ECSE 211 Lab 3

Navigation & Obstacle Avoidance

Group 65

Kathy Yim 260741794

Hanwen Wang 260778557

Design Evaluation

The goal of this lab is to design and implement a navigation system as well as an obstacle avoidance function, which allows the robot to move between specific points and avoid obstacles on a map.

Hardware Design: The robot in this lab is composed of three main components: two motors and an ultrasonic sensor. The ultrasonic sensor is placed in the middle of two motors. In addition, the two motors are placed on the sides of the EV3 brick rather than directly underneath to increase its stability when turning waypoints. We also made use of the metal ball fixed at the back of the EV3 for increased stability and mobility. Furthermore, we placed a joint between each wheel and motor to avoid friction between them. Shown in figure 1 below is an overview of the hardware design of our robot.

Figure 1: Hardware Design Overview

Software Design: In the implementation of robot navigation, we used two main methods: travelTo() and turnTo(). TurnTo allows the robot to turn at a smaller angle and travelTo allows the robot to move to a specific point on the map as indicated. In the flowchart below, current x is the horizontal value of the robot in current position and current y is the vertical position of the robot in current position.

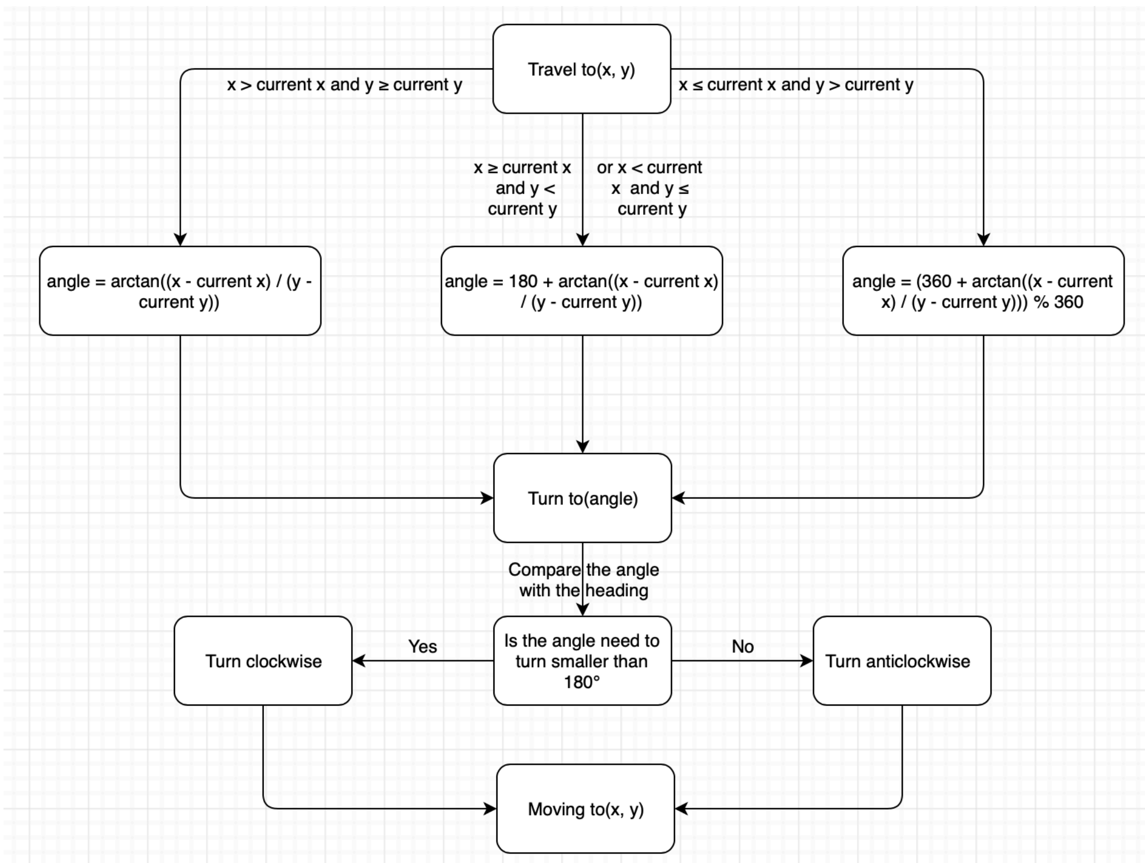

Figure 2: Navigation Flowchart

For our obstacle avoidance implementation, we need to read the value measured by ultrasonic sensor, and this informs us of whether the sensor detected an obstacle and the steps the robot needs to take to avoid this obstacle. If the distance detected is smaller than 15cm, the following flowchart is followed to help the robot avoid the obstacle in its path.

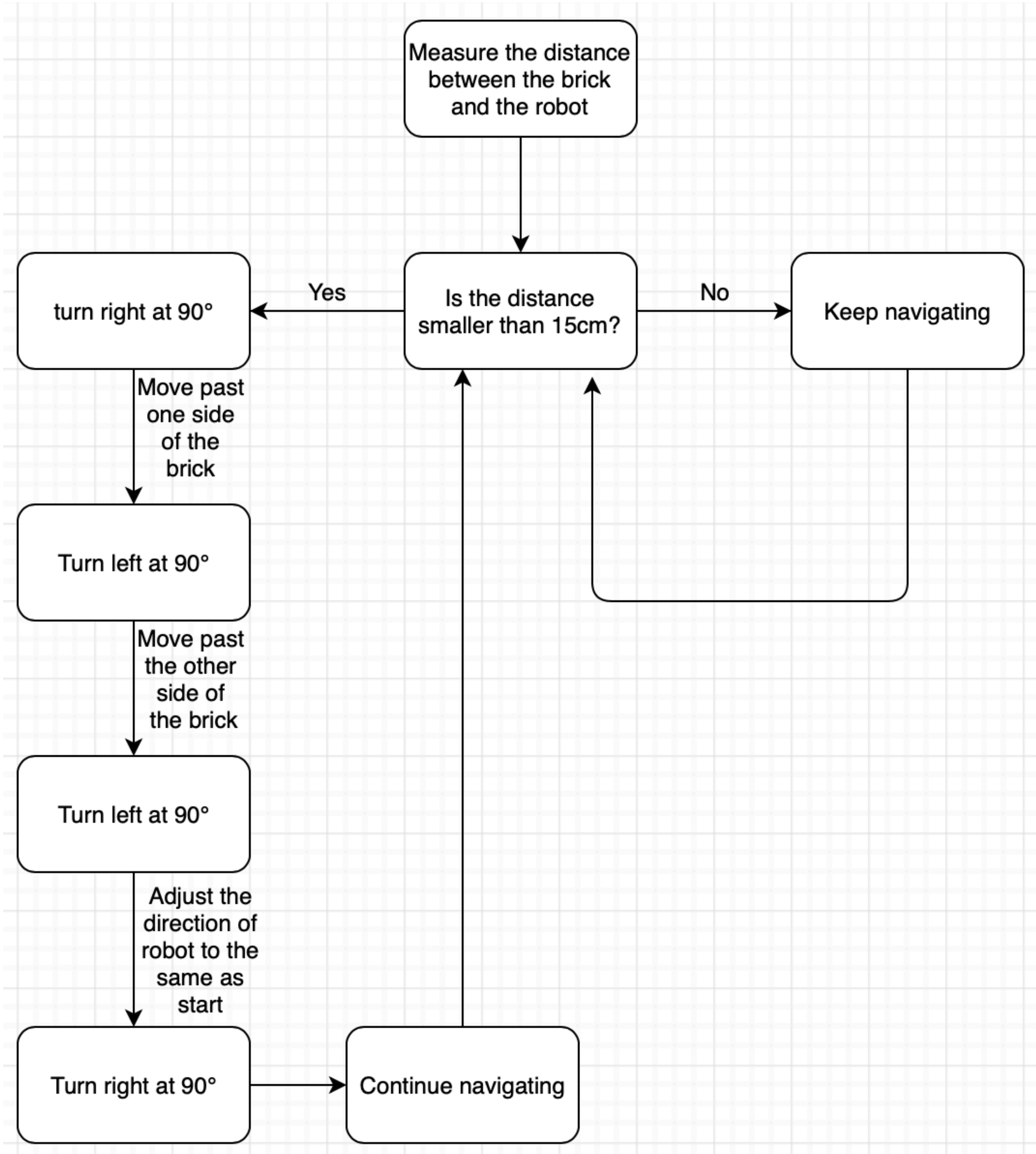

Figure 3: Obstacle Avoidance Flowchart

Test Data

Destination x/cm	Destination y/cm	Final position x/cm	Final position y/cm	Error ∈/cm
60.96	0	59.96	0.20	1.01
60.96	0	59.86	0.35	1.15
60.96	0	59.66	0.45	1.38
60.96	0	59.76	0.50	1.30
60.96	0	59.86	0.30	1.14
60.96	0	59.16	0.80	1.97
60.96	0	58.96	0.40	1.08
60.96	0	59.06	0.75	2.04
60.96	0	59.16	0.90	2.01
60.96	0	59.06	0.85	2.08

Table 1: Simple Navigation

Test Analysis

Mean value = $\frac{\sum_{i=1}^{i=N} \in(i)}{N}$

Standard deviation = $\sqrt{\frac{\sum_{i=1}^{i=N} (\in(i) - Mean\ value)^2}{N}}$

Mean ∈/cm	**Standard Deviation ∈/cm**
1.513	0.195

Observations and Conclusions

Are the errors you observed due to the odometer or navigator? What are the main sources?

Errors we observed are mainly due to the odometer because of the friction of the wheels. If one wheel is covered with dust and the other one is clean, they might not move at the same speed and turn by the same angle even if we set these values to be the same. Furthermore, the slight difference in turning angle due to friction will gradually deviate the robot from its projected path. This deviation will prevent the odometer from detecting real value, therefore causing the robot to move incorrectly. Since the robot is expected to turn and navigate around several corners in this lab, the small turning errors will eventually accumulate and result in a more significant error at the end of its travel.

How accurately does the navigation controller move the robot to its destination?

The mean error is around 1.5cm and the standard deviation is very small, this means that the navigation controller is quite accurate in our lab.

How quickly does it settle (i.e. stop oscillating) on its destination?

The robot never oscillates during its whole travel path since nothing is adjusted, the robot just directly heads to the next destination.

How would increasing the speed of the robot affect the accuracy of your navigation?

Increasing the speed will also increase the slip of wheels, which might cause larger errors and decrease the accuracy of our navigation. We tried three different motor speeds during the lab: 100, 150 and 200. The robot deviated more from its path as we increased the speed, so we decided to use 100 at last to increase accuracy.

Further Improvements

Hardware: We can increase the weight of the robot, and with larger mass, the robot's center of mass will be closer to the ground there decreasing the likelihood of slipping when moving. In addition, we can add a light sensor to the robot, so it can adjust its x and y values of every time it meets a grid line. Then the robot can adjust the angle it needs to turn and the direction it moves in.

Software: To reduce the errors using software, we will need to force the values of x and y every time when the robot meets a grid line. This step is similar to Odometer correction in the former lab. As well, we have to slow down the speed of the motors as mentioned above in order to reduce slip of the wheels.