ECSE 211 Winter 2020

Lab 2: Odometry

Group 54

Joanna Koo (260865846)

Muhammad Mustafa Javed (260808710)

Due January 30th, 2020

Table of Contents

Section 1: Design Evaluation

1.1 Design workflow

The objective was to design and implement an odometry that provides the robot's position and orientation from a fixed reference point. We started by understanding how the starter code was designed and what alterations were needed to adjust the code to our robot. After taking careful measurements, we tested and tuned the values of the wheelbase and the radius of tires to 12.1 cm and 2.1 cm respectively. Next, we carried out and analysed ten independent trials to evaluate the performance of the odometer.

1.2 Hardware design overview

The robot design used for Lab 1 was optimal for sharp turns and rectilinear motion. We made slight modifications to the robot for Lab 2. The ultrasonic sensor was removed as it was no longer needed. During the testing, it was observed that the wheelbase was a very sensitive measurement, and the strain on the axles during the turns was causing minor changes in the wheelbase radius. To overcome this, wheel rims and a strengthening rod was added to the wheelbase to improve the rigidity of the wheelbase. In order to get a better sense of the orientation of the robot, a front sight was added, which helped to verify the alignment of the robot with the back tri-wheel. The robot design can be found in *Figures 1, 2 and 3*.

Figure 1. Bird's eye view of the robot.

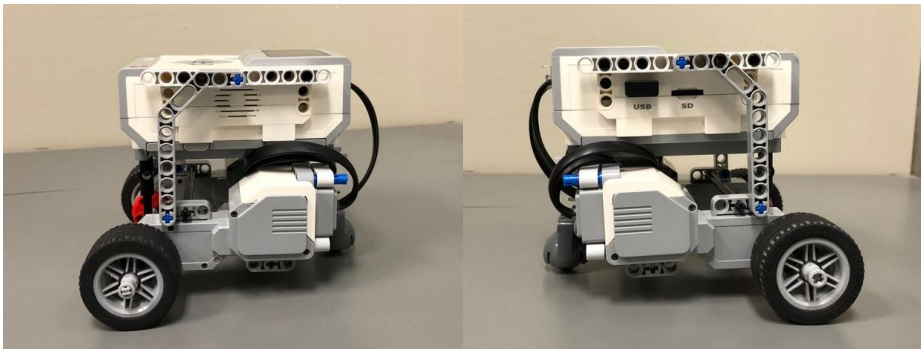

Figure 2. Left and right side profiles of the robot.

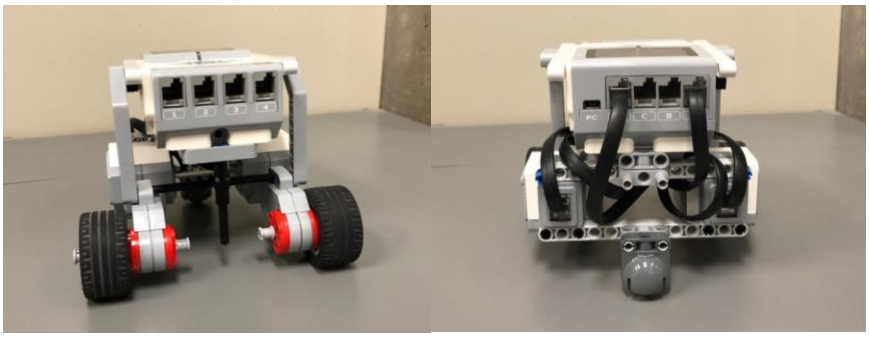

Figure 3. Front and back profiles of the robot.

1.3 Software overview

The code package consisted of five classes. The Main class controls the operation of the program. The Odometer class calculates the positions of X and Y by adding up the incremental distance moved in X and Y directions. To calculate the incremental distance, the odometer counts the rotation of the left and right motor by counting the tachometer count in a set-time period. The distance travelled is equal to the rotation times 2π times radius of the wheel as shown in *Equation 1*. If the robot is travelling at an angle, the distance in X-component is the sine of distance travelled, while the distance in Y-component is the cosine of the distance travelled, as calculated in *Equation 2*. If one of the wheels is rotating faster than the other, the odometer determines that the robot is turning and calculates the Theta value, by dividing the arc length by the wheelbase radius as in *Equation 3*. On the other hand, the SqaureDriver class works by calculating how many degrees it has to rotate the wheel for the robot to travel some distance and/or turn to a particular orientation. The flowchart illustrating the software functionality is shown in *Figure 4*.

Equation 1.

$$Motor\ rotation\ in\ degrees = \frac{(nowTachoCount - Last\ TachoCount\ )}{360}$$

$$LeftDistance = 2\pi * r * (rotation\ of\ the\ left\ motor\ in\ degrees)$$

$$RightDistance = 2\pi * r * (rotation\ of\ the\ right\ motor\ in\ degrees)$$

Equation 2.

$$Distance = \frac{(LeftDistance + RightDistance)}{2}$$

$$dx = Distance * sin\ \theta$$

$$X = X + dx$$

$$dy = Distance * \cos\theta$$

$$Y = Y + dy$$

Equation 3.

$$\Delta\theta = \frac{(leftDistance - RightDistane)}{WheelBase}$$

$$\theta = \theta + \Delta\theta$$

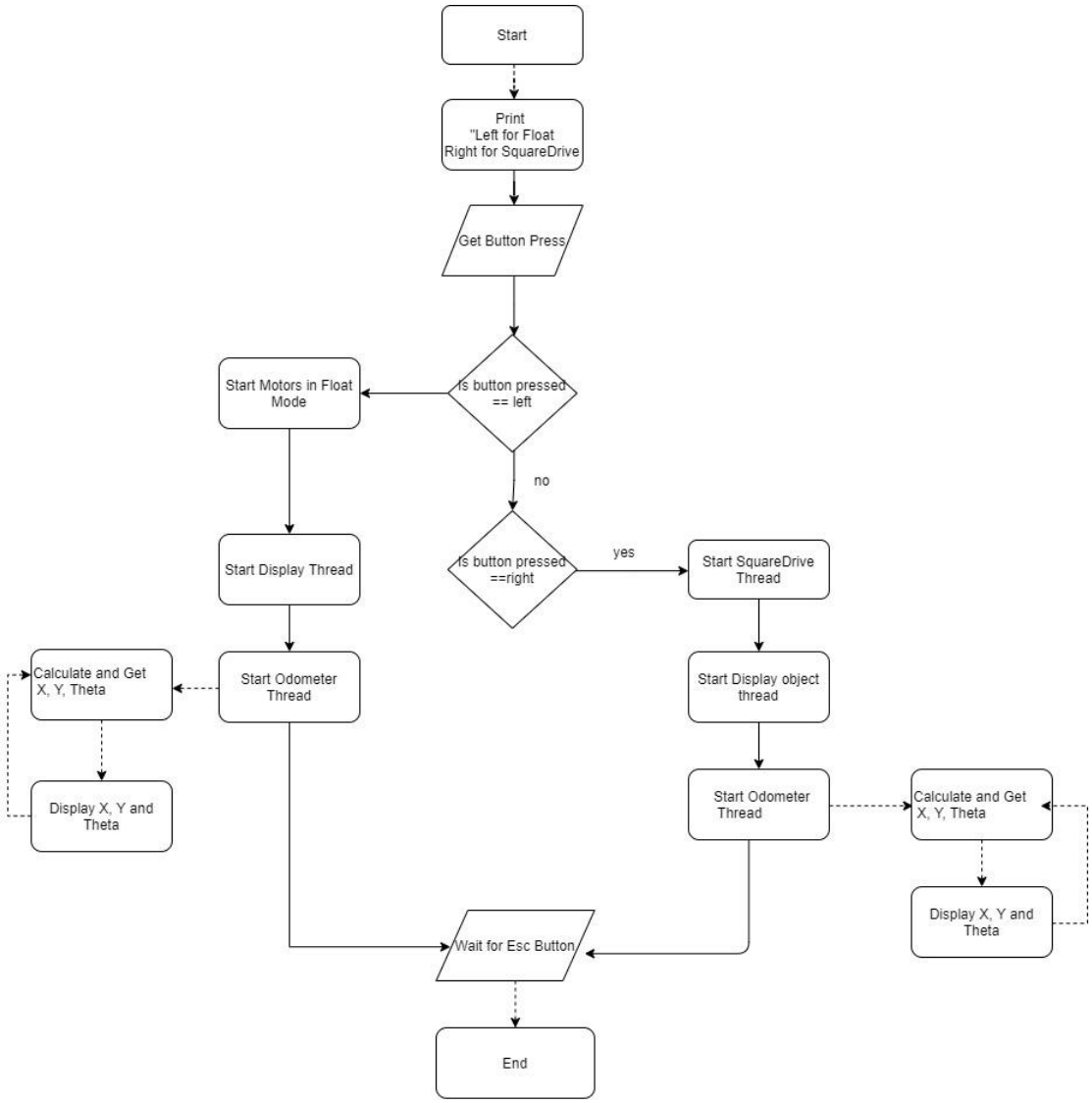

Figure 4. Flowchart for software functionality.

Section 2: Test Data

10 independent trials were performed for the odometry test and is recorded in *Table 1*. The robot was run in a 3-by-3 square using SquareDriver.java. The starting position S was treated as (0,0).

Trial	Measured X_F (cm)	Measured Y_F (cm)	Reported X (cm)	Reported Y (cm)
1	0.0	0.0	0.0	1.0
2	-0.5	1.0	-1.0	1.0
3	0.0	0.0	0.0	1.0
4	-1.0	3.5	0.0	1.0
5	1.0	3.5	0.0	1.0
6	2.0	1.0	0.0	0.0
7	-1.0	0.0	-1.0	2.0
8	-3.0	2.5	-1.0	1.0
9	2.0	1.0	-1.0	1.0
10	0.0	4.0	0.0	1.0

Table 1. Ten independent trials of the odometry test.

Section 3: Test Analysis

The Euclidean error distance ϵ is computed in *Table 2*. The following equation was used for calculations.

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

Trial	Euclidean error distance ϵ (cm)
1	1.0
2	0.5
3	1.0
4	2.7
5	2.7
6	2.2
7	2.0
8	2.5
9	3.0
10	3.0

Table 2. Euclidean error distance ϵ of the position for tests.

The mean and standard deviation for X, Y, and ϵ is computed in *Table 3*. The following equations were used for calculations.

$$\bar{a} = \frac{1}{n}\sum_{i=1}^{n} a_i$$

$$\sigma_a = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (a_i - \bar{a})^2}$$

	X (cm)	Y (cm)	ϵ (cm)
Mean	-0.4	1.0	2.1
Standard deviation	0.5	0.4	0.9

Table 3. Mean and standard deviation for tests.

The sample mean calculation for the X values is shown below.

$$\bar{X} = \frac{1}{10}\big(0.0 + (-1.0) + 0.0 + 0.0 + 0.0 + 0.0 + (-1.0) + (-1.0) + (-1.0) + (0.0)\big) = -0.4$$

The sample standard deviation calculation for the X values is shown below.

$$\sigma_X = \sqrt{\frac{1}{10}\big(0.0 - (-0.4)\big)^2 + \big((-1.0) - (-0.4)\big)^2 + \big(0.0 - (-0.4)\big)^2 + \ \ldots\ + (0.0 - (-0.4))^2} = 0.5$$

What does the standard deviation of X,Y and ϵ tell you about the accuracy of the odometer? What causes changes in standard deviation?

As seen in *Table 3*, the standard deviation of x and y values, 0.5 cm and 0.4 cm respectively, are relatively small, considering the fact that the robot traverses a 3-by-3 tile trajectory of 91.44 cm in width. The standard deviation of the error value, 0.9 cm, is also relatively small, although the value is slightly larger than the deviation for x and y. This is possibly due to the combinations of different x and y values that caused a greater variety of error values. These small standard deviation values demonstrate the small dispersion of reported data, showing consistent readings produced by the odometer. The change in standard deviation can occur by modifying the size of the grid, as the data collected from the trials run on a larger grid will more likely produce larger standard deviation values for x, y and ϵ.

What is the sampling frequency of your odometer (i.e. the frequency at which the tachocount is measured)? What is the tradeoff of having a high sampling frequency versus a low sampling frequency?

The ODOMETER_PERIOD in the Odometer class is set to 25 milliseconds. The sampling frequency can be calculated by dividing 1 by the period, which is 40 hertz. By having a higher sampling frequency, the odometer takes measurements more frequently, leading to more accurate measurement values of x and y. However, this leads to the thread in the Odometer class running for a longer period of time, reducing the time for the other threads to execute.

Section 4: Observations and Conclusions

Is the error you observed in the odometer tolerable for larger distances? What happens if the robot travels 5 times the 3-by-3 grid's distance?

The error observed in the odometer for larger distances is not tolerable. During multiple trials, we noticed that the robot did not turn at exactly 90 degrees at the corners, rather it was slightly off by a few degrees. Furthermore, the robot slightly shifted to the left when travelling straight. With a short path, the accumulated error was not as significant when the robot completed the trajectory. However, if the robot were to travel 5 times the 3-by-3 grid's distance, the accumulated error would be large enough that the robot fails to drive in a perfect square. As a result, the robot would fall off the board during the run or be extremely misaligned at the finish position, causing the odometer readings highly inaccurate.

Do you expect the odometer's error to grow linearly with respect to travel distance? Why?

We expect the odometer's error to grow linearly with respect to travel distance, due to the minor variations in the values entered as WHEEL_RAD and BASE_WIDTH, representing the radius of the wheel and the width of the robot. This is the main source of error, as it leads to inaccurate distance calculations. During each thread run in the Odometry class, calculations are performed and error is added from the new tacho-count whereas the previous error is discarded. Therefore, the error increases linearly as the thread goes through the run method.

Section 5: Further Improvements

Propose a means of reducing the slip of the robot's wheels using software.

In order to prevent the slip of the robot's wheels, the acceleration of the motors can be set to a lower value that will ensure that the force by the wheels on the floor is less than dynamic friction. A lower speed during the turn will also reduce the slip when the robot rotates at its position.

Propose a means of improving the accuracy of the odometer using one or more light sensors.

Two light sensors can be attached to the front-right and front-left of the robot. This will enable the robot to detect the black lines on the floor when crossing over a tile. If the robot is going straight, the two sensors would detect the line at the same time or with negligible time difference. However, if there is a noticeable time lag between the detections from the sensors, it would mean that the robot encountered an orientation error due to the drift during its rectilinear motion. If the right sensor detects the line first, the right motor would stop until the left motor catches up and the left sensor detects the line. Similarly, if the left sensor detects the line first, the left motor would stop until the right motor catches up. The odometer would be updated accordingly to correct orientation.