**Design evaluation:**

**Workflow**

After reading the requirements for the navigation and avoidance lab, the first step was to redesign the hardware of the robot. The color sensor was taken off and the ultrasonic sensor was put back. Then, since no sample code was provided, it had to completely designed from the bottom up. Numerous class like the odometer, the ultrasonic poller and the display class were reused. Two new major class were created: navigation and obstacle. Navigation was first coded because it accomplishes the main task of navigating from one point to another. Several tests and modifications were then performed to make sure the robot was able to arrive at the destination waypoint within 2 cm using a Euclidean distance measure. Following navigation, the obstacle class responsible for avoiding obstacles was coded. Once again, a copious amount of testing and modifications were done to make sure the software design was functional.

**Test Data:**

**Navigation test**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trials | XD | YD | XF | YF |
| 1 | 30.48 | 0 | 31.68 | 1.6 |
| 2 | 30.48 | 0 | 30.88 | 1.9 |
| 3 | 30.48 | 0 | 31.98 | 1.2 |
| 4 | 30.48 | 0 | 29.18 | -1.1 |
| 5 | 30.48 | 0 | 30.38 | -0.9 |
| 6 | 30.48 | 0 | 29.18 | -1.2 |
| 7 | 30.48 | 0 | 32.28 | -0.6 |
| 8 | 30.48 | 0 | 31.88 | -1.3 |
| 9 | 30.48 | 0 | 29.78 | 0 |
| 10 | 30.48 | 0 | 30.98 | 0 |

**Table 1: Destination waypoints (XD, YD) and final positions of the robot (XF, YF) for the navigation test**

**Test Analysis:**

|  |  |
| --- | --- |
| Trials | Euclidean error (𝛜) |
| 1 | 2 |
| 2 | 1.941649 |
| 3 | 1.920937 |
| 4 | 1.702939 |
| 5 | 0.905539 |
| 6 | 1.769181 |
| 7 | 1.897367 |
| 8 | 1.910497 |
| 9 | 0.7 |
| 10 | 0.5 |

**Table 2 – Euclidean error distances for the navigation test**

|  |  |
| --- | --- |
|  | Euclidean error (𝛜) |
| Mean | 1.524811 |
| Standard Deviation | 0.582096 |

**Table 3 – Mean and standard deviation for the Euclidean error distances of the navigation test**

**Formulas and sample calculations**

Euclidean error distance:

Using data from trial 1 of the navigation test:

Mean of the Euclidean error distances:

Standard deviation of the Euclidean standard distances:

**Observations and conclusion**

The errors observed were mostly due to the odometer. Since the color sensor was removed for this lab, odometry correction was impossible. As a result, the robot was especially vulnerable to the slipping of the wheels and such causes of errors as it was unable to verify if it had the right coordinates. Furthermore, according to the coordinates shown by the odometer, the robot is where it is supposed to be, which shows that the errors are not caused by the navigation controller.

Looking at the average value of the Euclidean error distances 1.524811 cm, the navigation controller moves the robot accurately. The highest error distance obtained during the testing was 2 cm, which is still accurate according to the requirements of the lab.

The robot took no time to settle to its destinations. In fact, there was no oscillating while the robot travelled between points either. Since the software design does not implement any correction of any kind, the robot always assumes it is on the right path to its destination. It is therefore particularly vulnerable to causes of errors such as slipping or low battery levels. On the other hand, it never oscillates and is very efficient in a controlled environment where causes of errors such as the ones previously mentioned are minimal.

Increasing the speed of the of the robot would negatively affect its accuracy. One of the biggest cause of errors is the slipping of the wheels and increasing the speed would only increase the chance of slipping. Furthermore, a higher speed does not improve the accuracy in any way. It could therefore only affect the accuracy negatively.

**Further improvements**

A software improvement that could reduce one of the major causes of error that is slipping would be to implement slow acceleration and deceleration. Indeed, slipping mostly occurs when the robot goes from immobile to mobile. Currently, the sudden acceleration of the robot to the desired speed is particularly propitious to that phenomenon. Implementing a gradual acceleration and deceleration instead of an instant one would diminish the probability of slipping and consequently improve accuracy. Another software improvement could be to implement the wall following system developed during the first lab into this one. Currently, when the robot detects an obstacle, it turns to a predetermined angle and starts a manoeuvre to avoid the obstacle. This manoeuvre is the same no matter the size of the object and therefore only works for obstacle of the same size or smaller than the wooden blocks available in the lab. Implementing the wall follower controllers would allow the robot to avoid obstacles of any size and shape.

One major hardware improvement would be to add back one or two color sensors to perform odometry correction during the navigation. This would improve the accuracy of the robot by adjusting faulty coordinates caused by slipping and such errors. Another hardware improvement could be to attach the ultrasonic sensor to a motor. This motor would allow the sensor to rotate and enlarge its angle of vision, which in turn allows the robot to make sure it has cleared the obstacle after maneuvering around it. It could also help it decide from which direction it should get around the obstacle. For example, after stopping in front of an object, the sensor could rotate left and right to determine if one side would necessitate a smaller manoeuvre. This shorter manoeuvre minimizes the probability of error happening and therefore maximize the probability of accurately arriving at destination.