Software Test Plan
Project 3: Multiple-Agents
AI Robotics - CSCE635

Anh Nguyen Tuan, Narayani Vedam and QiLi Huang

December 9, 2013

1 Introduction

This document is a high-level overview defining our testing strategy for the crowd blocking application. Its objective is to guard and block the exit of a room or stadium. This document will address the different standards that will apply to the unit and integration testing of the application.

In this second version, we will update the unit testing and integration testing for multiple agents system.

2 Test Objective

The objective of our test plan is to find and report as many bugs as possible to improve the integrity of our program. Although exhaustive testing is not possible, we will exercise a broad range of tests to achieve our goal.

We will be testing all components of the behavioral system: WALL-AVOIDANCE behavior, WATCHING behavior, APPROACHING behavior and THREATENING behavior. Finally, we will test the whole system with the GUI interface that user can interact with to see if the robot behaves correctly according to the design.

In this second version of the project, we introduce one more behavior: ROBOT-AVOIDANCE. We have multiple UAVs flying in the same plane. So, we do not want them to collide into each other. The test plan will cover this addition module.

In each behavior, we will test the correctness of PERCEPTUAL schema, MOTOR schema, SENSOR and the motor controller separately, and then test the whole behavior.

Concretely, we have 3 UAVs and our task is to guard 20ft wide exit door. The expected behavior of the overall system is that each UAV will monitor the crowds in its region. Each UAV will might have different targets and will act according to the position and movement of those targets.

3 How to Test

Because this system requires visual display to see behaviors. So most of the test will be done by controlling and watching user interface, including: Schema testing & integration testing. User's controls are defined as follows:

- Add more robot using keyboard button 'J'. This is used to test robot avoidance and wall avoidance.

- Destroy robot using keyboard button 'K'. This is used to test robot avoidance and wall avoidance

- Change mode: using keyboard button '1', '2', '3' and 'Space' to stop. This is used to test schemas and behaviors.

Sensor Testing will use automated test script in the code.

4 Testing Plan

The following outlines the types of testing that will be done for unit, integration test.

4.1 Unit Test

4.1.1 Sensor Testing

1. Hokuyo Testing:

   Hokuyo module is a simulated sensor that provides the position of the robot in the world. The robot uses this information to avoid colliding with the wall, ceiling and floor.

   This test is to make sure that the Hokuyo module will provide the correct position of the robot. We will place the robot in different position and then check if the Hokuyo module returns the correct value.

   For the second version of Project, the Hokuyo also provides the position of all UAVs so that the information could be used to avoid collisions with other.

   The unit test of the Hokuyo also is to test if the Hokuyo returns the correct position of other robots.

2. Camera Testing:

   This simulated camera mounted on the robot to give it the information about the crowd. This test is to make sure that the Camera module will return all crowd position, i.e. all people positions.

For example, if the crowd consists of 3 people, the camera will return the position of all 3 people.
The unit test will assure that the camera give the correct picture of people in the crowd.

The script for sensor testing can be found in file "Testing.js". To use it, user need to add this script to the airrobotJ 2 object. The script will print out the result in the console of Unity.

4.1.2 Perceptual Schema Testing

The perceptual schema takes input from the sensors and then extract the closest target for each robot. Each robot might have different target if there are many target coming at the same speed and distance.

4.1.3 Motor Schema Testing

Our system consists of 5 motor schemas: Wall-Avoidance, Robot-Avoidance, Watching, Approaching and Threatening.

1. Wall-Avoidance Testing, This schema gives the robot the ability to avoid collision with other obstacles: wall, ceiling, floor. Test cases include placing the robot in many places in the room to see if the robot can react to the obstacles

2. Robot-Avoidance, this schema gives the robot the ability to avoid collision with other robots.

   Test cases include placing the robot in many places in the room to see if the robot can react to the robots.

3. Watching the robot run, this schema in the case that there is no crowd or the crowd is far away from the robot. The expected result of this schema is the robot keep flying at a predefined altitude.

4. Approaching, the robot runs this schema in the scenario that there is crowd and the distance from the robot to the crowd is not too closed.

   The expected behavior of the robot is that it will low down its altitude and move close to the center of the person in the X-axis.

5. Threatening, the robot runs this schema in the scenario that the crowd is really closed to the robot. The expected behavior of the robot is that it will turn around move randomly in 2D to scare the crowd.

4.2 Integration Test

After all unit tests are passed, we will perform integration testing, which incrementally combine small part of the system in to a complete behavior and test that behavior.

We will test each behavior and then all behaviors to make sure it follows the design.

Due to the nature of the system, the integration test will be checked by looking at the user interface. This test also uses user's control as mention above.

Integration tests include the following sub-tests:

4.2.1 Watching Behavior Testing

This test case is to test the behavior of the robot when there is no crowd or the crowd is farther than a certain bound.

Test cases will include the following scenarios:

- There is no crowd
- There is crowd but distance from crowd to the robot is larger than a certain value.

4.2.2 Approaching Behavior Testing

This test case is to test the behavior of the robot when there is the crowd and its distance to the robot is in a certain range.

Testing for this behavior includes placing the crowd in many places and move in a different speed to check the reaction of the robot.

Note that there are 3 UAVs, so each UAV will only monitor the crowd in its region.

4.2.3 Threatening Behavior Testing

This test case is to test the behavior of the robot when there is the crowd and its distance to the robot is less than a certain value, which means it really closed to the robot.

Testing for this behavior includes placing the crowd in many places and move in a different speed to check the reaction of the robot.

Note that there are 3 UAVs, so each UAV will only threaten the crowd in its region.

5 Performance Test

We test the performance of our system in 2 setting: 1 UAV and 3 UAVs to compare the performance of the 1 robot system and multi-agents system.

In each setting, we test behavior of the UAVs in 3 modes:

- Easy mode: 1 target (person) at a time.
- Hard mode: 3 targets in the crowd moving with random speed and random direction.
- Crazy mode: randomly generate a lot of targets and let them move with random speed and random direction.

Then we record the number of success and failure and display it on the interface of the simulator. For each mode, the total number of crowd was generated was about 200 people.

This 3 modes is the idea of team Yue Zhuo and Han Wang. We use that to create our performance test.

5.1 Results: Blocking Rate

We record the blocking rate of the system and compare when using 1 robot versus 3 robots. This statics based on running the system where the total number of crowd was generated was about 200 people.

1 UAV

	Easy Mode	Hard Mode	Crazy Mode
Blocking Rate	100%	99%	84%

3 UAVs

	Easy Mode	Hard Mode	Crazy Mode
Blocking Rate	100%	100%	100%