Test document

1. Navigation/Odometer class

Description of the test: The robot starts at the center of a tile (0,0), facing 0 degrees. Using the code below, the robot navigates to four points and returns to the origin (0,0). The goal of this test is to determine if the robot's position and heading after a few displacements are accurate and how far off they are from the odometer's lecture.

code:

nav.travelTo(61,61);

nav.travelTo(61,0);

nav.travelTo(0,61);

nav.travelTo(0,0);

nav.turnTo(findAngle(0)); //To turn to "absolute" 0 degrees

Table 1 - Raw data for the class Navigation

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Robot's X position (cm)** | **Robot's Y position (cm)** | **Robot's heading (degrees)** | **Odometer's X (cm)** | **Odometer's Y (cm)** | **Odometer's heading (degrees)** | **X error (cm)** | **Y error (cm)** | **Heading error (degrees)** |
| 3.4 | 1.1 | -4 | 0.29 | 0.69 | -3 | 3,1 | 0,4 | -1 |
| 3.7 | 0.4 | -3 | 0.34 | 0.65 | -2 | 3,4 | -0,3 | -1 |
| 3.2 | 0.7 | -5 | 0.31 | 0.68 | 0 | 2,9 | 0,0 | -5 |
| 3.4 | 1,0 | -5 | 0.37 | 0.68 | -1 | 3,0 | 0,3 | -4 |
| 3.0 | 1,0 | -3 | 0.31 | 0.69 | 1 | 2,7 | 0,3 | -4 |
| 2.9 | 0.8 | -5 | 0.35 | 0.68 | -2 | 2,6 | 0,1 | -3 |
| 3.1 | 0.5 | -4 | 0.37 | 0.64 | -2 | 2,7 | -0,1 | -2 |
| 3.4 | 0.6 | -6 | 0.29 | 0.65 | -4 | 3,1 | -0,1 | -2 |
| 3.5 | 0.7 | -5 | 0.32 | 0.68 | -3 | 3,2 | 0,0 | -2 |
| 3.2 | 0.6 | -6 | 0.34 | 0.67 | -4 | 2,9 | -0,1 | -2 |

Table 2 - Average of the errors for the class Navigation

|  |  |
| --- | --- |
| **Average X error** | 3,0 |
| **Average Y error** | 0,1 |
| **Average Heading error** | -2,6 |

Table 3 - Standard deviation of the errors for the class Navigation

|  |  |
| --- | --- |
| **Standard deviation of X error** | 0,24 |
| **Standard deviation of Y error** | 0,20 |
| **Standard deviation of Heading error** | 0,5 |

Interpretation of the test results: The average error for the position and the heading of the robot is relatively small, since the robot travels about 8 tiles (295 cm) and rotates 450 degrees.

The small standard deviation tells us that the data is not spread much. The errors are mostly due to the slippery nature of the floor and the weight of the robots.

This means that when our robot will travel 295 cm, it should be within the ellipse of error:

Height = 0,2 cm

Length = 6,0 cm

Improvements: Make the robot lighter (less weight on the wheels, less friction), add Play-Dough in the wheels to make them more stable, more rigid and less likely to deform.

2. USLocalizer class

Description of the test: Starting from random orientations (both falling and rising edges), the robot will do its ultrasonic localization, find its heading and then go to zero degrees. The robot final orientation will be measured, with respect to the odometer's reading.

Table 4 - Raw data for the class USLocalizer

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Falling edge** | | | **Rising edge** | | |
| **Odometer's heading (degrees)** | **Robot's heading (degrees)** | **Error (degrees)** | **Odometer's heading (degrees)** | **Robot's heading (degrees)** | **Error (degrees)** |
| 349 | 0 | 11 | 358 | 1 | 3 |
| 350 | 1 | 11 | 357 | 0 | 3 |
| 350 | 2 | 12 | 358 | 2 | 4 |
| 358 | 2 | 4 | 356 | 1 | 5 |
| 352 | 3 | 11 | 354 | 3 | 9 |
| 350 | 2 | 12 | 358 | 359 | 1 |
| 353 | 4 | 11 | 357 | 2 | 5 |
| 351 | 2 | 11 | 358 | 0 | 2 |
| 352 | 1 | 9 | 357 | 1 | 4 |
| 350 | 2 | 12 | 358 | 3 | 5 |

Table 5 - Average of the errors for the class USLocalizer

|  |  |
| --- | --- |
| **Average for the error for falling edge (degrees)** | 10,4 |
| **Average for the error for rising edge (degrees)** | 4,1 |

Table 6 - Standard deviation for the class USLocalizer

|  |  |
| --- | --- |
| **Standard deviation for the error for falling edge (degrees)** | 2,41 |
| **Standard deviation for the error for rising edge (degrees)** | 2,18 |

Interpretation of the test results: Our localization is more precise for the rising edge than the falling edge. The fact that the error averages for the test are quite high is due to the slippery nature of the floor. This is the approximation heading scheme:

Falling edge Rising edge

Angle : 20,8 degrees Angle : 8,2 degrees

Improvements: Since the standard deviation is high in comparison to the average, we can't really hardcode a constant to make up for the offsets. However, this does not matter, because this class is only used to get an approximation of the right heading, and get the robot looking at about 0 degrees. The LightLocalizer class will then take care of finding the exact heading. Reducing the rotation speed of the robot would possibly make the result more precise, but precision is not what we aim for in the execution of this class.

3. LightLocalizer class

Description of the test: The robot will do its light localization and will then go to (0,0) and turn to 0 degrees. The distance from the center and the exact heading will be measured, with respect to the odometer's reading.

Table 7 - Raw data for the class LightLocalizer

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Robot's X position (cm)** | **Robot's Y position (cm)** | **Robot's heading (degrees)** | **Odometer's X (cm)** | **Odometer's Y (cm)** | **Odometer's heading (degrees)** | **X error (cm)** | **Y error (cm)** | **Heading error (degrees)** |
| 2,5 | 1,0 | 3 | 0,42 | 0,52 | -2 | 2,08 | 0,48 | 5 |
| 3,2 | 0,9 | 4 | 0,49 | 0,33 | -2 | 2,71 | 0,57 | 6 |
| 2,9 | 1,3 | 6 | 0,22 | 1,13 | 2 | 2,68 | 0,17 | 4 |
| 2,7 | 0,0 | -2 | 0,03 | -0,76 | -3 | 2,67 | 0,76 | 1 |
| 3,4 | 1,9 | 3 | 1,21 | 1,32 | -4 | 2,19 | 0,58 | 7 |
| 0,9 | -1,1 | -4 | -0,31 | -2,34 | -4 | 1,21 | 1,24 | 0 |
| 1,3 | -0,5 | 3 | 0,24 | -1,45 | -5 | 1,06 | 0,95 | 8 |
| -0,6 | 1,9 | 2 | -1,30 | 1,48 | -4 | 0,7 | 0,42 | 6 |
| 1,4 | 2,3 | -4 | 0,03 | 3,53 | -7 | 1,37 | -1,23 | 3 |
| -1,3 | -1,5 | -5 | -2,21 | -1,22 | -6 | 0,91 | -0,28 | 1 |

Table 8 - Average of the errors for the class LightLocalizer

|  |  |
| --- | --- |
| **Average X error** | 1,758 |
| **Average Y error** | 0,366 |
| **Average Heading error** | 4,1 |

Table 9 - Standard deviation for the class LightLocalizer

|  |  |
| --- | --- |
| **Standard deviation of X error** | 0,79 |
| **Standard deviation of Y error** | 0,70 |
| **Standard deviation of Heading error** | 2,77 |

Interpretation of the test results: Our light localization is quite precise for the position, but a bit off for the heading. Since the standard deviation for the heading is high, it will be hard to fix it software-wise.

Improvements: A possible improvement would be to reduce the speed of rotation of the robot, but it would be a trade-off between speed and precision.

4. LightSearch class

Description of the test: The lightSearch class makes the robot rotate 360 degrees and seek for the brightest orientation. The test consists of aiming a light source towards the robot at 90 degrees, relative to the robot's angle. The robot will then turn 360 degrees and stop at the brightest orientation, using the line of code below. The orientation of the robot will then be measured, with two different distances between the light source and the robot.

code:

nav.turnTo(lightSearch.getBright());

Table 10 - Raw Data for LightSearch class

|  |  |  |  |
| --- | --- | --- | --- |
| **Heading of the robot (degrees)** | | | |
| **Light source at 30 centimetres** | **Error at 30 centimetres** | **Light source at 100 centimetres** | **Error at 100 centimetres** |
| 79 | -11 | 80 | -10 |
| 86 | -4 | 86 | -4 |
| 82 | -8 | 102 | 12 |
| 91 | 1 | 94 | 4 |
| 88 | -2 | 83 | -7 |
| 94 | 4 | 100 | 10 |
| 87 | -3 | 96 | 6 |
| 90 | 0 | 95 | 5 |
| 86 | -4 | 103 | 13 |
| 92 | 2 | 93 | 3 |

Table 11 - Average of the error for the class LightSearch

|  |  |
| --- | --- |
| **Average of orientation error at 30 centimetres from the source** | -2,5 |
| **Average of orientation error at 100 centimetres from the source** | 3,2 |

Table 12 - Standard deviation for the class LightSearch

|  |  |
| --- | --- |
| **Standard deviation of the orientation error at 30 centimetres from the source** | 4,58 |
| **Standard deviation of the orientation error at 100 centimetres from the source** | 7,90 |

Interpretation of the test results: The average of orientation error is low, but the standard deviation is high, so software-wise, there is not much to be done.

Improvements: We could reduce the speed of rotation of the robot. From a mechanical point of view, we could add a part to the light sensor which would act like a telescope, focusing on the light at one point.

5. ArmMovement Class

Description of the test: Using the code below, the robot's arm will go down, close its claws on the beacon and lift it up at a 90 degrees, perpendicular to the ground. The speeds of all the components are relatively slow, in order to ensure more stability and precision. This test will be done 9 times, with different orientations of the beacon, relative to the robot's claws. The success or failure of the test will be measured.

claw.setSpeed(50);

setPullerSpeed(200);

setArmSpeed(25);

puller.rotate(720,true);

leftArm.rotate(80,true);

rightArm.rotate(80,false);

nav.travelTo(40);

claw.rotate(-100,true);

Delay.msDelay(3000);

Table 13 - Raw data for the class ArmMovement

|  |  |
| --- | --- |
| **Orientation of the beacon (degrees)** | **Success or Failure** |
| 0 | Success |
| 10 | Success |
| 20 | Success |
| 30 | Success |
| 40 | Failure |
| 50 | Failure |
| 60 | Success |
| 70 | Success |
| 80 | Success |

Interpretation of test results: The robot only failed to grab while the beacon was at 40 and 50 degrees.

Improvements: We will add wheels and sandpaper to the claws, to add more friction between the arm and the beacon. Moreover, we will make the robot go forward a small distance when the arm is down and the claws are open, in order for the beacon to properly go in the claws at a flat angle (0 degrees).