ECSE 211 Design Principles and Methods

Lab 4 - Localization

Group 53
Spencer Handfield
260805699
Yi Zhu
260716006

1.Design Evaluation

The robot is composed of 2 large motors controlling the wheels and the brick supported just above it so as to minimize its width. To reduce the axis bending caused by the force, the connection axis between both large motors was reinforced and the length of wheel axis on each large motor was reduced. To accomplish both ultrasonic and light localization, we used both the ultrasonic sensor and the color sensor. We placed an ultrasonic sensor at the front bottom robot in a zero-degree angle due to the detection efficiency. The color sensor is placed at the right back bottom of the robot for black line detection and orientation correction.

A central class lab4 controls the flow of execution of the program based on user input. Two classes were added to the existing framework from prior labs like the odometer and display. The ultrasonic localizer class contains two primary methods, the rising edge and falling edge methods. Both works based on the exact same principle that the robot will rotate indefinitely until it locates the first edge, at which time is sets its theta to zero and then begins rotating the opposite direction until finding the second edge. At which time the calculated theta, divided by 2, based on the symmetry of the corner and edge detection, will rotate it to 45 degrees and then a last 45-degree adjustment so its angle is zero along the y-axis. A filter was implemented around the edges to prevent faulty detection by the US. The rising edge method begins facing wall and rotates until the ultrasonic sensor detects a distance above 40 (signaling the edge as it is leaving the wall), turns back to detect the opposite point of 40cm separation signaling the second edge. The falling edge method does the same except it begins facing away from the wall and turns towards it until a distance of less than 40 is detected and the same series of detection and angle correction as illustrated above is followed. A counter is used to break the conditions of the turning loops when the edges are detected. The turning methods are taken from previous labs and adapted to take the counter conditions and appropriate directions passed at the correct moments in the code. The value of 40 was tested and found to be the most effective versus other values, given the 30cm size of the square sides.

The light localizer code works off the assumption that the robot is aligned to 0 degrees along the Y-axis and that there is a displacement between the center of the robot and the light sensor on the rear. The robot advances indefinitely until a black line is detected. The first black line detected is the x-axis, it then reverses the known displacement to bring the center exactly on the axis. A counter is used to keep track of how many lines have been detected and what stage of the code to enter. The robot again advances until it detects the Y-axis and returns the known displacement, placing it on the origin. It rotates a final 90 degrees to be facing along the Y. All classes have helper methods to move the robot forward and backwards (be it indefinitely or a known distance) and a turn method to accomplish the same in rotations. Conversion methods between radians were included to ease the difference between rads and cm or degrees. The decision behind the modified version of the light localization was a question of simplifying from the tutorial slides method. Instead of spinning around 270 degrees to detect the 3 lines and adjust, it builds off the confidence in the US angle to simply advance and adjust to the lines with a counter. The hardware was set up accordingly with the light sensor offset to the side so as to not falsely detect multiple lines at the origin intersection and see the axis lines leading into such as the markers to correct to, crucially, one at a time.

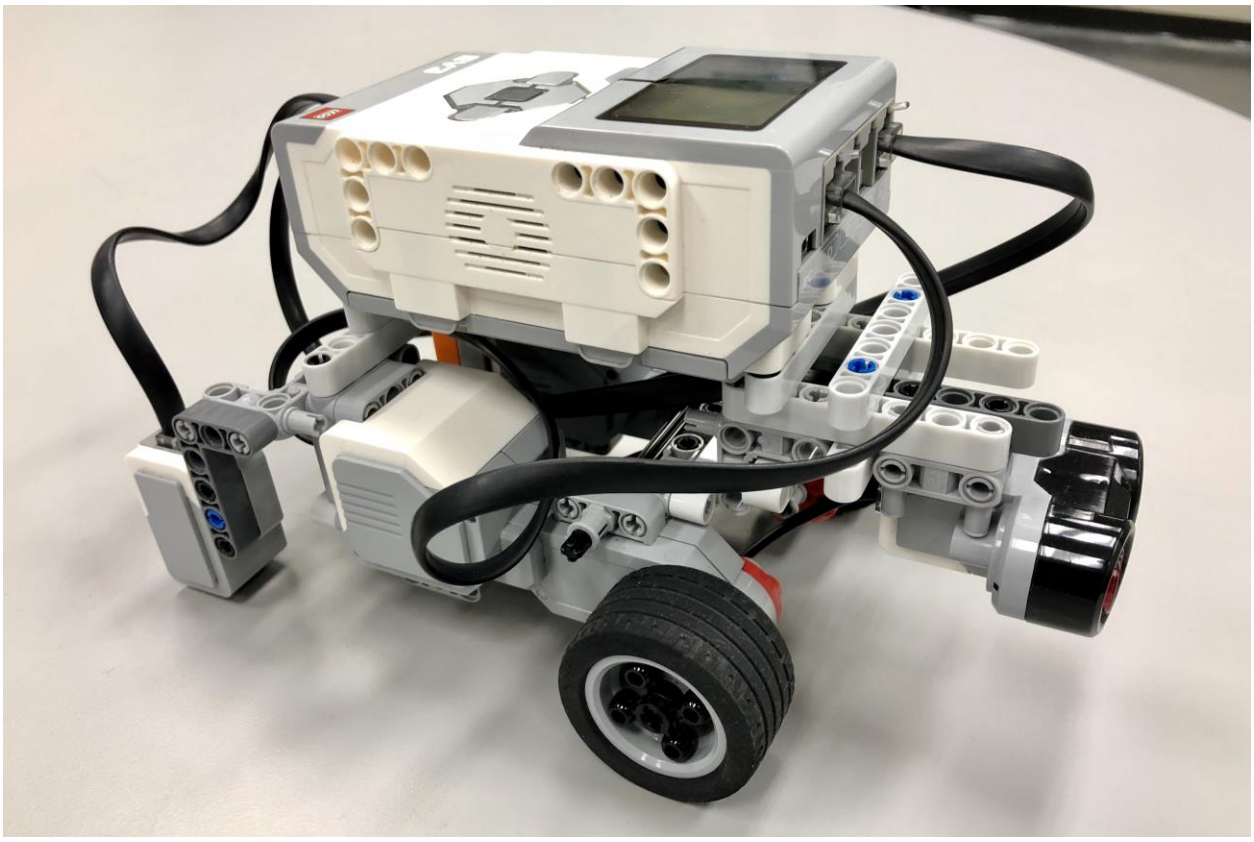

Figure 1

Flowchart for Ultrasonic Sensor Localization

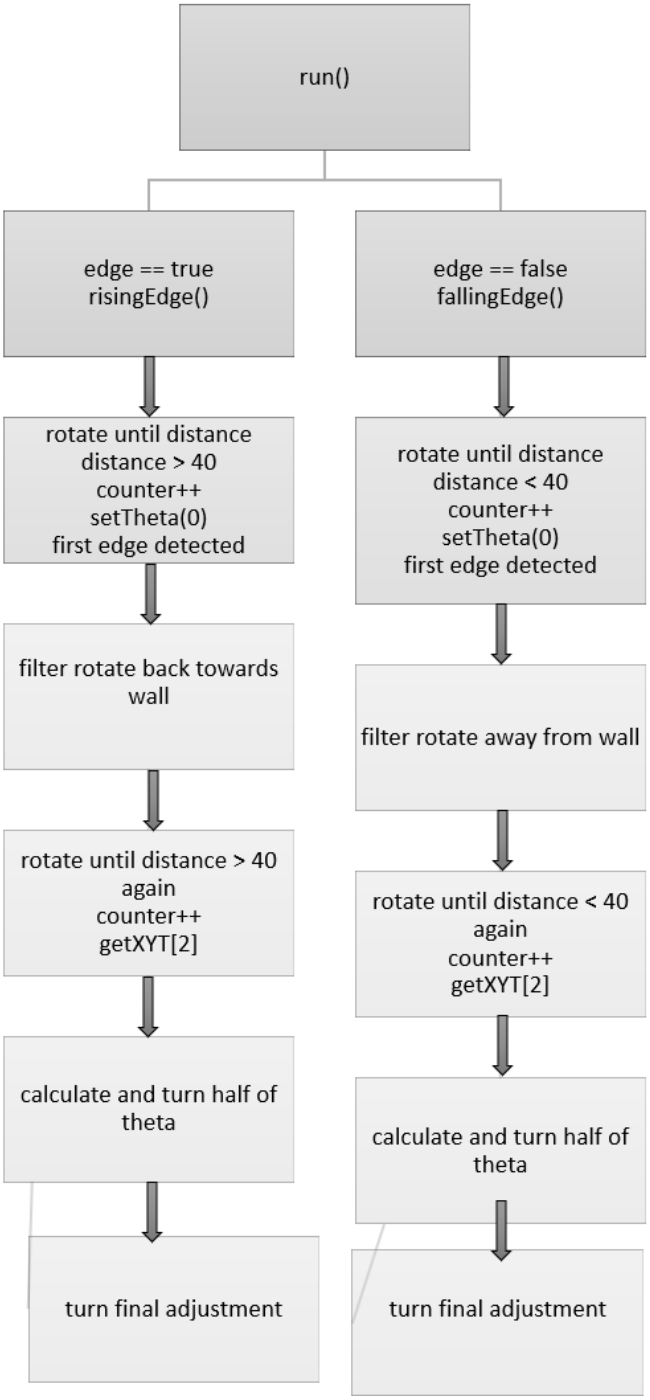

Figure 2

Flowchart for Light Sensor Localization

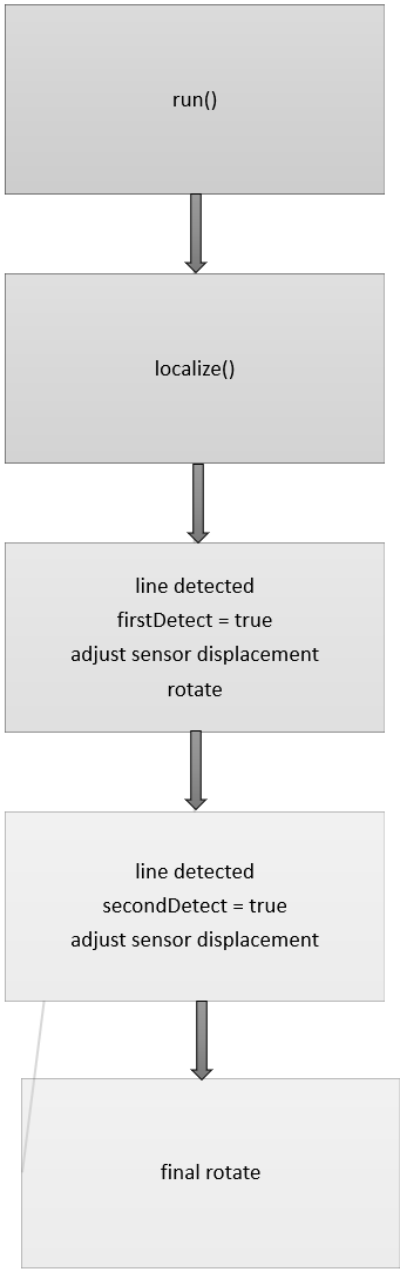

Figure 3

2.Test Data

2.1 Rising Edge

Trial	US angle error(deg)	Final angle error(deg)	Final x(cm)	Final y(cm)	Euclidean distance error(cm)
1	0	0	0.7	0.0	0.70
2	-3	0	0.2	0.0	0.20
3	-2	-2	0.5	0.3	0.58
4	-3	-1	0.0	0.0	0.00
5	2	2	0.0	-1.2	1.20
6	0	0	0.2	0.0	0.20
7	0	1	0.2	0.0	0.20
8	1	0	0.0	0.0	0.00
9	4	3	0.2	1.1	1.11
10	1	1	0.5	0.0	0.50

Chart 1

2.2 Falling Edge

Trial	US angle error(deg)	Final angle error(deg)	x(cm)	y(cm)	Euclidean distance error(cm)
1	-2	-2	0.0	0.5	0.50
2	0	0	0.2	0.0	0.20
3	0	0	0.3	0.0	0.30

4	1	1	0.2	0.0	0.20
5	0	0	0.2	0.0	0.20
6	7	5	1.1	-0.7	1.30
7	2	1	1.5	0.0	1.50
8	0	0	0.2	0.0	0.20
9	0	1	0.4	0.0	0.40
10	0	0	0.3	0.0	0.30

Chart 2

3.Test Analysis

3.1 Rising Edge

Name	Mean	Standard Deviation
US angle error	0.00	2.09
Final angle error	0.40	1.35
Euclidean distance error	0.47	0.04

Chart 3

3.2 Falling Edge

Name	Mean	Standard Deviation
US angle error	0.80	2.27
Final angle error	0.60	1.68
Euclidean distance error	0.51	0.46

Chart 4

Mean:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Standard Deviation:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}$$

Euclidean error:

$$\epsilon = \sqrt{(X_F)^2 + (Y_F)^2}$$

4.Observations and Conclusions

4.1 Which of the two localization routines performed the best?

With an US error mean of 0, a final angle error mean of 0.4 and a euclidean error mean of 0.47, along with smaller standard deviations across all three, it is clear that by all measurable stances that the rising edge routine performed the best.

4.2 Was the final angle impacted by the initial ultrasonic angle?

Yes, since the software was setup to only turn set amounts along the axis of the square for light localization, an initial error in the angle would translate through the also predetermined correction distance and deviate the robot further off course the more it moved throughout the process thus creating larger final angle error as a result

4.3 What factors do you think contributed to the performance of each method?

Both methods performed with very high levels of accuracy, we feel that this is the result of the fortification of the hardware support resulting in more accurate turning and movements, along with software additions like the filters and slow speeds which heighten accuracy and reduce potential error. The main factor which contributed to the difference in the actual methods would mainly be attributable to the consistency of measurements that the rising edge had by always facing the wall whereas we've seen the US suffers when it must deal with no objects in close proximity leading to increased chances of error.

4.4 How do changing light conditions impact the light localization?

Changing light conditions have a significant impact on the light localization, especially due to the use of the color sensor and not the red sensor. The light in the lab is consistent however, this statement was proven when certain tiles boards with dirtier tiles would completely ruin the light localization (detecting a "line" over a dirty patch) thus confirming that shifting light conditions did have an effect. In the other lab where lighting was more bright and the tiles more clean, the robot performed ideally

5. Further Improvements

5.1 Propose a software or hardware way to minimize errors in the ultrasonic sensor

Given the very satisfactory performance of the US localization, it would seem that the implementation of proposed solutions from previous labs is the greatest asset to reducing errors. In that vein, it would seem the most effective was adjusting hardware appropriately and compensating in the software. In terms of the hardware, not much can be said other than continuously ensuring it own symmetry and stability for precise movements, via reinforced wheel support and balancing the components to such. On top of optimization, to truly ensure that the the robot is facing the correct way, we could add a rotating motor to the US sensor which, when the edge is found, can rotate to each side, checking the angle it turns and the values it reads on both sides to confirm it is truly facing the edge and not just a momentary false reading not handled by any of our current filters.. In terms of the software, further filters seem to be the most effective at compensating for the hardware, particularly in terms of dealing with the falling edge inaccuracies a more sophisticated filter can be implemented to deal with the expected error prone undetermined distances created by not having a wall present.

5.2 Propose another form of localization other than rising-edge or falling-edge

Another form of localization could be a complete 360-degree rotation, where throughout, based on a gated if condition and a counter could detect the 2 nearest points of the wall to the robot. Based on the lab assumptions and the symmetry, this would give a point along the diagonal of the square that the robot is situated on and then can allow for calculation of that hypotenuse along with the angle to adjust to.

5.3 Discuss how and when light localization could be used outside of the corner to correct Odometry error, e.g. having navigated to the middle of the larger floor

Same as we did in our light localization. To re-localize the robot in the middle of the board, let the robot move until it detects the grid line and stops, then move back half of its body, which is 15.5 cm in our case. This is the way to correct x coordinates. After that, turn the robot 90 degrees and then move and detect the line and move back again. Finally turn the robot to head right 0 degree and reset all coordinates to be (0, 0, 0) on the display screen.