Lab 5 – Searching for Objects (Group 51)

Alex Bhandari-Young – 260520610, Neil Edelman – 110121860

2013-10-20

Abstract

Object detection is implemented using colour. Then, the robot localises in a set environment, searches for the foam object in a sea of wooden blocks, captures, and brings it to a set location in under five minutes.

1 Data

The robot was programmed with two initial ultrasonic sensor localisations: one, it turns until it is facing the arena, and turns until it gets a near sensor reading signalling the other wall (falling edge;) two, it turns until it faces the wall and keeps turning until it is facing the arena (rising edge.) It then turns until it is facing zero within a certain tolerance. The data for falling edge is in Table 1 and the data for rising edge is in Table 2. The θ_{start} is the starting position of our robot.

2 Observations and Conclusions

2.1 Discussion

We see from the data in Table 1 and Table 2 that it doesn't matter what position it starts in, it will localise successfully.

The discrepancies always on one side are caused by the polling. The polling rate is 100 ms and the turn rate is $10°s^{-1}$, leading to maximum error in Equation 1. Each method approached the stopping point from a different direction, but the mean error for both is zero within experimental error.

$$\frac{10°s^{-1}}{100\,\text{ms}} = 1° \tag{1}$$

Experimentally, other robots using the same frequency and pinging off our robot sometimes confused it; the sound waves caused it to read a wall or gap when none was there.

The data for rising edge theoretically takes up to three times longer to collect; from the corner, the wall is taking up three quarters of the solid angle while the arena only takes one quarter. However, this does not happen in practice because the robot takes up a non-negligible space when compared to 40 cm, which is the wall detection threshold. If we increased it, then objects in the playing field would be more likely to register as walls (we could better this by making a more complex algorithm.)

We could store the data from the rising edge and later calculate the wall distances without going back and measuring them again. This would suggest using rising edge localisation.

However, `getFilteredData` was not filtered. However, it was better than Lab 3[**?**] because it waited for pinging. This disproportionally affects rising edge. Instead of writing complex code for filtering, the more reliable falling edge was used. The reason is the error is disproportionally false negatives; facing the open space was preferable. We then do a ping of each wall to compete our ultrasonic localisation. We found that the readings of the sensor are inaccurate at short distances so these are approximate.

The threshold calculation only falls on discrete intervals controlled by `getFilteredData` (50 ms.) divided by the speed of rotation ($50°s^{-1}$.) An improvement would be to wait until the sensor fell above or below the threshold, then take the previous value and compute a linear intercept to get a more accurate value.

It then moves to $(-3, -3)$ and turns to 45° to begin it's light sensor localisation as in the lab specifications[**?**]. When entering the light sensor localisation, the orientation must be in the first quadrant.

There were a number of problems with the provided code that cost us a large amout of time. The class structure, for instance, was rather poor. There

θ_{start} (°)	θ_{final} (°)	θ_{reported} (°)	θ_{error} (cm)
45	359	1.8366	-2.8366
0	359	0.9664	-1.9664
315	0	1.6627	-1.6627
270	0	1.8364	-1.8364
225	359	1.4885	-2.4885
180	359	1.9662	-2.9662
135	0	1.8364	-1.8364
90	359	1.0112	-2.0112
345	0	1.0104	-1.0104
15	359	2.1846	-3.1846

Table 1: Facing out using `LocalizationType.FALLING_EDGE`. θ_{start} is the starting orientation of the robot. The error mean is -2.1799, variance is 0.46, and the corrected sample standard deviation is 0.67.

θ_{start} (°)	θ_{final} (°)	θ_{reported} (°)	θ_{error} (cm)
45	358	358.2120	-0.2120
0	1	358.5603	2.4397
315	0	358.4330	1.5670
270	0	358.9086	1.0914
225	0	358.5604	1.4396
180	358	358.4733	-0.4733
135	359	358.3862	0.6138
90	358	359.2564	-1.2564
345	0	358.4733	1.5267
15	0	358.2121	1.7879

Table 2: Facing the wall using `LocalizationType.RISING_EDGE`. θ_{start} is the starting orientation of the robot. The error mean is 0.8524, variance is 1.4, and the corrected sample standard deviation is 1.2.

was additional code provided by one of the TA's that was well written, but we did not realise this until after the code was started.

2.2 Questions

1. If it means rising edge or falling edge, see the discussion above.

   The light sensor localization is by far a more accurate method than ultrasonic localization. This makes sense because of the simplicity of the method. There is little room for error with light sensor localization if it is performed correctly (explained in the answer to question two below). On the other hand the ultrasonic sensor is very susceptible to noise and the position at which the robot is placed can affect its ability to localize.

   For example, we noticed when the robot was placed very close to the wall near the corner the ultrasonic sensor would read values that were higher than they were actually were. This could be caused by a number of factors such as reflection of sound waves off the wall perpendicular to the wall the robot is facing (and other types of error caused by reflection), detection of other ultrasonic waves in the environment that might be concentrated in the corners of the field when they reflect off the two perpendicular walls, or simply a fault in the ultrasonic sensor's design. To address this non-ideal behaviour, we experimented with the static attribute variable for the distance of the sensor from the centre of robot. Humidity and air pressure are also a factor because it affects the speed sound waves travel through the air. This might not introduce a significant error on most days, but could explain variation in the effectiveness of ultrasonic localization over different days.

A calibration of the sensor before the program runs by measuring a known distance could also be a solution. Similarly, the light sensor readings are affected by the level of ambient lighting in the room, and this can similarly be corrected by doing a calibration before the robot runs or by measuring change in values. In this lab, the lighting seemed to be constant, and we did not have a problem with this.

2. The light sensor is not as susceptible to noise, thus it provides greater accuracy in localisation of the robot.

   The contrast between the black lines and tan/yellow board allows the light sensor to detect a sudden change from light to dark as the robot rotates over the line. The lines are thick enough to easily detect, but narrow enough that when the robot reads dark it can estimate its position with relation to the lines along the given axis to within a few millimeters. The fact that the lines are straight, perpendicular, and equally spaced makes it very easy for the robot to determine its position within a square with accuracy. However, this knowledge is useless if the robot does not know which square it is in, and that is the purpose of the ultrasonic localization. Using only the light sensor it is impossible for the robot to determine which square it is in because it does not know where the edges of the field are. However, once the ultrasonic sensor get the robot within a known square, the light sensor can determine the robots position to within half a centimeter. Accuracy could probably be improved to a millimeter by introducing optimizations such as detecting rising and falling edges and using the average to get the center of the line. The ultrasonic sensor is less effective and only used for approximating the robots position because of noise, which drastically reduces its effectiveness. Additionally, the differing sound wave absorption properties of different materials, such as the wooden walls, and the variation in absorption properties within a single material introduce more error.

3. Turn all around and record the ultrasonic sensor readings. The minima of the result will be the closest things to it; in this case, we expect to have two minima within 30 cm corresponding to the two walls. However, the ultrasonic sensor is very imprecise and is only good up to a few cm. We may get several minima close to the real minima. We could apply an averaging filter to reduce the noise, but this blocks out small objects which we would otherwise detect.

3 Error Calculations

The following are falling edge calculations.

Calculate the differences in Equation 2–11.

$$d_1 = ((359) - (1.8366))_{360[-180,180]} = -2.8366 \tag{2}$$

$$d_2 = ((359) - (0.9664))_{360[-180,180]} = -1.9664 \tag{3}$$

$$d_3 = ((0) - (1.6627))_{360[-180,180]} = -1.6627 \tag{4}$$

$$d_4 = ((0) - (1.8364))_{360[-180,180]} = -1.8364 \tag{5}$$

$$d_5 = ((359) - (1.4885))_{360[-180,180]} = -2.4885 \tag{6}$$

$$d_6 = ((359) - (1.9662))_{360[-180,180]} = -2.9662 \tag{7}$$

$$d_7 = ((0) - (1.8364))_{360[-180,180]} = -1.8364 \tag{8}$$

$$d_8 = ((359) - (1.0112))_{360[-180,180]} = -2.0112 \tag{9}$$

$$d_9 = ((0) - (1.0104))_{360[-180,180]} = -1.0104 \tag{10}$$

$$d_{10} = ((359) - (2.1846))_{360[-180,180]} = -3.1846 \tag{11}$$

Calculate the sum of the differences (Equation 2–11) in Equation 12.

$$\begin{aligned}\text{sum} &= \sum_{i=1}^{10} d_i \\ &= (-2.8366)+ \\ &\quad (-1.9664)+ \\ &\quad (-1.6627)+ \\ &\quad (-1.8364)+ \\ &\quad (-2.4885)+ \\ &\quad (-2.9662)+ \\ &\quad (-1.8364)+ \\ &\quad (-2.0112)+ \\ &\quad (-1.0104)+ \\ &\quad (-3.1846) \\ &= -21.7994 \end{aligned} \tag{12}$$

Calculate the sum of the differences (Equation 2–11) squared in Equation 13.

$$\begin{aligned}\text{ssq} &= \sum_{i=1}^{10} {d_i}^2 \\ &= (-2.84)^2+ \\ &\quad (-1.97)^2+ \\ &\quad (-1.66)^2+ \\ &\quad (-1.84)^2+ \\ &\quad (-2.49)^2+ \\ &\quad (-2.97)^2+ \\ &\quad (-1.84)^2+ \\ &\quad (-2.01)^2+ \\ &\quad (-1.01)^2+ \\ &\quad (-3.18)^2 \\ &= 51.62 \end{aligned} \tag{13}$$

Calculate the mean from Equation 12 in Equation 14.

$$\begin{aligned}\text{mean} &= \frac{\text{sum}}{N} \\ &= \frac{-21.7994}{10} \\ &= -2.179940 \end{aligned} \tag{14}$$

Calculate the variance from Equation 12 and 13 in Equation 15.

$$\begin{aligned}\sigma^2 &= \frac{\text{ssq} - \frac{\text{sum}^2}{N}}{N-1} \\ &= \frac{51.6208 - \frac{-21.7994^2}{10}}{10-1} \\ &= 0.455492 \end{aligned} \tag{15}$$

Calculate the corrected sample standard deviation from the variance (Equation 15) in Equation 16.

$$\begin{aligned}\sigma &= \sqrt{\sigma^2} \\ &= \sqrt{0.455492} \\ &= 0.674902 \end{aligned} \tag{16}$$

This is facing the wall with rising edge.
Calculate the differences in Equation 17–26.

$$\begin{aligned} d_1 &= ((358) - (358.2120))_{360[-180,180]} \\ &= -0.2120 \end{aligned} \tag{17}$$

$$\begin{aligned} d_2 &= ((1) - (358.5603))_{360[-180,180]} \\ &= 2.4397 \end{aligned} \tag{18}$$

$$\begin{aligned} d_3 &= ((0) - (358.4330))_{360[-180,180]} \\ &= 1.5670 \end{aligned} \tag{19}$$

$$\begin{aligned} d_4 &= ((0) - (358.9086))_{360[-180,180]} \\ &= 1.0914 \end{aligned} \tag{20}$$

$$\begin{aligned} d_5 &= ((0) - (358.5604))_{360[-180,180]} \\ &= 1.4396 \end{aligned} \tag{21}$$

$$\begin{aligned} d_6 &= ((358) - (358.4733))_{360[-180,180]} \\ &= -0.4733 \end{aligned} \tag{22}$$

$$\begin{aligned} d_7 &= ((359) - (358.3862))_{360[-180,180]} \\ &= 0.6138 \end{aligned} \tag{23}$$

$$\begin{aligned} d_8 &= ((358) - (359.2564))_{360[-180,180]} \\ &= -1.2564 \end{aligned} \tag{24}$$

$$\begin{aligned} d_9 &= ((0) - (358.4733))_{360[-180,180]} \\ &= 1.5267 \end{aligned} \tag{25}$$

$$\begin{aligned} d_{10} &= ((0) - (358.2121))_{360[-180,180]} \\ &= 1.7879 \end{aligned} \tag{26}$$

Calculate the sum of the differences (Equation 17–26) in Equation 27.

$$\begin{aligned} \text{sum} &= \sum_{i=1}^{10} d_i \\ &= (-0.2120)+ \\ &\quad (2.4397)+ \\ &\quad (1.5670)+ \\ &\quad (1.0914)+ \\ &\quad (1.4396)+ \\ &\quad (-0.4733)+ \\ &\quad (0.6138)+ \\ &\quad (-1.2564)+ \\ &\quad (1.5267)+ \\ &\quad (1.7879) \\ &= 8.5244 \end{aligned} \tag{27}$$

Calculate the sum of the differences (Equation 17–26) squared in Equation 28.

$$\begin{aligned} \text{ssq} &= \sum_{i=1}^{10} {d_i}^2 \\ &= (-0.21)^2+ \\ &\quad (2.44)^2+ \\ &\quad (1.57)^2+ \\ &\quad (1.09)^2+ \\ &\quad (1.44)^2+ \\ &\quad (-0.47)^2+ \\ &\quad (0.61)^2+ \\ &\quad (-1.26)^2+ \\ &\quad (1.53)^2+ \\ &\quad (1.79)^2 \\ &= 19.42 \end{aligned} \tag{28}$$

Calculate the mean from Equation 27 in Equation 29.

$$\begin{aligned} \text{mean} &= \frac{\text{sum}}{N} \\ &= \frac{8.5244}{10} \\ &= 0.852440 \end{aligned} \tag{29}$$

Calculate the variance from Equation 27 and 28 in Equation 30.

$$\begin{aligned} \sigma^2 &= \frac{\text{ssq} - \frac{\text{sum}^2}{N}}{N-1} \\ &= \frac{19.4229 - \frac{8.5244^2}{10}}{10-1} \\ &= 1.350704 \end{aligned} \tag{30}$$

Calculate the corrected sample standard deviation from the variance (Equation 30) in Equation 31.

$$\begin{aligned} \sigma &= \sqrt{\sigma^2} \\ &= \sqrt{1.350704} \\ &= 1.162198 \end{aligned} \tag{31}$$

4 Further Improvements

1. The clipping filter is a very rudimentary method for removing extraneous values because it only accepts values from the sensor that are within a certain range, and all values outside this range are ignored. A more accurate method of removing small errors is to consider them with respect to the rest of the values returned by the sensor. A single extraneous value should be ignored, but multiple extraneous values mean that a wall the robot was sensing could have suddenly ended. To accomplish this, the robot should process multiple sensor values at a time or have a certain amount of memory to store sensor values. This could be done by median filtering or removing outliers.

2. Radio waves as active radar is the *de facto* method of detection. However, radar is long-range, over-the-horizon, which we don't need. An optical laser would be more accurate and have a faster response time and would be probably ideal for this use.

   The ultrasonic sensor receives much of its error from interference and from its own sound waves

bouncing off multiple objects or objects to the side of the sensor and then returning to it. A very simple fix that addresses all of these problems is to narrow the scope of the sensor so it only sends sound waves straight out in front of it instead of in all directions. This can be accomplished by wrapping some type of material around the sensor that focuses the waves out the front of the sensor. Cardboard or paper could be used for this. The problem with this is diffraction[**?**] which causes it to be scattered.

3. We could use an adaptive method that remembered the lines on the floor and constantly fit using an iterative method to what it knows the world looks like, ie three tiles square.

   Another possible form of ultrasonic localization is to use the method described in question three in the Observations and conclusions section by comparing minimum values. To do this the ultrasonic sensor would need to not report extraneous minimums. There could be filtered out using the method described above in Question 3 of this section. It would also work anywhere in the field, not just when the robot is placed in the corner tile.