Control of Mobile Robotics Lab 2 – Wall Following

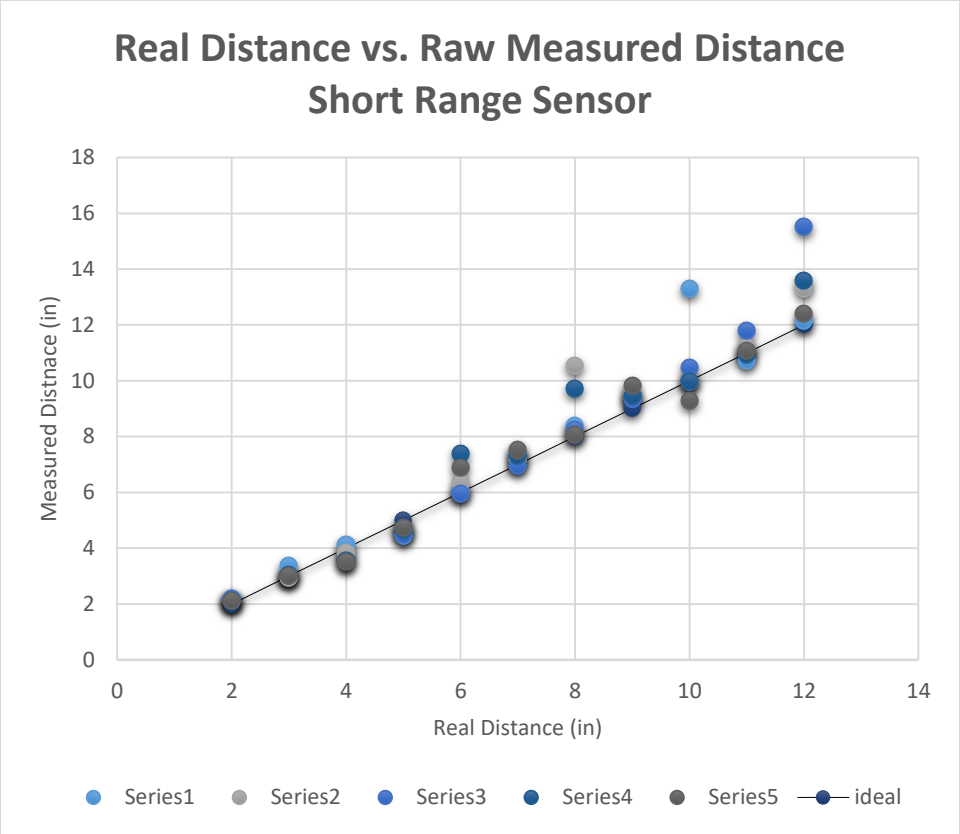
Section I: List of Files

```
wallDistance.ino     // Task 1
wallFollowing.ino    // Task 2
fixedSensorTest.h    // Sensor Test
MySharpSensor.h      // B.1.1.2
MyFollowFuncs.h      // Header File for Task 2
MyEncoders.h         // Header file used for kinematics
MyServos.h           // Header file to control speed of servos
```

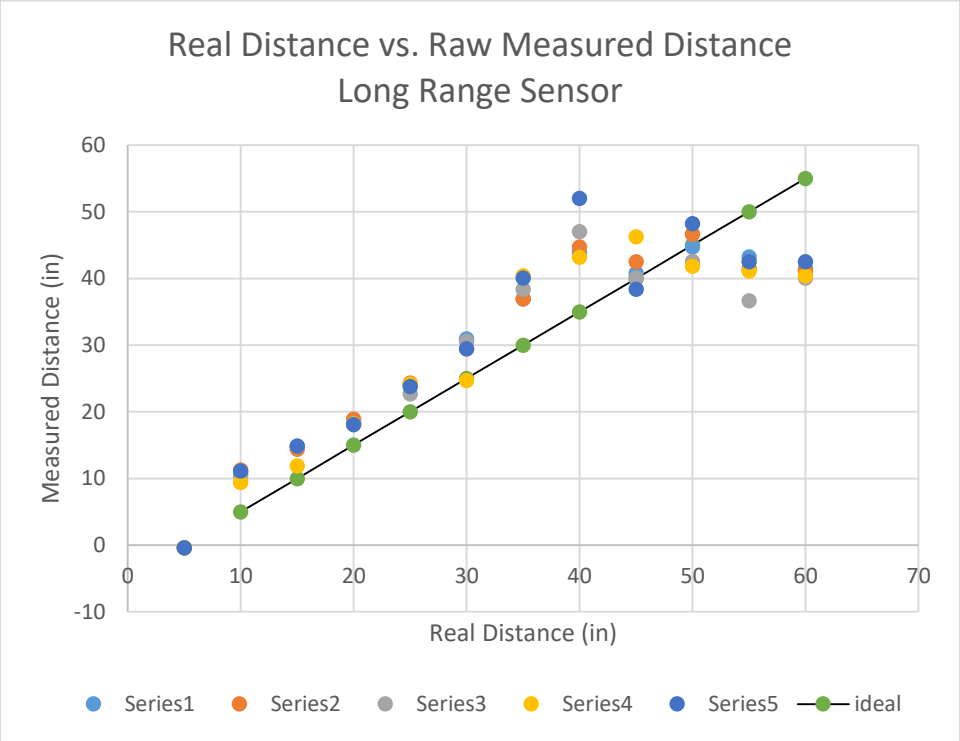
Section II: Data and Analysis

The following section presents the data gathered from the short and long range sensors of the robot. Since each sensor has a particular range of distances it can detect (represented by a non-linear curve), certain distances picked up by the sensor will be more precise than others. Generally speaking, the closer a robot is to a wall, the less accurate the *long distance* sensor will be; conversely, the further away the robot is from a wall, the less accurate the *short distance* sensor will be.

Plots of real distance vs actual distance gathered by 5 consecutive raw sensor readings are shown below with the line y = x plotted for reference (representing ideal results). Figure(1.1) represents the short range sensor and Figure(1.2) represents the long range sensor.


Real Distance vs. Raw Measured Distance
Short Range Sensor
Measured Distnace (in)
Real Distance (in)
0
2
4
6
8
10
12
14
16
18
Series1
Series2
Series3
Series4
Series5
ideal

Figure(1.1)


Real Distance vs. Raw Measured Distance
Long Range Sensor
Measured Distance (in)
Real Distance (in)
-10
0
10
20
30
40
50
60
70
Series1
Series2
Series3
Series4
Series5
ideal

Figure(1.2)

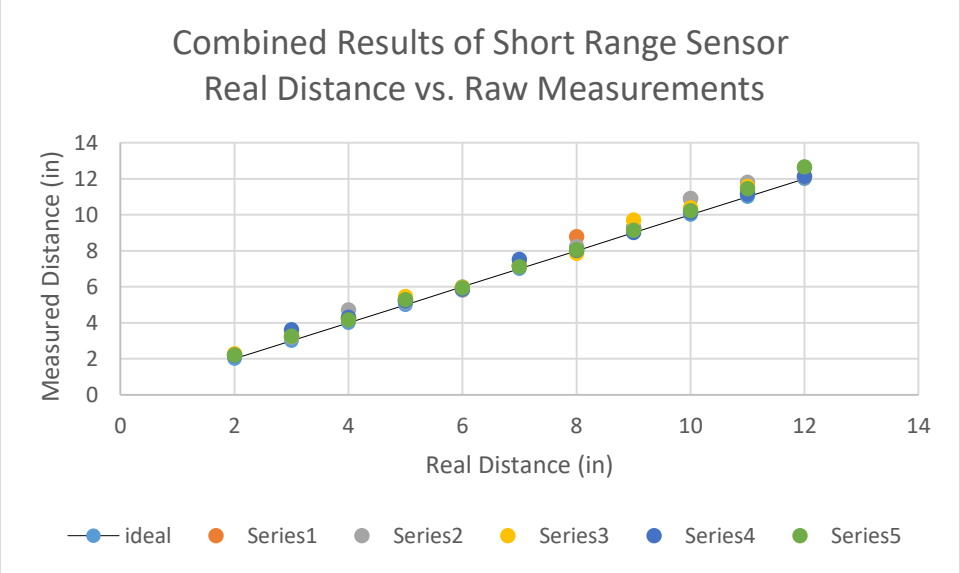
Observing both graphs, we can safely say that both sensors are very noisy, resulting in frequently inaccurate measurements. To refine these measurements, we must use some form of combinational method that smooths out the noisiness of the sensors and yields a more accurate result.

Using the graphs above, we determined which combinational method would work best for each sensor. We noticed that many measured values fell directly onto the line y = x, with one or two outliers that were far off from the general reading. For this reason, we chose to apply the median as our combinational method for the short sensor.

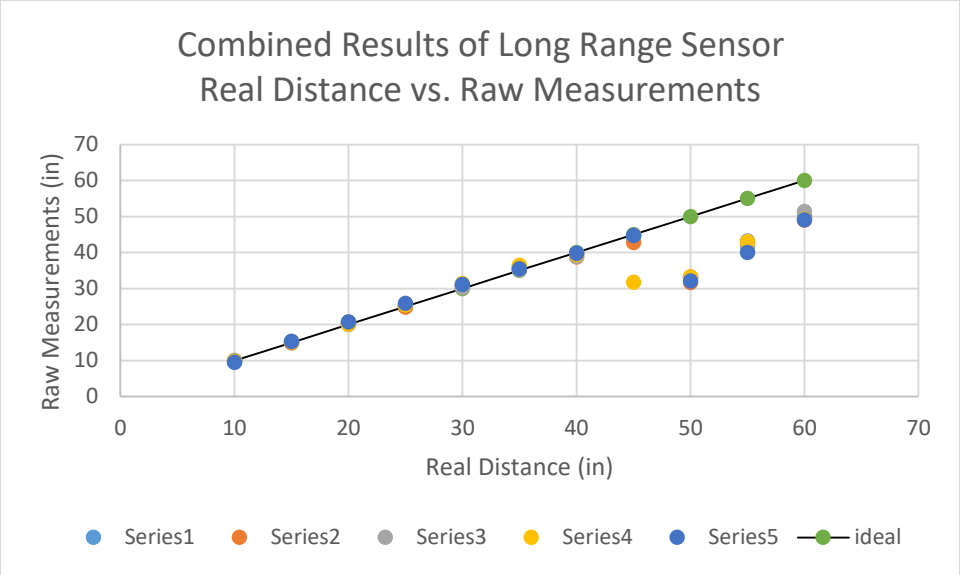
Contrarily, the long range sensor contained very few cases where values fell directly onto the line y = x. Many of the readings honed into a general region that was slightly skewed above the line. For this reason, we choose to apply the mean as our combinational method for the long range sensor.

Using an array of size N, with N = 10, we sampled N values for each sensor, and applied the appropriate combinational method; a swap function and a bubble sort algorithm were used to sort the data needed to obtain the median.

The same plots from before are shown below, now using our refined sensor values. Figure(1.3) represents the short range sensor, and Figure(1.4) represents the long range sensor.



Figure(1.3)



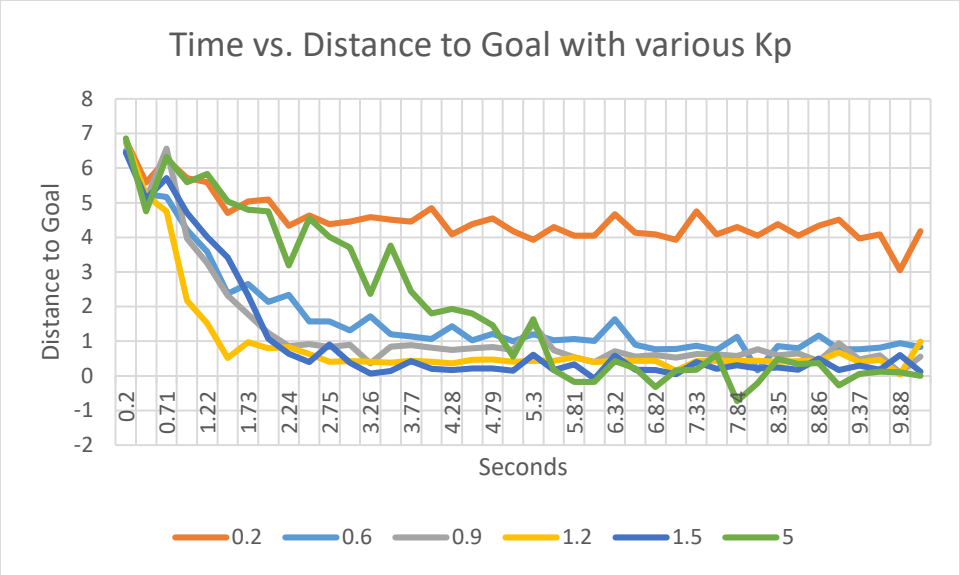
Figure(1.4)

As observed in the graphs, our sensor measurements were greatly improved by applying the combinational methods. Despite this, as the sensor approached its boundary ranges, the sensors once again became fairly noisy, particularly for the long range sensor.

Based off of this information, we concluded that data gathered at very long ranges isn't very reliable and any application of it should compensate for this factor.

During this lab, one of our tasks was to implement a Wall Distance function that maintained a certain target distance between the wall and the robot. If the robot was moved to a distance greater or less than the target, it would adjust its position depending on how far or how close it was from the wall. A proportional gain scalar factor was used to scale the magnitude of the velocity proportional to the distance the robot was from the wall. In other words, if the robot was very far from the wall, it would move very quickly to close the gap; conversely, if it was very close, it would move very slowly to close the gap.

We used various values of Kp (proportional gain scalar constant) to scale the velocity. These values included 0.2, 0.6, 0.9, 1.2, 1.5, and 5.0. in Figure (1.5), the plot for distance vs time for the various Kp values is shown.



Figure(1.5)

Section III: Proportional Control and Wall Following

In the previous section, we discussed the concept of a proportional gain used to scale the speed of the robot in accordance to its measured distance from the wall. We expanded on this same idea when implementing wall-following capabilities to the robot.

The following control diagram illustrates the closed loop controls used by the robot to implement both wall distance and wall-following capabilities.

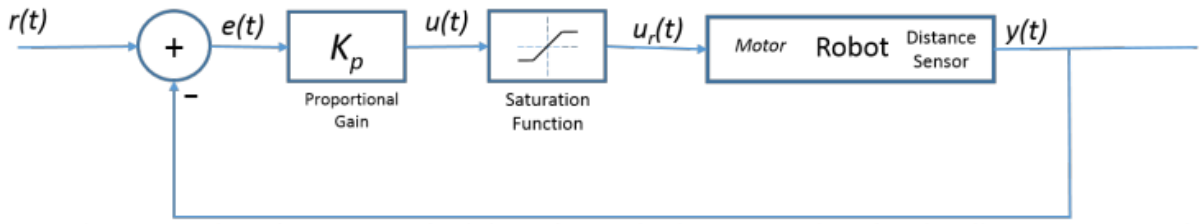


Figure 3: Close loop control of the robot velocity proportional to its distance to a goal.

The following functions outline the steps taken by the robot to carry out the closed loop control diagram.

$$u(t) = K_p * e(t) \quad \text{(Eq. 1)}$$
$$u_r(t) = f_{Sat}(\, u(t)) \quad \text{(Eq. 2)}$$
$$u_r(t) = f_{Sat}(\, K_p * e(t)\,) \quad \text{(Eq. 3)}$$
$$e(t) = r(t) - y(t) \quad \text{(Eq. 4)}$$
$$u_r(t) = f_{Sat}(\, K_p \left(r(t) - y(t)\right)\,) \quad \text{(Eq. 5)}$$

Eq. 1 calculates the value u(t), which represents the proportional gain in velocity of the robot, where Kp is the proportional gain scalar factor and e(t) is the error between the goal and the current distance.

Eq. 2 and Eq. 3 take the value obtained in Eq. 1 and ensures it does not exceed the limitations of the robot. Namely, if u(t) is a speed outside of the range of -6.0 in/sec and 6.0 in/sec, then choose the boundary value closest to it.

Eq. 4 calculates the error distance of the robot by subtracting its current distance from the goal. As seen in Eq. 1, the smaller the error becomes, the more the robot will reduce in speed.

Eq. 5 illustrates all data necessary to obtain a value from the closed loop feedback. The robot will constantly update this data and readjust the value of Ur(t) accordingly.

This closed loop control diagram was used during the wall-following portion of the lab to correct the translational speed of the robot as it approached a wall, as well as correct the angular velocity of the robot to maintain a 5in distance from a parallel wall.

Section IV: Conclusions

Over the course of this lab, we encountered many challenges caused by factors both in and out of our control. One major issue we noticed was that the long range sensor on our robot was scratched on the surface of the material; this most likely impacted the accuracy of the readings obtained by that sensor, which also explains the noisiness seen in the plots from Section II. As we progressed through the lab, the performance of the robot did not go as expected when the long range sensor became more relevant to our tasks.

Other issues included jittery behavior exhibited by the robot, poor calibration behavior exhibited by the right servos due to physical defects, and inadequate response time to changing stimuli. We also struggled to choose a sample size N throughout the lab. In certain cases, having a quicker response time was more desirable than having an accurate reading. In most cases, the median yielded the best result for both the long and short range sensor. For this reason, we settled on an odd sample size value of N = 9.