**Team 1 Testing Document**

**Project:** Obstacle track-racer – A competitive robot design project

**Task:** Design and construct a machine that can autonomously navigate to a race track on an island and complete as many laps as possible within a 5 minute period, eventually returning to its starting point.

**Document Version Number**: 3.2

**Date:** 21/03/2021

**Edit History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Editor | Date | Operation |
| 1.0 | Junjian Chen | 2021/3/11 | The introduction, the table of contents and the brief test timeline. |
| 2.0 | Junjian Chen, Shichang Zhang | 2021/3/12 | The test plan explanation, the general purpose and test ideas of each test.  Complete the initial test plan. |
| 3.0 | Junjian Chen, Shichang Zhang | 2021/3/14 | Classify tests by their stages, revise the table of contents, test timeline.  Add Stage 1 Test Record |
| 3.1 | Dominic Chan | 2021/3/14 | Formatting and template of document, test format. |
| 3.2 | Junjian Chen, Shichang Zhang | 2021/3/21 | Complete Odometer, Ultrasonic Localization, Light Localization, Navigation parts of unit test. |

**Table Of Contents**

[**1.0 INTRODUCTION** 1](#_heading=h.gjdgxs)

[**2.0 TEST PLAN** 2](#_heading=h.30j0zll)

[**2.1 TEST OUTLINE** 2](#_heading=h.1fob9te)

[**2.2 TEST DESCRIPTION** 3](#_heading=h.3znysh7)

[**2.3 TEST TIMELINE** 6](#_heading=h.2et92p0)

[**3.0 TEST RECORDS** 6](#_heading=h.tyjcwt)

[**3.1 STAGE 1 TEST RECORD** 6](#_heading=h.3dy6vkm)

[**3.2 STAGE 2 TEST RECORD** 20](#_heading=h.1t3h5sf)

[**3.3 STAGE 3 TEST RECORD** 20](#_heading=h.4d34og8)

[**3.4 STAGE 4 TEST RECORD** 20](#_heading=h.2s8eyo1)

[**3.5 STAGE 5 TEST RECORD** 20](#_heading=h.17dp8vu)

# **1.0 INTRODUCTION**

The objective of the testing document is to build a system that can help to evaluate and optimize the performance of our project design.

The testing document consists of two parts-Test Plan (Part 2.0) and Test Record (Part 3)

The Test Plan involves a timeline of tests to be performed by weeks. Each test has a simple description of its objective, the methods used to test .

The Test Record will record every test performed and be classified by its component. A record has the following information:date, tester, hardware/software version number, purpose, objective, procedure, expected result output, test report, conclusion, action need to be done and distribution.

# **2.0 TEST PLAN**

In order to get a design that performs desirable functionality of client requirements, testing is an essential process of design. The testing plan of our project is divided into five main stages: Hardware Capability Test, Unit Test of each component, Integration test on combinational functionalities, Beta Demo Test and Acceptance Test on the completed design. These tests are from simple to complicated. They are carried step by step. The tests on basic components (ex. ultrasonic reading, moving forward for a certain distance) will tell us whether these basic components are reliable when working individually. If we know these basic components are reliable, when we deal with comprehensive functionalities such as navigation, we can pay more attention to how to combine these components to work out the complex function instead of going back to check these components’ functionality. Except for these three main test stages, we will also conduct tests when a hardware or software component is improved to know whether the updated part of hardware or software is consistent with other components and works better than the old one.

The test plan has a list of planned tests with some general information and a timeline that indicates when the planned test will be performed. The detailed information and implementation of listed tests is in the test record part of this testing document.

## **2.1 TEST OUTLINE**

1. Hardware Capability Test
   1. Stability
   2. Wheels
   3. Base Width
   4. Ultrasonic Sensor
   5. Light Sensor
2. Unit Test
   1. Odometer
   2. Ultrasonic Localization
   3. Light Localization
   4. Navigation
   5. Obstacle Avoidance
3. Integration Test
   1. Ultrasonic Localization and Light Localization Integration
   2. Navigation and Obstacle
4. Beta Demo Test
5. Acceptance Test

## **2.2 TEST DESCRIPTION**

**Stage 1: Hardware Capability Test**

In Stage 1, we will perform tests only in hardware components. We will test the capability of each hardware component individually.

* **Stability**

As in the previous lab, we found that when the robot ran straight for a large distance, the robot sometimes would deviate from the planned path due to the unstable hardware. We added ball casters to solve this problem. This time we also need to carry out the test on stability of the robot. We plan to make the robot move straight for a certain distance, such as 2 tile\_size, 2.5 tile\_size, 2.7 tile\_size. We will record the initial translation and final translation of the robot to confirm whether the robot deviates from the planned path.

* **Wheel**

As in the previous labs, we found that the default value of wheel radius was not accurate, in this time, we will make the robot move straight for a different distance and check if it can move for that distance correctly. For example, we will let the robot move straight for 1 tile\_size and check the translation of the robot to see whether it moves 1tile\_size. Then we let the robot move straight for 2 tile\_size, 2.5 tile\_size, 2.7 tile\_size and so on. If we fail in the test, we will adjust the value of wheel radius in the Resource class in our project and test again. Finally we will come up with an accurate wheel radius value.

* **Base Width**

The general idea for this test is to let the robot rotate for a certain angle. The accuracy of rotation depends on the accuracy of wheel radius and accuracy of base width. Given the wheel radius is accurate enough (i.e. we successfully pass the wheel test ), the rotation accuracy depends on the base width only. So we can record the start angle and final angle of the robot to confirm whether the base width is precise enough. If we fail the test, we will adjust the value of base width in the Resource class in our project. We will finally get an accurate base width.

* **Ultrasonic Sensor**

The test mainly tests the helper methods improving the ultrasonic sensor reading. Actually, it is also a test for whether the robot can detect an object on the path. From previous labs, we observed that the accuracy of feedback data of the ultrasonic sensor was undesirable. The ultrasonic sensor reading tends to fluctuate considerably when the robot orients to open space. We have developed some methods (average filter and median filter) to reduce the inaccuracy. We need to test whether the helper methods we developed can assist us to get a precise ultrasonic reading.

For the test, the general idea is to put an object in front of the robot and adjust the distance between the robot and the object. We will record the output of the median filter and average filter of the ultrasonic sensor readings. Then we will analyze whether the accuracy of these filters meets the requirements of our project and determine in which condition we should use which kind of filter. .

* **Light Sensor**

This test focuses on examining whether the robot can recognize black color of the line. We are using light localization to help reduce error during the navigation. The capability of detecting a black line is significant as it will affect the accuracy of localizing the robot to a waypoint. Similar to ultrasonic sensors’ readings, there are large fluctuation readings in light sensors’ readings and we also apply the same filter methods to make them more stable and trustable.

The procedure for this test is placing the robot on the center of a tile with different colors(Red,Green,Yellow) as a starting point. After that, the robot will move straight. If it detects a black color successfully, it will stop. We will record the readings of the light sensors and whether it is able to detect the lines.

**Stage 2: Unit Test**

In Stage 2, we will perform unit tests of each software component. We will examine the logic of the codes and test the performance of each software component individually.

* **Odometer**

The objective of the test is to evaluate the error of the odometer. Although the odometer class is already implemented as a part of starter code, it is not perfect and its error still exists. When the robot moves or turns, there is a deviation between odometer reading and real-world position. We want to find out how large the deviation is.

The testing method is making the robot move straight for different distances and turns by different angles. The corresponding odometer readings will be recorded and the deviation between it and its expected value will be calculated.

* **Ultrasonic Localization**

The test mainly tests whether Ultrasonic Localizer can localize the robot to our expected starting angle before light localization.

For the test, we will simulate the first step of the project, placing the robot with a random initial angle at the corner of the world (red region or green region). Then we will start the ultrasonic localizer with different initial angles and corners and record the final angle that the robot orients to to check whether the ultrasonic localization is accurate.

* **Light Localization**

The test mainly checks whether the Light Localizer can help the robot localize to the point we expect (the start point at beginning or the nearest point during the process of navigation).

For the test, we will test whether the light localizer can help to localize the nearest point while navigating. We will just let the robot move from a random start point to a random endpoint on the island. At the end of moving, we will let the robot use Light Localizer to localize to the endpoint. The final coordinate of the robot after localization will be recorded for the error analysis.

* **Navigation**

The test mainly checks whether the navigation system can function desirably in the final project. Our current Navigation System can localize the robot to waypoints with low errors. However, the large time consumption of it will lower the performance of the robot. The Navigation System software will be modified and different versions of it will be tested and evaluated.

For the test, similar to lab 5, we will pass some random way points to our design. We expect the robot to move to these way points in order. We expect to observe the error is within the same range of the current design. But we want the robot to complete the map for a much shorter time. Moreover, since the final project asks the robot to run the laps as many as possible, we will also conduct a test on whether the navigation system can assist the robot to run a map for multiple times and finally end with small errors.

* **Obstacle Avoidance**

This test focuses on evaluating the performance of obstacle avoidance functionality. The expected obstacle avoidance system enables the robot to avoid the obstacle on the path regardless of how the obstacle is placed, the angle the robot is facing.

In this test we will change the position of the obstacle relative to the robot to examine the robot to limits and find out the case that our obstacle avoidance system does not work. For example, we will place an obstacle on the navigation path and place another obstacle near the one on the path. In this condition, the robot is expected to not only avoid the obstacle on the path, but also avoid the obstacle closed to the former obstacle when avoiding the former obstacle.

**Stage 3:Integration test**

In Stage 3, we will integrate several software components to test the suitability of them.

* **Ultrasonic Localization and Light Localization Integration**

The test mainly evaluates the performance of the combination functionality of ultrasonic localization and light localization. We expect the robot can achieve the client needs in this part, firstly localizing to a certain angle and then localizing to the start point.

In the test, we will conduct similar steps as the unit test of ultrasonic localization. We will place the robot with a random initial angle at the corner of the world (green region or red region). Then we will run the localization methods and expect the robot to finally localize at the start point with the correct angle.

* **Navigation and Obstacle**

The test emphases on the performance of navigation when there are some obstacles on the planned path.

In this test, we plan to conduct two types of tests. The first one is similar to what we have done in lab 5. We give several way points on the island to the robot and place some obstacles on the navigation path. We expect the robot can travel to way points in order. During the process, we want that the robot can avoid the obstacles between waypoints. Finally we wish the robot to return to the start point with tolerable translation error and orientation error. The second one is testing whether the robot can keep the functionality of navigation and obstacle avoidance if it already has hit the obstacle. We will also pass some way points as the parameter to the robot and place obstacles on the path. We will run the program as normal. When the robot has travelled for a period of time, we will suddenly place an obstacle in front of the robot to let the robot hit the obstacle. Then we will remove the obstacle and record the following robot actions to determine whether it can keep the functionality of navigation and obstacle avoidance.

**Stage 4:Beta demo test**

**Stage 5:Acceptance test**

**Additional: Alternative Hardware Design Test**

Our current hardware design generally satisfies our needs for the final project. So we will regard the current hardware design as our project hardware design. We will conduct Hardware Capability Test on current hardware design firstly and then test the alternative hardware designs functionalities and compatibilities soon. During the development of software, if we encounter problems that must need the integration of hardware, we will switch the current hardware design to alternative hardware designs or integrate some component of current hardware design. At that time, we will conduct the hardware capability test again on the integrated hardware. This is depending on the software development, so there is no clear timeline for this kind of hardware component integration test.

## **2.3 TEST TIMELINE**

**Week of 3.14(3.14-3.21)**

* Stability
* Wheels
* Base Width
* Ultrasonic Sensor
* Light Sensor
* Odometer
* Ultrasonic Localization
* Light Localization

**Week of 3.22(3.22-3.28)**

* Navigation
* Obstacle Avoidance
* Ultrasonic Localization and Light Localization Integration
* Navigation and Obstacle

**Week of 3.29(3.29-4.4)**

* Beta Demo Test
* Acceptance Test

**Week of 4.5(4.5-4.11)(Final Week)**

* Complete System Test

# **3.0 TEST RECORDS**

## **3.1 STAGE 1 TEST RECORD**

**Hardware Capability Test (alternative 1) - Stability**

**Test:** Stability

**Date :** 2021/3/13

**Tester:** Shichang Zhang

**Author:** Shichang Zhang

**Hardware version:** 1.0

**Software version:** 1.0

**Test Purpose:**

Determine whether the hardware design is stable.

**Test Procedure:**

If the robot is not stable, the robot will deviate from the path when moving straight.

1. The robot is placed at (1,3). The starting translation of robot is (0.3048, 0.9144).
2. The robot is oriented to 90°. The starting angle is 90°.
3. Let the robot move straight for a certain distance.
4. Stop the robot and record the final translation and angle

**Test Data:**

|  |  |
| --- | --- |
| Trial# | Distance (m) |
| 1 | 0.30 |
| 2 | 0.30 |
| 3 | 0.30 |
| 4 | 0.85 |
| 5 | 0.85 |
| 6 | 0.97 |
| 7 | 1.22 |
| 8 | 1.54 |
| 9 | 1.78 |
| 10 | 2.02 |

**Expected Results:**

|  |  |  |
| --- | --- | --- |
| Trial# | Final Translation (m,m) | Final angle (deg) |
| 1 | (0.6048,0.9144) | 90.00 |
| 2 | (0.6048,0.9144) | 90.00 |
| 3 | (0.6048,0.9144) | 90.00 |
| 4 | (1.1548,0.9144) | 90.00 |
| 5 | (1.1548,0.9144) | 90.00 |
| 6 | (1.2748,0.9144) | 90.00 |
| 7 | (1.5248,0.9144) | 90.00 |
| 8 | (1.8448,0.9144) | 90.00 |
| 9 | (2.0848,0.9144) | 90.00 |
| 10 | (2.3248,0.9144) | 90.00 |

**Test Results:**

|  |  |  |
| --- | --- | --- |
| Trial# | Final Translation (m,m) | Final angle (deg) |
| 1 | (0.6047,0.9143) | 90.09 |
| 2 | (0.6047,0.9143) | 90.09 |
| 3 | (0.6047,0.9143) | 90.09 |
| 4 | (1.1550,0.9142) | 90.09 |
| 5 | (1.1550,0.9142) | 90.09 |
| 6 | (1.2749,0.9142) | 90.09 |
| 7 | (1.5249,0.9141) | 90.09 |
| 8 | (1.8450,0.9140) | 90.10 |
| 9 | (2.0851,0.9139) | 90.10 |
| 10 | (2.3252,0.9138) | 90.10 |

|  |  |  |
| --- | --- | --- |
| Trial# | Y-Translation Error (m,m) | Angle Error (deg) |
| 1 | 0.0001 | 0.09 |
| 2 | 0.0001 | 0.09 |
| 3 | 0.0001 | 0.09 |
| 4 | 0.0002 | 0.09 |
| 5 | 0.0002 | 0.09 |
| 6 | 0.0002 | 0.09 |
| 7 | 0.0003 | 0.09 |
| 8 | 0.0004 | 0.10 |
| 9 | 0.0005 | 0.10 |
| 10 | 0.0006 | 0.10 |

**Test Report:**  The test is performed 10 times for different input distances. In summary, we expect the robot’s y-translation and angle to be unchanged. In these 10 tests, we observe that with the increasing of distance, the robot will deviate more. The maximum change in y-translation of the robot is 0.0006m, and the maximum change in angle is 0.1deg. The errors are very small, so we can conclude that the current hardware design is stable.

**Conclusion:** Pass

**Action:** None

**Distribution:** software development, hardware development

**Hardware Capability Test (alternative 1) - Wheels**

**Test:** wheels

**Date :** 2021/3/13

**Tester:** Shichang Zhang

**Author:** Shichang Zhang

**Hardware version:** 1.0

**Software version:** 1.0

**Test Purpose:**

Determine whether the wheel radius of the robot is accurate.

**Test Procedure:**

If the wheel radius of the robot is larger, the robot will move a smaller distance than expected when moving straight. If the wheel radius of the robot is smaller, the robot will move a further distance than expected when moving straight.

1. The robot is placed at (1,3). The starting translation of robot is (0.3048, 0.9144).
2. The robot is oriented to 90°. The starting angle is 90°.
3. Let the robot move straight for a certain distance.
4. Stop the robot and record the final translation and angle

**Test Data:**

|  |  |
| --- | --- |
| Trial# | Distance (m) |
| 1 | 0.30 |
| 2 | 0.30 |
| 3 | 0.30 |
| 4 | 0.85 |
| 5 | 0.85 |
| 6 | 0.97 |
| 7 | 1.22 |
| 8 | 1.54 |
| 9 | 1.78 |
| 10 | 2.02 |

**Expected Results:**

|  |  |
| --- | --- |
| Trial# | Final Translation (m,m) |
| 1 | (0.6048,0.9144) |
| 2 | (0.6048,0.9144) |
| 3 | (0.6048,0.9144) |
| 4 | (1.1548,0.9144) |
| 5 | (1.1548,0.9144) |
| 6 | (1.2748,0.9144) |
| 7 | (1.5248,0.9144) |
| 8 | (1.8448,0.9144) |
| 9 | (2.0848,0.9144) |
| 10 | (2.3248,0.9144) |

**Test Results:**

|  |  |
| --- | --- |
| Trial# | Final Translation (m,m) |
| 1 | (0.6047,0.9143) |
| 2 | (0.6047,0.9143) |
| 3 | (0.6047,0.9143) |
| 4 | (1.1550,0.9142) |
| 5 | (1.1550,0.9142) |
| 6 | (1.2749,0.9142) |
| 7 | (1.5249,0.9141) |
| 8 | (1.8450,0.9140) |
| 9 | (2.0851,0.9139) |
| 10 | (2.3252,0.9138) |

|  |  |  |
| --- | --- | --- |
| Trial# | X Error (m,m) | Y Error (m,m) |
| 1 | 0.0001 | 0.01 |
| 2 | 0.0001 | 0.01 |
| 3 | 0.0001 | 0.01 |
| 4 | 0.0002 | 0.02 |
| 5 | 0.0002 | 0.02 |
| 6 | 0.0001 | 0.02 |
| 7 | 0.0001 | 0.03 |
| 8 | 0.0002 | 0.04 |
| 9 | 0.0003 | 0.05 |
| 10 | 0.0004 | 0.06 |

**Test Report:**

The test is performed 10 times for different input distances. In summary, we expect the x- translation of the robot to increase for the value of the input distance. We observed that the maximum error is 0.0004m. The error is very small. So our wheel radius is accurate.

**Conclusion:** Pass

**Action:** None

**Distribution:** software development, hardware development

**Hardware Capability Test (alternative 1)-Base width**

**Test:** base width

**Date :** 2021/3/13

**Tester:** Shichang Zhang

**Author:** Shichang Zhang

**Hardware version:** 1.0

**Software version:** 1.0

**Test Purpose:**

Determine whether the parameter base width of the robot is accurate.

**Test Procedure:**

We will let the robot turn by a certain degree. If the base width of the robot is larger, the robot will turn less degrees than expected. If the wheel radius of the robot is smaller, the robot will turn greater degrees than expected.

1. The robot is placed at (1,3). The starting translation of robot is (0.3048, 0.9144).
2. The robot is oriented to 90°. The starting angle is 90°.
3. Let the robot turn by a certain angle.
4. Stop the robot and record the final translation and angle

**Test Data:**

|  |  |
| --- | --- |
| Trial# | Angle (deg) |
| 1 | 20.0 |
| 2 | 45.0 |
| 3 | 75.0 |
| 4 | 90.0 |
| 5 | 90.0 |
| 6 | -90.0 |
| 7 | -90.0 |
| 8 | 102.0 |
| 9 | 110.5 |
| 10 | 135.0 |
| 11 | 180.0 |
| 12 | 360.0 |

**Expected Result:**

|  |  |
| --- | --- |
| Trial# | Final angle (deg) |
| 1 | 110.0 |
| 2 | 135.0 |
| 3 | 165.0 |
| 4 | 180.0 |
| 5 | 180.0 |
| 6 | 0.0 |
| 7 | 0.0 |
| 8 | 192.0 |
| 9 | 200.5 |
| 10 | 225.0 |
| 11 | 270.0 |
| 12 | 90.0 |

**Output:**

|  |  |  |
| --- | --- | --- |
| Trial# | Final Angle (deg) | Angle Error (deg) |
| 1 | 109.9 | 0.1 |
| 2 | 134.8 | 0.2 |
| 3 | 164.7 | 0.3 |
| 4 | 179.7 | 0.3 |
| 5 | 179.7 | 0.3 |
| 6 | 359.6 | 0.4 |
| 7 | 359.6 | 0.4 |
| 8 | 191.2 | 0.8 |
| 9 | 200.5 | 0.0 |
| 10 | 224.6 | 0.4 |
| 11 | 269.9 | 0.1 |
| 12 | 90.2 | 0.2 |

**Test Report:**

The test is performed 12 times for different input angles. In summary, we expect the robot to turn to the input angle. We observed that the maximum error is 0.8deg. The error is very small. So our base width is accurate.

**Conclusion:** Pass

**Action:** None

**Distribution:** software development, hardware development

**Hardware Capability Test (alternative 1)-ultrasonic sensor**

**Test:** ultrasonic sensor

**Date :** 2021/3/13

**Tester:** Shichang Zhang

**Author:** Shichang Zhang

**Hardware version:** 1.0

**Software version:** 1.0

**Test Purpose:**

Determine whether the feedback of the ultrasonic is acceptable after applying the filter on datas.

**Test Procedure:**

We will place an obstacle in front of the robot. We will see whether the filtered feedback meets our expectations.

1. The robot is placed at (1,3). The starting translation of robot is (0.3048, 0.9144).
2. The relative position of ultrasonic sensor is (0.0092, 0.0384), so the real translation of the ultrasonic sensor is (0.3140, 0.9528)
3. The length of the ultrasonic sensor is approximately 0.05m, so the detect point of the ultrasonic sensor is (0.3640, 0.9528)
4. The robot is oriented to 90°. The starting angle is 90°. The ultrasonic sensor angle is 90°.
5. Place an square obstacle with side length 25.9cm in front of the robot with a certain distance.
6. Run the ultrasonic sensor at a sample rate of 25Hz and use the filter methods to deal with samples. .
7. Print the three filtered feedback on the console.
8. Stop the robot and record the printed data.

**Test Data:**

|  |  |
| --- | --- |
| Trial# | Distance (m) |
| 1 | 0.20 |
| 2 | 0.20 |
| 3 | 0.30 |
| 4 | 0.30 |
| 5 | 0.40 |
| 6 | 0.40 |
| 7 | 0.55 |
| 8 | 0.55 |
| 9 | 0.80 |
| 10 | 0.90 |
| 11 | 1.10 |
| 12 | 1.10 |
| 13 | 1.20 |
| 14 | 1.40 |
| 15 | 1.50 |

**Expected Result:**

|  |  |
| --- | --- |
| Trial# | Distance (m) |
| 1 | 0.20 |
| 2 | 0.20 |
| 3 | 0.30 |
| 4 | 0.30 |
| 5 | 0.40 |
| 6 | 0.40 |
| 7 | 0.55 |
| 8 | 0.55 |
| 9 | 0.80 |
| 10 | 0.90 |
| 11 | 1.10 |
| 12 | 1.10 |
| 13 | 1.20 |
| 14 | 1.40 |
| 15 | 1.50 |

**Test Results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trial# | Median Filter (m) | Maximum error (m) | Minimum error (m) | Mean of three filtered sample (m) |
| 1 | 0.17, 0.17, 0.20 | 0.03 | 0.00 | 0.18 |
| 2 | 0.17, 0.17, 0.20 | 0.03 | 0.00 | 0.18 |
| 3 | 0.25, 0.25, 0.29 | 0.05 | 0.01 | 0.26 |
| 4 | 0.25, 0.25, 0.29 | 0.05 | 0.01 | 0.26 |
| 5 | 0.40, 0.40, 0.40 | 0.00 | 0.00 | 0.40 |
| 6 | 0.40, 0.40, 0.40 | 0.00 | 0.00 | 0.40 |
| 7 | 0.54, 0.54, 0.54 | 0.01 | 0.01 | 0.54 |
| 8 | 0.54, 0.54, 0.54 | 0.01 | 0.01 | 0.54 |
| 9 | 0.74, 0.74, 0.74 | 0.06 | 0.06 | 0.74 |
| 10 | 0.81, 0.81, 0.81 | 0.09 | 0.09 | 0.81 |
| 11 | 0.95, 0.95, 0.95 | 0.15 | 0.15 | 0.95 |
| 12 | 0.95, 0.95, 0.95 | 0.15 | 0.15 | 0.95 |
| 13 | 0.78, 0.78, 0.97 | 0.42 | 0.23 | 0.84 |
| 14 | 0.83, 0.83, 1.04 | 0.57 | 0.36 | 0.90 |
| 15 | 0.85, 0.85, 1.06 | 0.65 | 0.44 | 0.92 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trial# | Average Filter (m) | Maximum error (m) | Minimum error (m) | Mean of three filtered sample (m) |
| 1 | 0.17, 0.22, 0.22 | 0.03 | 0.02 | 0.20 |
| 2 | 0.17, 0.22, 0.22 | 0.03 | 0.02 | 0.20 |
| 3 | 0.25, 0.32, 0.32 | 0.05 | 0.02 | 0.27 |
| 4 | 0.25, 0.32, 0.32 | 0.05 | 0.02 | 0.27 |
| 5 | 0.40, 0.40, 0.40 | 0.00 | 0.00 | 0.40 |
| 6 | 0.40, 0.40, 0.40 | 0.00 | 0.00 | 0.40 |
| 7 | 0.54, 0.44, 0.58 | 0.11 | 0.01 | 0.52 |
| 8 | 0.54, 0.44, 0.58 | 0.11 | 0.01 | 0.52 |
| 9 | 0.74, 0.59, 0.80 | 0.21 | 0.00 | 0.71 |
| 10 | 0.81, 0.64, 0.88 | 0.26 | 0.02 | 0.78 |
| 11 | 0.95, 0.74, 1.04 | 0.36 | 0.06 | 0.91 |
| 12 | 0.95, 0.74, 1.04 | 0.36 | 0.06 | 0.91 |
| 13 | 1.02, 0.78, 1.11 | 0.42 | 0.09 | 0.97 |
| 14 | 0.83, 1.19, 1.19 | 0.57 | 0.21 | 1.07 |
| 15 | 0.94, 1.12, 1.12 | 0.56 | 0.38 | 1.06 |

**Test Report:**

The test is performed 15 times for different input distances. In summary, we expect that the ultrasonic sensor returns the value closed to the input distance . We observed that for small values of input distances, the median filter will have stable but not so accurate outcome and the average filter will have relatively fluctuating readings. But the minimal error of the average filter is very small for short distances, meaning that one of outputs of the average filter is very close to our expectations. And we observe that the mean of the outputs of the average filter is closer to the expectation than that of the median filter for the short distance. But for the relatively large distance, both the median filter and average filter fail to work out accurate results. We need to avoid this case in the project since it will cause large errors.

**Conclusion:** Fail

**Action:** Need an algorithm to avoid long distances detection when using the ultrasonic sensor.

**Distribution:** software development, hardware development

**Hardware Capability Test (alternative 1) - light sensor**

**Test:** light sensor

**Date :** 2021/3/15

**Tester:** Shichang Zhang

**Author:** Shichang Zhang

**Hardware version:** 1.0

**Software version:** 1.0

**Test Purpose:**

Determine whether the filtered feedback of the light is acceptable after applying the medium filter on datas.

**Test Procedure:**

We will place the robot stationary at the center of the colored tile and let the robot detect the color of the tile. We want the robot to correctly detect the tile color. After a short period, we will let the robot move forward. We want the robot to stop when light sensors detect the black grid line.

1. The robot is placed at the center of a tile. The exact position will be determined by the input coordinate.
2. Depending on the input coordinate, tile is expected to be red, or green, or yellow.
3. The robot is oriented to 90°. The starting angle is 90°.
4. Run the light sensors at a sample rate of 25Hz and use the filter methods to deal with samples. .
5. Print the three filtered feedback on the console.
6. Let the robot move forward.
7. Let both light sensors successfully detect the black grid line.
8. Stop the robot.
9. Print the three filtered feedback on the console.
10. Record the printed data.

**Test Data:**

|  |  |  |
| --- | --- | --- |
| Trial# | Coordinate (ft, ft) | Color |
| 1 | (0.5, 0.5) | yellow |
| 2 | (0.5, 0.5) | yellow |
| 3 | (0.5, 0.5) | yellow |
| 4 | (0.5, 8.5) | red |
| 5 | (0.5, 8.5) | red |
| 6 | (0.5, 8.5) | red |
| 7 | (13.5, 8.5) | green |
| 8 | (13.5, 8.5) | green |
| 9 | (13.5, 8.5) | green |

**Expected Result:**

|  |  |  |
| --- | --- | --- |
| Trial# | Start Point Tile Color Reading | Black Line reading |
| 1 | 240 | 70 |
| 2 | 240 | 70 |
| 3 | 240 | 70 |
| 4 | 255 | 70 |
| 5 | 255 | 70 |
| 6 | 255 | 70 |
| 7 | 200 | 70 |
| 8 | 200 | 70 |
| 9 | 200 | 70 |

**Test Results:**

|  |  |  |
| --- | --- | --- |
| Trial# | Left Sensor Start Point Tile Color Reading | Left Sensor Black Line reading |
| 1 | 243, 243, 244 | 72, 71, 72 |
| 2 | 242, 242, 243 | 70, 70, 74 |
| 3 | 243, 243, 244 | 72, 72, 76 |
| 4 | 251, 251, 252 | 71, 71, 71 |
| 5 | 248, 251, 252 | 71, 71, 76 |
| 6 | 248, 252, 253 | 74, 73, 72 |
| 7 | 196, 196, 197 | 70, 67, 70 |
| 8 | 193, 193, 197 | 74, 73, 74 |
| 9 | 193, 198, 198 | 68, 70, 70 |

|  |  |  |
| --- | --- | --- |
| Trial# | Right Sensor Start Point Tile Color Reading | Right Sensor Black Line reading |
| 1 | 240, 243, 244 | 72, 72, 76 |
| 2 | 243, 243, 244 | 71, 71, 71 |
| 3 | 240, 243, 244 | 71, 71, 71 |
| 4 | 248, 251, 252 | 71, 70, 70 |
| 5 | 248, 251, 252 | 70, 67, 70 |
| 6 | 249, 252, 252 | 73, 73, 73 |
| 7 | 193, 196, 197 | 77, 74, 77 |
| 8 | 194, 195, 196 | 70, 67, 70 |
| 9 | 194, 194, 197 | 77, 73, 71 |

**Test Report:**

The test is performed 9 times for different input distances. We expect the robot to detect the tile color and black grid line successfully. From the tested output, we can see that for color yellow, red, green we get very little error and for the black line we get relatively large error but are tolerable. So we can conclude that the medium filter helps the light sensor return accurate data so that the robot can clearly know the color of the area detected by the light sensors.

**Conclusion:** Pass

**Action:** None

**Distribution:** software development, hardware development

**Hardware Capability Test - Going up the slope of the bridge.**

**Date :** 21/03/2021

**Tester:** Dominic Chan,

**Author:** Dominic Chan

**Hardware version:** 1.1

**Software version:**

**Purpose:** To determine whether the robot can navigate and cross the bridge connecting the starting island to the main island successfully.

**Procedure:**

1. The robot is placed at the center of the starting island.
2. The robot navigates forward and onto the inter-connecting bridge.
3. The robot crosses the bridge successfully in a straight line.

**Expected Results:**

The robot is able to cross the bridge in a straight line for all trials.

**Test Results:**

|  |  |  |
| --- | --- | --- |
| Trial | Pass/Fail | Comments |
| 1 | Fail | The robot gets stuck when trying to go up the bridge. |
| 2 | Fail | Same as trial 1. |
| 3 | Fail | Same as trial 1. |
| 4 | Fail | Same as trial 1. |
| 5 | Fail | Same as trial 1. |

**Test Report:**

The test was performed a total of 5 times, and for each time, the robot gets stuck when trying to get onto the slope of the bridge. The issue appears to be because the front-end of the robot is too low so that it gets caught with the slope.

**Conclusion:**

Test failed as the robot is unable to make it across the bridge.

**Action:**

Should re-evaluate the hardware/mechanical design of the robot to ensure this doesn’t happen.

**Distribution:**

Hardware Design Team

**Hardware Capability Test - Going down the slope of the bridge**

**Date :** 21/03/2021

**Tester:** Dominic Chan.

**Author:** Dominic Chan

**Hardware version:** 1.2

**Software version:**

**Purpose:** To determine whether the robot can navigate and cross the bridge connecting the starting island to the main island successfully.

**Procedure:**

1. The robot is placed at the center of the starting island.
2. The robot navigates forward and onto the inter-connecting bridge.
3. The robot crosses the bridge successfully in a straight line.

**Expected Results:**

The robot is able to cross the bridge in a straight line for all trials.

**Test Results:**

|  |  |  |
| --- | --- | --- |
| Trial | Pass/Fail | Comments |
| 1 | Fail | The robot plunges forward and rolls down the slope. |
| 2 | Fail | Same as trial 1. |
| 3 | Fail | Same as trial 1. |
| 4 | Fail | Same as trial 1. |
| 5 | Fail | Same as trial 1. |

**Test Report:**

The test was performed a total of 5 times, and for each time, when the robot goes down the bridge, it plunges forward and rolls down the slope.

**Conclusion:**

Test failed as the robot is unable to make it across the bridge.

**Action:**

The issue appears to be because the robot’s center of gravity is faced forward so when the robot navigates down, it plunges forward and rolls down the hill. Should re-evaluate the hardware/mechanical design of the robot to ensure this doesn’t happen.

**Distribution:**

Hardware Design Team

## **3.2 STAGE 2 TEST RECORD**

**Unit Test (alternative 1) - Ultrasonic Localization**

**Test:** ultrasonic Localization

**Date :** 2021/3/22

**Tester:** Shichang Zhang

**Author:** Shichang Zhang

**Hardware version:** 1.2

**Software version:** 1.3

**Test Purpose:**

Determine whether the ultrasonic localizer can help the red team robot to turn to the desired angle with limited errors.

**Test Procedure:**

We will place the robot at the upper left corner on the 45° line with the random initial angle. We expect the robot can finally turn to 180° according to the ultrasonic sensor feedback.

1. The robot is placed at coordinate (0.5, 8.5), that is (0.1524, 2.5908) in meters, which is on the 45° line of the upper left corner of the world.
2. Depending on the input angle, the robot will be set to orient to the angle.
3. Start the odometer. Initializing the odometer with value (0.1524, 2.5908, input θ).
4. Run the ultrasonic localizer with ultrasonic sensor running at a sample rate of 25Hz and use the filter methods to deal with samples.
5. When the ultrasonic localizer stops running, stop the program .
6. Print the final angle value indicated by the webot.
7. Print the final angle value indicated by the odometer.

**Test Data:**

|  |  |
| --- | --- |
| Trial# | Initial Angle |
| 1 | 0.0 |
| 2 | 20.0 |
| 3 | 55.0 |
| 4 | 82.0 |
| 5 | 120.0 |
| 6 | 165.0 |
| 7 | 200.0 |
| 8 | 280.0 |
| 9 | 330.0 |

**Expected Result:**

|  |  |  |
| --- | --- | --- |
| Trial# | Final Angle (Webot) | Final Angle (Odometer) |
| 1 | 180.0 | 180.0 |
| 2 | 180.0 | 180.0 |
| 3 | 180.0 | 180.0 |
| 4 | 180.0 | 180.0 |
| 5 | 180.0 | 180.0 |
| 6 | 180.0 | 180.0 |
| 7 | 180.0 | 180.0 |
| 8 | 180.0 | 180.0 |
| 9 | 180.0 | 180.0 |

**Test Results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trial# | Final Angle (Webot)/deg | Final Angle (Odometer)/deg | Error (Webot)/deg | Error (Odometer)/deg |
| 1 | 178.1 | 180.0 | 1.9 | 1.9 |
| 2 | 177.3 | 180.0 | 2.7 | 2.7 |
| 3 | 178.5 | 180.0 | 1.5 | 1.5 |
| 4 | 178.2 | 180.0 | 1.8 | 1.8 |
| 5 | 177.7 | 180.0 | 2.3 | 2.3 |
| 6 | 178.4 | 180.0 | 1.6 | 1.6 |
| 7 | 179.2 | 180.0 | 0.8 | 0.8 |
| 8 | 178.6 | 180.0 | 1.4 | 1.4 |
| 9 | 177.9 | 180.0 | 2.1 | 2.1 |

**Test Report:**

The test is performed 9 times for different input initial angles. We expect the robot to turn to 180° and the error is expected to be within 1 deg. From the tested output, we can see that the ultrasonic localizer generates maximum error 2.7 deg and minimum error 0.8 deg. We can also observe that the odometer generates some inaccuracy here. Overall, some errors are larger than our expectation, but they are within 5 deg requirements. The upcoming light localization can reduce the inaccuracy dramatically.

**Conclusion:** Pass

**Action:** None

**Distribution:** software development

**Unit Test (alternative 1) - Ultrasonic Localization**

**Test:** ultrasonic Localization

**Date :** 2021/3/22

**Tester:** Shichang Zhang

**Author:** Shichang Zhang

**Hardware version:** 1.2

**Software version:** 1.3

**Test Purpose:**

Determine whether the ultrasonic localizer can help the green team robot to turn to the desired angle with limited errors.

**Test Procedure:**

We will place the robot at the upper right corner on the 45° line with the random initial angle. We expect the robot can finally turn to the desired angle according to the ultrasonic sensor feedback.

1. The robot is placed at coordinate (14.5, 8.5), that is (4.4604, 2.5908) in meters, which is on the 45° line of the upper right corner of the world.
2. Depending on the input angle, the robot will be set to orient to the angle.
3. Start the odometer. Initializing the odometer with value (4.4604, 2.5908, input θ).
4. Run the ultrasonic localizer with ultrasonic sensor running at a sample rate of 25Hz and use the filter methods to deal with samples.
5. When the ultrasonic localizer stops running, stop the program .
6. Print the final angle value indicated by the webot.
7. Print the final angle value indicated by the odometer.

**Test Data:**

|  |  |
| --- | --- |
| Trial# | Initial Angle |
| 1 | 0.0 |
| 2 | 20.0 |
| 3 | 55.0 |
| 4 | 82.0 |
| 5 | 120.0 |
| 6 | 165.0 |
| 7 | 200.0 |
| 8 | 280.0 |
| 9 | 330.0 |

**Expected Result:**

|  |  |  |
| --- | --- | --- |
| Trial# | Final Angle (Webot) | Final Angle (Odometer) |
| 1 | 270 | 270 |
| 2 | 270 | 270 |
| 3 | 270 | 270 |
| 4 | 270 | 270 |
| 5 | 270 | 270 |
| 6 | 270 | 270 |
| 7 | 270 | 270 |
| 8 | 270 | 270 |
| 9 | 270 | 270 |

**Test Results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trial# | Final Angle (Webot)/deg | Final Angle (Odometer)/deg | Error (Webot)/deg | Error (Odometer)/deg |
| 1 | 275.2 | 270 | 5.2 | 5.2 |
| 2 | 275.1 | 270 | 5.1 | 5.1 |
| 3 | 275.3 | 270 | 5.3 | 5.3 |
| 4 | 274.5 | 270 | 4.5 | 4.5 |
| 5 | 276.7 | 270 | 6.7 | 6.7 |
| 6 | 275.3 | 270 | 5.3 | 5.3 |
| 7 | 277.4 | 270 | 7.4 | 7.4 |
| 8 | 276.0 | 270 | 6.0 | 6.0 |
| 9 | 276.2 | 270 | 6.2 | 6.2 |

**Test Report:**

The test is performed 9 times for different input initial angles. We expect the robot to turn to 270° and the error is expected to be within 1 deg. From the tested output, we can see that the ultrasonic localizer generates maximum error 7.4 deg and minimum error 4.5 deg. We can also observe that the odometer generates large inaccuracy here. Overall, most errors are larger than our expectation, and they exceed 5 deg requirements.

**Conclusion:** Fail

**Action:** Integrate ultrasonic localizer that fits green team

**Distribution:** software development

**Unit Test-Odometer**

**Test:** Odometer

**Date:** 2021/3/22

**Tester:** Junjian Chen

**Author:** Junjian Chen

**Hardware version:** 1.2

**Software version:** 1.3

**Purpose:** Evaluate the deviation of readings of odometer the actual angles and positions of the robot

**Procedure:** 1.Place the robot where the center of the wheels are located at (1,8) and face 0 degree

1.Set the odometer to (0.3048,2.4384,0)

2. Start the odometer

3. Turn the robot by different sets of angle

4. Record the actual angles in webot and the theta of the odometer

5. Calculate the error of angle by the following formula:

error=Final Angle(Odometer)-Final Angle(Webot)

6. After completing different sets of angle test, place the robot at (1,1) and face 90 degree

7. Set the odometer to (0.3048,0.3048,90)

8. Remove the obstacle and let the robot move straight for different sets of distances

9. Record the actual position of the robot and the coordinate(in feets) of the odometer

10. Calculate the error of distance by applying the formula of Euclidean error distance

**Expected Results:**

Test of angle error of odometer:

|  |  |  |  |
| --- | --- | --- | --- |
| Trial# | Final Angle/degree | Odometer Reading/degree | Error  /degree |
| 1 | 30 | 30 | 0 |
| 2 | 60 | 60 | 0 |
| 3 | 90 | 90 | 0 |
| 4 | 120 | 120 | 0 |
| 5 | 150 | 150 | 0 |
| 6 | 180 | 180 | 0 |
| 7 | 270 | 270 | 0 |
| 8 | 360 | 360 | 0 |

Test of distance error of odometer:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trial# | Distance Moved  /feet | Actual Coordinate  /(x,y) | Odometer Reading  /(x,y) | Error distance  /feet |
| 1 | 1 | (2,1) | (2,1) | 0 |
| 2 | 2 | (3,1) | (3,1) | 0 |
| 3 | 3 | (4,1) | (4,1) | 0 |
| 4 | 4 | (5,1) | (5,1) | 0 |
| 5 | 5 | (6,1) | (6,1) | 0 |
| 6 | 6 | (7,1) | (7,1) | 0 |
| 8 | 7 | (8,1) | (8,1) | 0 |
| 7 | 8 | (9,1) | (9,1) | 0 |

**Test Results:**

Test of angle error of odometer:

|  |  |  |  |
| --- | --- | --- | --- |
| Trial# | Final Angle(Webot)  /degree | Final Angle(Odometer)  /degree | Error  /degree |
| 1 | 29.51 | 29.96 | 0.45 |
| 2 | 59.59 | 59.92 | 0.33 |
| 3 | 89.38 | 89.88 | 0.50 |
| 4 | 118.60 | 119.84 | 1.24 |
| 5 | 148.40 | 149.80 | 1.40 |
| 6 | 178.42 | 179.63 | 1.21 |
| 7 | 267.87 | 269.92 | 2.05 |
| 8 | 356.95 | 359.80 | 2.85 |

Test of distance error of odometer:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trial# | Distance Moved  /feet | Actual Coordinate  /(x,y) | Odometer Reading  /(x,y) | Error distance  /feet |
| 1 | 1 | (2.00,1.00) | (2.00,1.00) | 0 |
| 2 | 2 | (3.00,1.00) | (3.00,1.00) | 0 |
| 3 | 3 | (4.00,1.00) | (4.00,1.00) | 0 |
| 4 | 4 | (5.00,1.00) | (5.00,1.00) | 0 |
| 5 | 5 | (6.00,1.00) | (6.00,1.00) | 0 |
| 6 | 6 | (7.00,1.00) | (7.00,1.00) | 0 |
| 7 | 7 | (8.00,1.00) | (8.00,1.00) | 0 |
| 8 | 8 | (9.00,1.00) | (9.00,1.00) | 0 |

**Test Report:**

The unit test of odometer consists of two parts: test of angle and distance. They are performed with 8 sets of different angles to rotate and distance to move respectively. As we can see the error of the distance test is very tiny and can be ignored. When it comes to the angles, the difference between the actual angle and the odometer reading increases as the turning angle increases, with a maximum of 2.85 degrees when turning by 360 degree.

**Conclusion:** The error of theta of the odometer increases with turning angles and the error of x and y is extremely low.

**Action: None**

**Distribution:** software development

**Unit Test - Light Localization**

**Date:** 2021/3/21

**Tester:** Junjian Chen

**Author:** Junjian Chen

**Hardware version:** 1.2

**Software version:** 1.3

**Purpose:** Evaluate the error of when localizing the robot to a nearest point while navigation

**Procedure:** 1. Place the robot to (4,3)

2. Use the method in Navigation Class directTravelTo() (light localization is used in this method) to make the robot travel to 8 different points.

3. Record the actual position of the robot after localization

4. Calculate the error by Euclidean error distance

**Expected Results:**

|  |  |  |
| --- | --- | --- |
| Trial# | Starting point  /(x,y) | End point  /(x,y) |
| 1 | (4,3) | (4,5) |
| 2 | (4,3) | (4,1) |
| 3 | (4,3) | (2,3) |
| 4 | (4,3) | (6,3) |
| 5 | (4,3) | (2,5) |
| 6 | (4,3) | (5,5) |
| 7 | (4,3) | (6,1) |
| 8 | (4,3) | (1,1) |

**Test Results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trial# | Starting point  /(x,y) | Expected End point  /(x,y) | Actual End point  /(x,y) | Error distance  /feet |
| 1 | (4,3) | (4,5) | (4.00,4.99) | 0.01 |
| 2 | (4,3) | (4,1) | (3.98,1.12) | 0.03 |
| 3 | (4,3) | (2,3) | (2.01,3.00) | 0.01 |
| 4 | (4,3) | (6,3) | (5.98,3.01) | 0.02 |
| 5 | (4,3) | (2,5) | (1.99,4.99) | 0.01 |
| 6 | (4,3) | (5,5) | (4.99,4.99) | 0.01 |
| 7 | (4,3) | (6,1) | (6.01,1.01) | 0.01 |
| 8 | (4,3) | (1,1) | (0.99,1.01) | 0.01 |

**Test Report:**

The error of the light localizer is very low, most of the errors are between 0.01 and 0.02 feet, with only one trial at 0.03. So the performance of the light localization is good.

**Conclusion:** Pass

**Action:** None

**Distribution:** software development

**Unit Test (alternative 1) - Navigation**

**Test:** navigation

**Date :** 2021/3/22

**Tester:** Shichang Zhang

**Author:** Shichang Zhang

**Hardware version:** 1.2

**Software version:** 1.3

**Test Purpose:**

Determine whether the navigation system can help the robot to go through the waypoints in order and finally end at (1,4) with a small error. .

**Test Procedure:**

We will place the robot at (0.5, 4.5) as the start point. And we will pass a series of waypoints to the robot. We expect the robot can go through these waypoints in order and finally stop at (1,4) with small error. We also expect the time cost of the navigation process to be low. .

1. The robot is placed at coordinate (0.5, 4.5), that is (0.1524, 1.3716) in meters.
2. The robot is set to orient to 180 degrees.
3. Pass the input waypoints to the robot.
4. Set the forward speed to 500, rotate speed to 200, localize speed to 200.
5. Start the odometer. Initializing the odometer with value (0.1524, 1.3716,180).
6. Run the navigation system, start the timer.
7. Record whether the robot goes through all points in order.
8. When the robot ends at (1,4), stop the program, end the timer.
9. Print the final translation and angle indicated by Webot.
10. Record the time taken for the navigation.

**Test Data:**

|  |  |
| --- | --- |
| Trial# | way points (ft,ft) |
| 1 | (1,4), (4,2),(6,5),(4,5),(11,1),(14,4),(8,4),(1,4) |
| 2 | (1,4), (5,2),(9,4),(7,5),(9,1),(14,4),(11,5),(14,2),(4,5),(1,4) |
| 3 | (1,4), (4,5),(9,1),(6,5),(14,2),(14,4),(8,4),(4,1),(1,4) |
| 4 | (1,4),(3,1),(11,1),(11,5),(8,4),(10,2),(14,2),(7,5),(5,2),(1,4) |
| 5 | (1,4),(6,2),(8,5),(9,1),(11,5),(13,1),(14,4),(10,1),(6,5),(3,2),(1,4) |

**Expected Result:**

|  |  |  |  |
| --- | --- | --- | --- |
| Trial# | Translation (m,m) | Angle (deg) | Timer (HH:MM:SS) |
| 1 | (0.3048,1.2192) | 270 | 00:05:00>time |
| 2 | (0.3048,1.2192) | 270 | 00:05:00>time |
| 3 | (0.3048,1.2192) | 270 | 00:05:00>time |
| 4 | (0.3048,1.2192) | 270 | 00:05:00>time |
| 5 | (0.3048,1.2192) | 270 | 00:05:00>time |

**Test Results:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Trial# | Translation (m,m) | Angle (deg) | Timer (HH:MM:SS) | Translation error(m,m) | Angle error (deg) |
| 1 | (0.3059,1.2193) | 272.1 | 00:03:24 | (0.0011,0.0001) | 2.1 |
| 2 | (0.3070,1.2198) | 270.3 | 00:03:50 | (0.0022,0.0006) | 0.3 |
| 3 | (0.3035,1.2165) | 270.8 | 00:03:11 | (-0.0013,-0.0027) | 0.8 |
| 4 | (0.3089,1.1288) | 270.5 | 00:03:34 | (0.0041,-0.0004) | 0.5 |
| 5 | (0.3078,1.1289) | 270.9 | 00:04:08 | (0.0030,-0.0003) | 0.9 |

**Test Report:**

The test is performed 5 times for different input waypoints. We expect the robot to travel to each waypoint in order and end at (1,4) with limited translation error and angle error. From the tested output, we can see that the navigation system generates little translation and angle error. We also observe that the time taken for each trial is around 3-4 minutes, which means the robot can accurately travel through 9-10 waypoints quickly given there is no obstacle on the path. Overall, the navigation system gives really little error and it also runs fast. .

**Conclusion:** Pass

**Action:** None

**Distribution:** software development

## **3.3 STAGE 3 TEST RECORD**

## **3.4 STAGE 4 TEST RECORD**

## **3.5 STAGE 5 TEST RECORD**

TEST TEMPLATE (TEMPORARY)

**Date :**

**Tester:**

**Author:**

**Hardware version:**

**Software version:**

**Purpose:**

**Objectives:**

**Procedure:**

**Expected Results:**

**Test Results:**

**Test Report:**

**Conclusion:**

**Action:**

**Distribution:**