Lab Report #4 – Localization

Group #55: Alex Hale (260672475) and Xianyi Zhan (260672960)

ECSE 211 – Design Principles and Methods

October 13, 2017

Design Evaluation

Hardware

A simple three-wheeled robot was created using two motors, two wheels, one ball bearing, one ultrasonic sensor, one colour sensor, one EV3 brick, and various connecting pieces (Figure 1). The ultrasonic sensor was mounted to the front of the robot, above the brick, and the colour sensor was mounted to the front of the robot, just in front of the wheel axle.

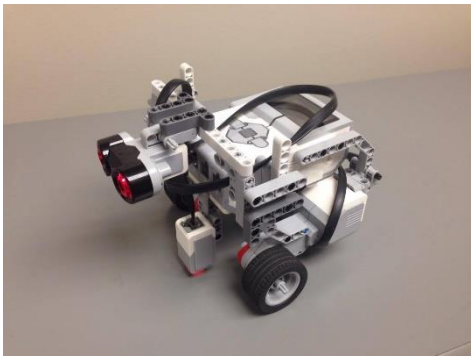

Figure 1 – Robot Hardware Design

Software

The software structure is outlined in Figure 2.

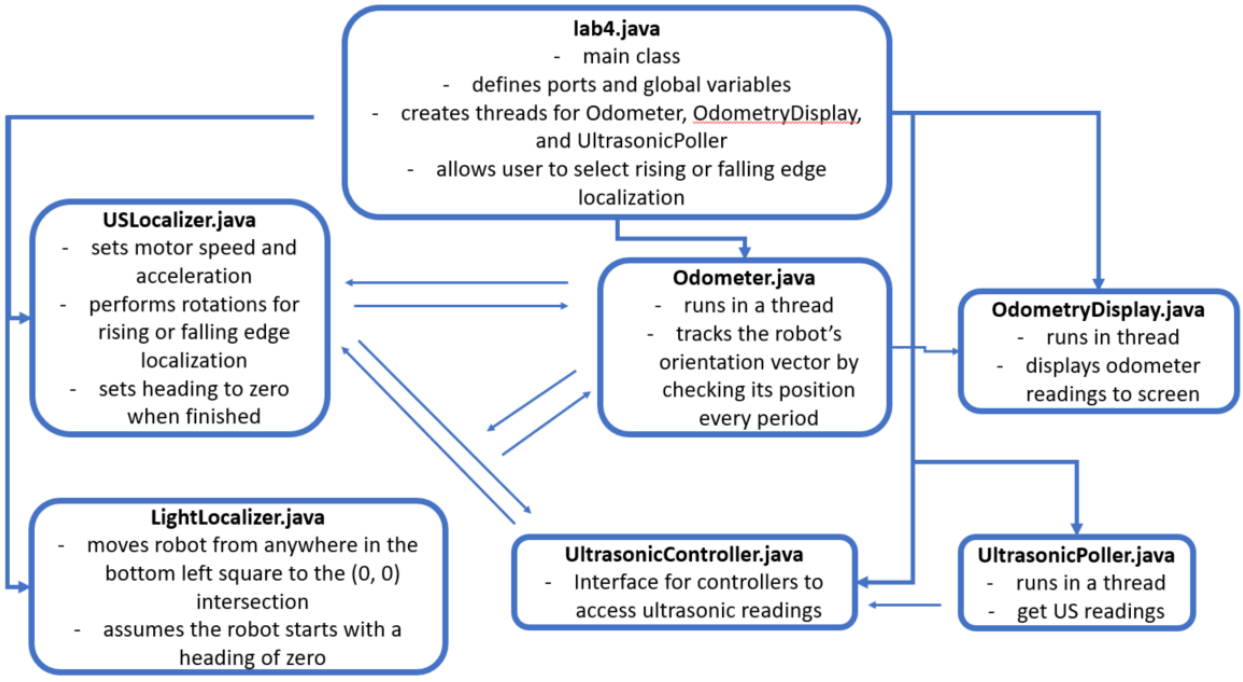

Figure 2 – Software Structure

Workflow (hours are a total of both lab partners)

1) Design robot based on given criteria (2 hours)
2) Write software based on and design criteria (5 hours)
3) Test design on grid floor. Identify errors in hardware and software that cause the robot to fail the design criteria, fix them, repeat (12 hours)
4) Demo project to TA (0.5 hours)
5) Lab report (3 hours)

Test Data

Rising Edge Test

Test results are shown in Figure 3.

Test	US Angle Error	Euclidian Distance Error	Final Angle Error
1	4.00	4.29	4.02
2	4.10	3.76	4.02
3	5.74	6.59	5.74
4	6.74	6.01	6.70
5	4.04	4.37	4.06
6	5.51	5.48	5.50
7	2.73	3.67	2.76
8	6.74	6.12	6.66
9	4.08	4.58	4.10
10	2.13	1.72	2.13

Figure 3 – Rising Edge Results

Falling Edge Test

Test results are shown in Figure 4.

Test	US Angle Error	Euclidian Distance Error	Final Angle Error
1	0.61	0.68	0.64
2	1.64	1.32	1.57
3	2.70	3.49	2.75
4	2.70	2.57	2.70
5	0.83	1.57	0.87
6	0.26	-0.55	0.25
7	2.09	2.62	2.18
8	2.88	2.15	2.86
9	0.36	0.86	0.44
10	2.48	2.26	2.40

Figure 4 – Falling Edge Results

Test Analysis

$$Mean\ (\mu) = \frac{sum\ of\ all\ values}{number\ of\ values\ \ (N)}$$

$$Standard\ deviation = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

Rising Edge Analysis

Analysis is shown in Figure 5.

Value	Mean	Standard Deviation
US Angle Error	4.58	1.57
Euclidian Distance Error	4.66	1.45
Final Angle Error	4.57	1.54

Figure 5 – Rising Edge Analysis

Falling Edge Test

Analysis is shown in Figure 6.

Value	Mean	Standard Deviation
US Angle Error	1.66	1.05
Euclidian Distance Error	1.70	1.17
Final Angle Error	1.67	1.04

Figure 6 – Falling Edge Analysis

Observations and Conclusions

The falling edge localization routine performed better on average than the rising edge localization. This observation was made while testing our robot, and it is confirmed with the data above – the mean of the error in the rising edge test is higher than the mean of the error in the falling edge test.

On average, the final angle error was the same magnitude as the ultrasonic angle error. This make sense because our light sensor localization routine *only* moved the robot to the (0, 0) point. It made no effort to change the angle after the ultrasonic localization. In fact, the light sensor localization routine *assumed* that the ultrasonic localization routine resulted in a heading of near-zero, and would not perform correctly if the ultrasonic localization error was too high.

The rising edge routine produced errors because the ultrasonic sensor was inconsistent in detecting the rising edge of the wall. It is unconfirmed whether this issue was due to hardware (e.g. the quality or positioning of the ultrasonic sensor) or software (e.g. the filter for erroneous readings or the criteria of the while loop that detected the rising edge). In hindsight, our best theory is that the ultrasonic sensor had a filter that was too heavy-handed. It was reused from the wall follower lab, when the robot had to

filter out gaps in the wall. However, in our testing, the ultrasonic sensor experienced erroneous readings only on the back wall, not on the left wall, and the filter does not explain this phenomenon.

The ultrasonic sensor was much more accurate in detecting the falling edge of the wall. The filter did not come into play here, because the ultrasonic sensor was moving toward the wall (i.e. the filter would already have reached its limit before the robot started detecting the wall).

Both routines struggled to turn the robot to face in the 0^o direction after taking ultrasonic readings. This is likely because the ultrasonic sensor did not read the rising or falling edge in the same location on each wall. These errors were relatively consistent between trials, so we fixed them by adjusting the amount the robot turned away from the mathematically ideal value. These changes are indicated by comments in the code.

Our light sensor was more accurate in this lab than previous labs, mostly because we mounted it closer to the floor. The sensor did not behave any differently in the two light conditions we had available (the large lab room and the small demo room). Overall, the light sensor had very few erroneous readings.

Further Improvements

Small errors in ultrasonic sensor readings could be reduced by introducing a filter. A moving-average filter, as demonstrated during lecture, could ignore any erroneous readings from the ultrasonic sensor. Alternatively, when the ultrasonic sensor detects a rising or falling edge, the robot could stop rotating, pan slowly back and forth over the edge location, then settle on the position where the ultrasonic sensor reading is closest to the required distance.

Another localization routine could use the corner of the field to localize. The robot could rotate until it detected the local maximum that occurs when the ultrasonic sensor is pointed toward the corner, then rotate 135^o clockwise to face a heading of 0^o. Similarly, another localization routine could be devised using the points where the ultrasonic sensor is pointed perpendicularly to the walls. The ultrasonic sensor distance is at a local minimum when it is pointed directly at the wall. The two local minima of the two walls could be used to localize in the same way that the two falling or rising edges of the walls are used.

As outlined in the Lab 2 report, one or two light sensors could be used throughout the robot's navigation. If only a single light sensor were available, it could be used to measure the distance traveled between the detection of two lines, then the trigonometric calculation in Figure 7 could be used to adjust the odometer. If two light sensors were available, the distance traveled between the moment when the two light sensors detect the line could be measured, then the trigonometric calculation in Figure 8 could be used to adjust the odometer.

s

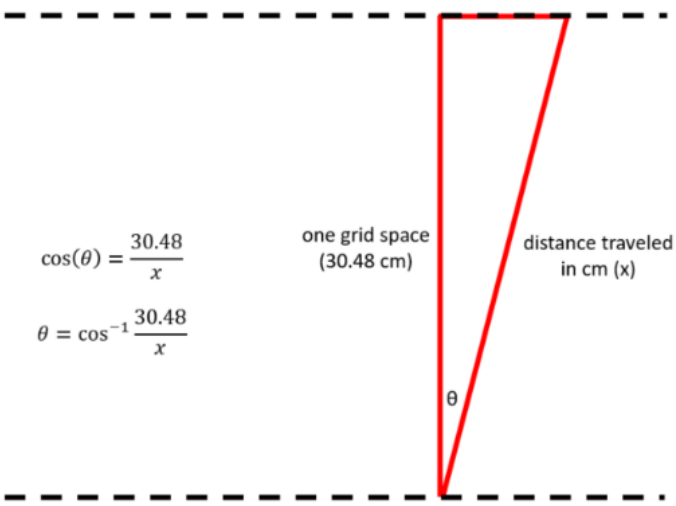

Figure 7 – one-sensor angle correction

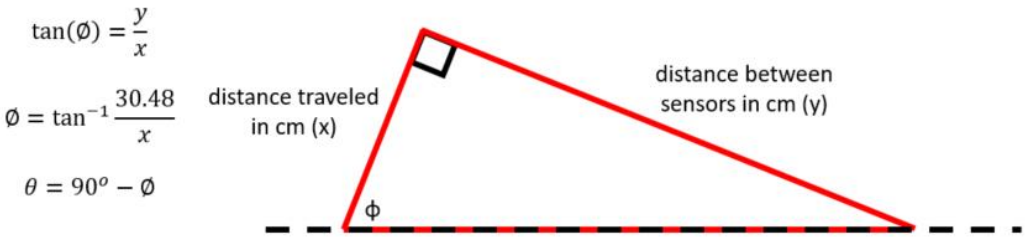

Figure 8 – two-sensor angle correction