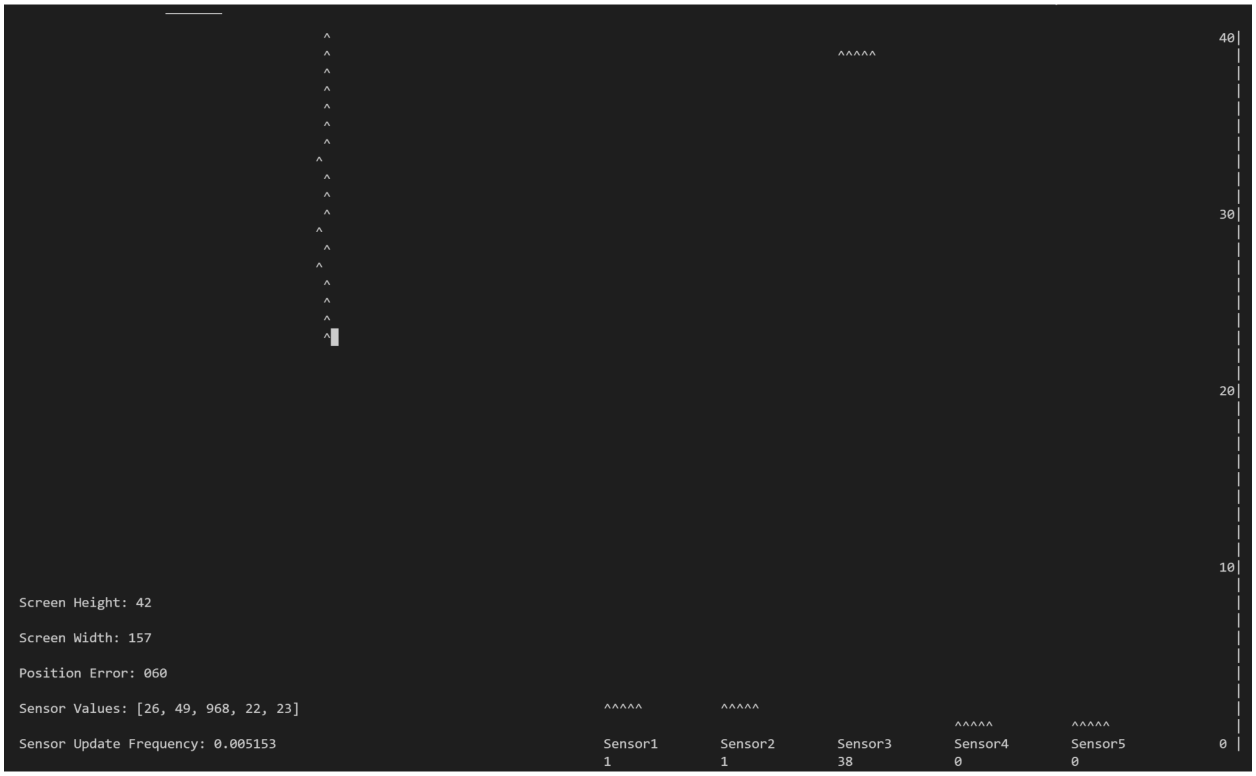
Lab2 Report
AJ Funari, Blake Dunaway

Line Sensor Module:

In our line sensor module code, we started off by manipulating the original code given to us in Alphabot. This code was already very useful because it extracted important data from each of the 5 sensors. The calibrating function was especially useful because each time you run the function, it would calibrate it to a different sized line if needed. The main function we used was TR.readline(), which would return us an array that helped each sensor value from 1000 meaning on the line, and 0 being off of it. The function would also return a position value, which was from 0 to 4000. 0 represents the very left sensor being on the line, 2000 the middle sensor on the line, and 4000 the very right sensor being on the line.

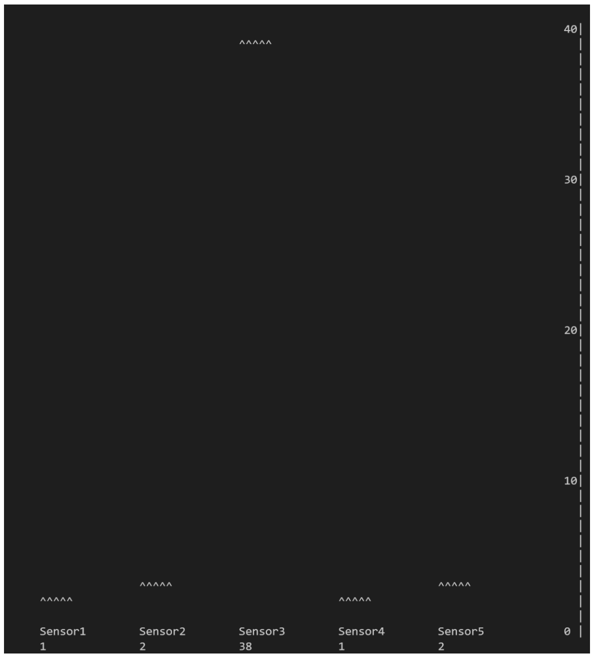
Curses Application:



```
                                                                                  40|
                                                                                  30|
                                                                                  20|
                                                                                  10|
Screen Height: 42

Screen Width: 157

Position Error: 060

Sensor Values: [26, 49, 968, 22, 23]          ^^^^^      ^^^^^
                                                                     ^^^^^      ^^^^^
Sensor Update Frequency: 0.005153             Sensor1    Sensor2    Sensor3    Sensor4    Sensor5     0 |
                                              1          1          38         0          0
```

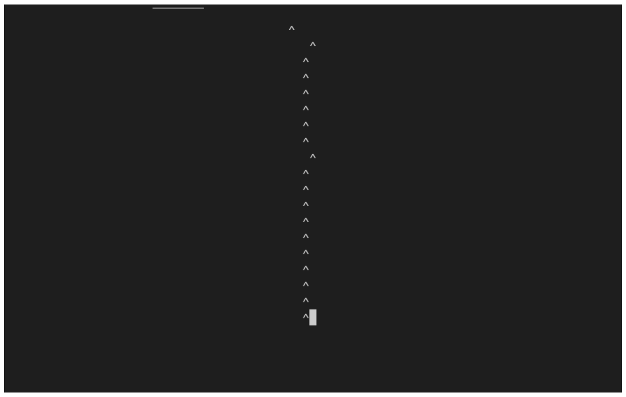
Part 1: Setting up the screen:
The first part of the application is setting up the dimensions of the screen and deciding where we are going to display specific text. We use the getmaxyx() to obtain the dimensions of the screen. We created a class called cursclass with three different functions to display text on the screen. The screensetup() function determines where we are going to display the bar chart for the sensor values. We divided the screen into 10 sections (a = width / 10). The first half of the

screen is dedicated to dynamic position values of the Alphabot with respect to the line. The second half of the screen at x coordinates a*5, a*6, a*7, a*8, a*9 represents the position for each sensor to output the dynamic value in bar chart style. The addbar() takes a y position coordinate (which is each rescaled sensor value), and a x coordinate(the same position we dedicated each sensor value to). The removebar() function works exactly like the addbar() function except it writes blank spaces to the previous reading of each sensor value. This will allow the bar chart reading to update its position each time through the while loop.



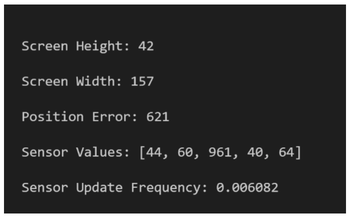
```
                                                                                     40|
                                   ^^^^^                                               |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                     30|
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                     20|
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                     10|
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                                                                                       |
                    ^^^^^                                        ^^^^^                 |
     ^^^^^                                        ^^^^^                                |
                                                                                       |
     Sensor1        Sensor2        Sensor3        Sensor4        Sensor5             0 |
     1              2              38             1              2
```

Part 2: The position of the robot relative to the line
The second part of the application is outputting the position of the robot relative to the line. To obtain the position value, we used a function provided by TRSensors.py called readLine(). Calling this function returns a value between 0 and 4000. If the most left sensor is directly on the line, the value returned will be 0, and if the most right sensor is directly on the line, the value returned will be 0. Since we do not have 4000 pixels of width resolution, we rescaled the value by the width of the screen using the formula x = (float(position) / 4000.0 * width. This value is then divided by 2 to only add text on the first half of the screen. Using the function stdscr.addstr(), the y coordinate given to the function is initially 0 (the top of the screen), and the x coordinate is the rescaled position value. In the while loop, we increment y by 1 each pass through to output a dynamic line moving down the screen.



```
^
   ^
  ^
  ^
  ^
  ^
  ^
  ^
   ^
  ^
  ^
  ^
  ^
  ^
  ^
  ^
  ^
  ^
  ^
```

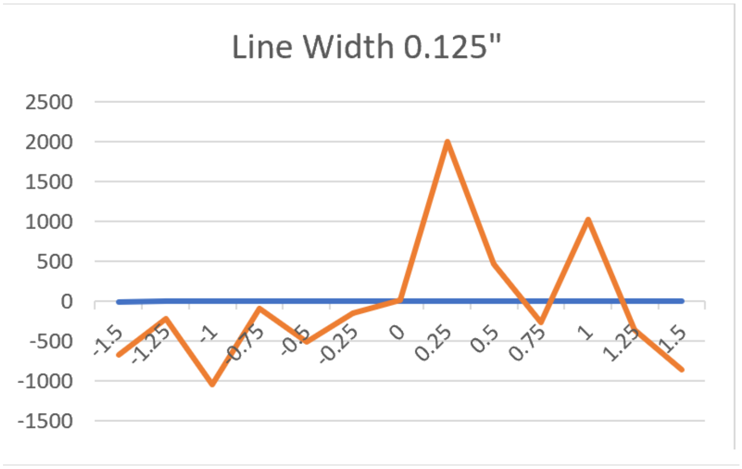
Part 3: Displaying important information
The third part of the application serves to show the screen height, the screen width, the position error of the robot relative to the line, each sensor value, and the sensor update frequency.

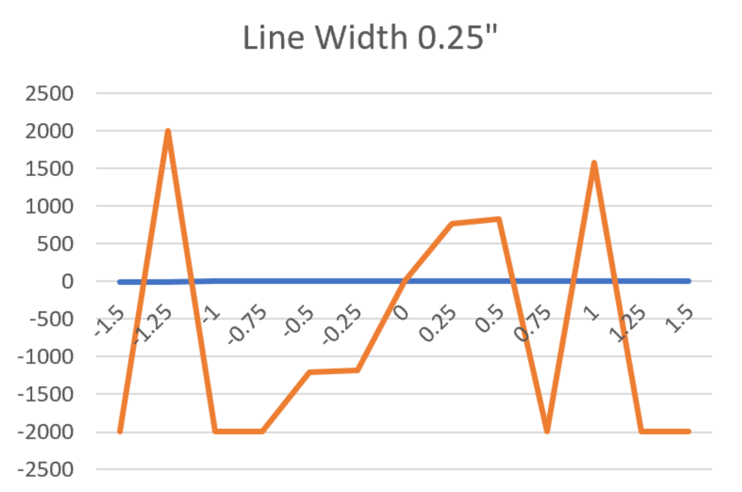


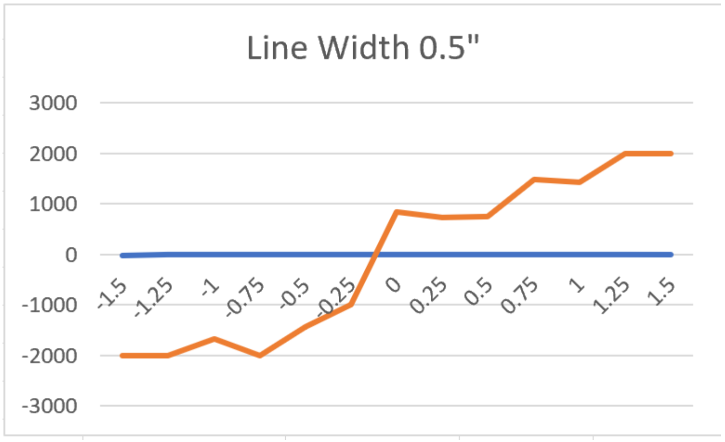
```
Screen Height: 42

Screen Width: 157

Position Error: 621

Sensor Values: [44, 60, 961, 40, 64]

Sensor Update Frequency: 0.006082
```

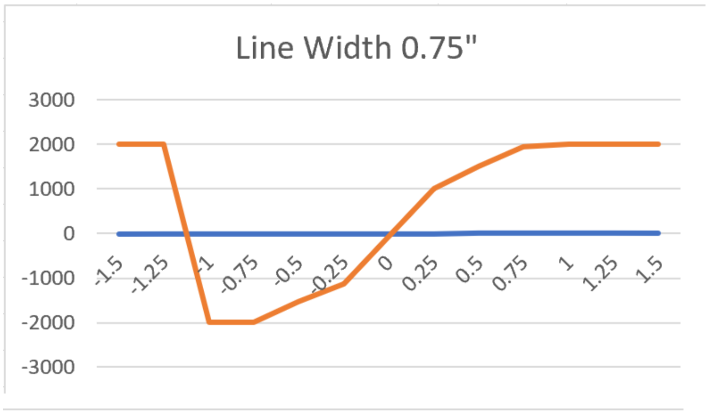
<u>Line Accuracy Graphs:</u>

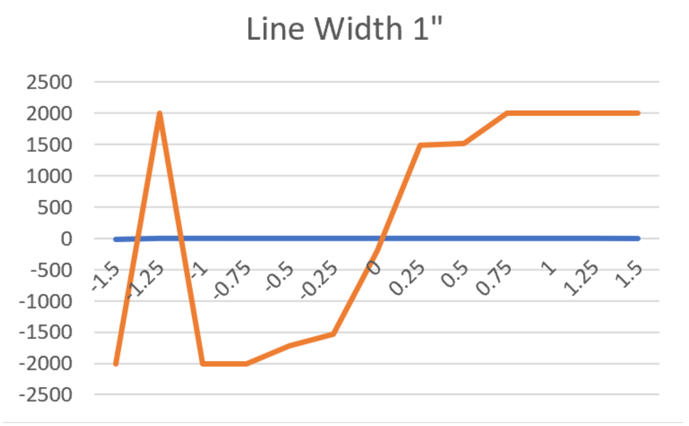
For the line sensor accuracy measurements, we measured the reading from the sensors of the position relative to the line compared to the actual displacement of the Alphabot with respect to the line. To obtain the most accurate measurements, we averaged together 5 accuracy tests per line width. From the accuracy measurements, the readings from each test were fairly inconsistent. When we finished collecting the data, and created a graph for each line width, we determined the most accurate data we collected was using line width of 0.5”. From the graphs you can see the readings from the sensor value were the most consistent and accurate on both sides of the 0.5” line, thus this would be the best option for the track width.


Line Width 0.125"
2500
2000
1500
1000
500
0
-500
-1000
-1500
-1.5
-1.25
-1
-0.75
-0.5
-0.25
0
0.25
0.5
0.75
1
1.25
1.5


Line Width 0.25"
2500
2000
1500
1000
500
0
-500
-1000
-1500
-2000
-2500
-1.5
-1.25
-1
-0.75
-0.5
-0.25
0
0.25
0.5
0.75
1
1.25
1.5


Line Width 0.5"
3000
2000
1000
0
-1000
-2000
-3000
-1.5
-1.25
-1
-0.75
-0.5
-0.25
0
0.25
0.5
0.75
1
1.25
1.5





Link to Video
https://drive.google.com/file/d/1Cs8lU6J9dJgBlBocOK48rZuEtttKcnDj/view?usp=sharing