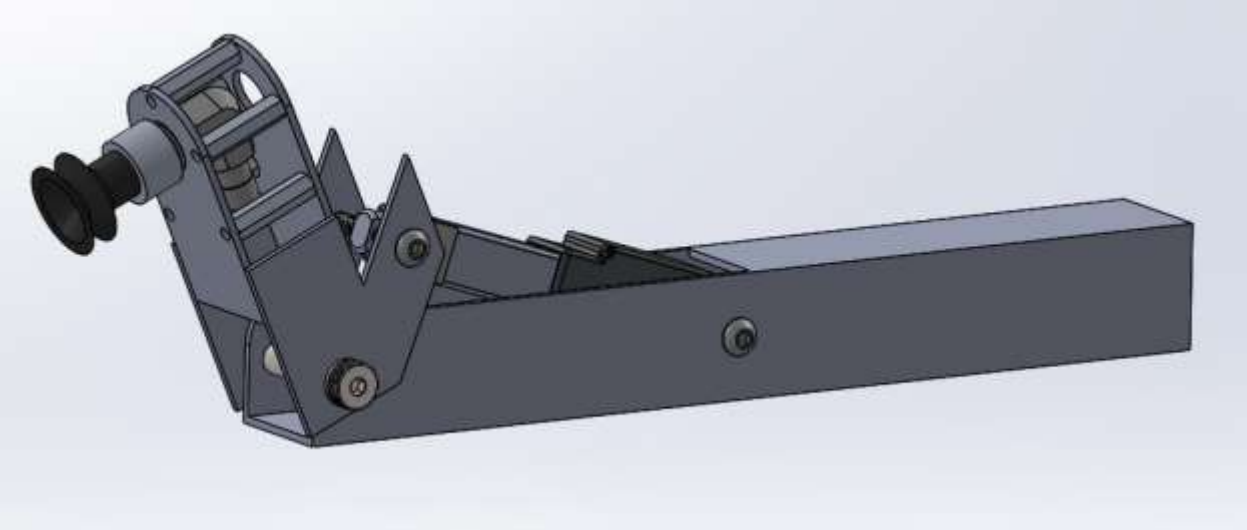

Figure 4: Preliminary Gripper Design