Final Report: RoboViz Capstone Project

Boyd Kane
KNXBOY001
KNXBOY001@myuct.ac.za

Imaad Ghoor
GHRIMA002
GHRIMA002@myuct.ac.za

Jesse Sarembock
SRMJES001
SRMJES001@myuct.ac.za

The RoboViz Project extends an existing open source genetic evolution platform (RoboGen) to permit the visualisation of multiple robots, called a swarm. The RoboViz Project accepts multiple parameter to fine-tune the simulation of the swarm, and adheres to OOP best practices.

Contents

1.   Introduction

This project - RoboViz - involves extending an existing visualiser [**robogen**] to enable the visualisation of multiple of multiple robots simultaneously. Robogen allows researchers to define a robot structure and then make use of genetic algorithms (paired with a fitness function) to evolve robots that that gradually perform better (as a measured by the relevant fitness function) as more generations of robots are simulated.

After a set number of generations, the final robot can also be visualised, although currently the software only allows for the visualisation of a single robot. The software also generates STL files which describe the 3D body parts of the robot, such that a 3D printer can take those files and 3D print the body components of the evolved robot. INO files are also generated, which can be loaded onto the Arduino platform and define the robot's 'brain' as a software defined artificial neural network which was evolved by the genetic algorithm.

This project involves modifying the source code of the RoboGen software and proving that the modifications are efficient enough for at least 3 (but preferably more) robots to be simulated at once.

An agile software development approach was taken to develop this project. The project team has worked over WhatsApp and MS Teams, informing each other on their work and designating tasks from there. Time was spent on creating functional code to implement a swarm of robots and testing that code before adding more functionality. The team has had frequent meetings with project stakeholders, receiving feedback directly from the stakeholders while developing the project. Throughout development, there have been a significant number of changes needed as development progressed and these were added to the project plan over time.

A vertical prototype was chosen for this project since it focuses on implementing a specific feature - swarm of robots in the visualiser - this was the most appropriate prototype as it tests key components during early stages of the project to check key functions.

The Doxygen documentation tool [**doxygen**] was also added to the project, in order to automatically generate documentation from the source code of the project. Following this, the source code was annotated with well formatted comments that could be parsed by Doxygen. The html documentation can be found in `src/docs/html/index.html` (but it is not kept under version control).

2. Requirements Captured

2.1. Functional Requirements

The final project must be able to visualise at least 3 robots simultaneously. The morphology and neural network defining these robots must be defined in a file as per existing RoboGen guidelines for defining robots (as either `json` or specially formatted `txt` files). The user must be able to zoom in and out of the simulation, as well as pan across to view different parts of the simulation with more clarity. The user must also be able to pause and unpause the simulation.

2.2. Non-functional Requirements

The user should be able to interact smoothly with the simulation once started. That is, the simulation should not close before the specified simulation duration is over. The simulation should start within 2 minutes of running the `robogen-file-viewer` executable. On an adequately powerful machine, the swarm should be simulated at more than 15 frames per second.

2.3. Usability Requirements

While the simulation is running, console based output should inform the user of the details of the simulation, for example, when robots are added to the swarm or which configuration files are being read. This output should be well formatted and provide information on various levels useful for finding problems, warning the user about potential issues, and if an error occurs, providing the user with sufficient information to solve the error.

2.4. Use Case Narratives

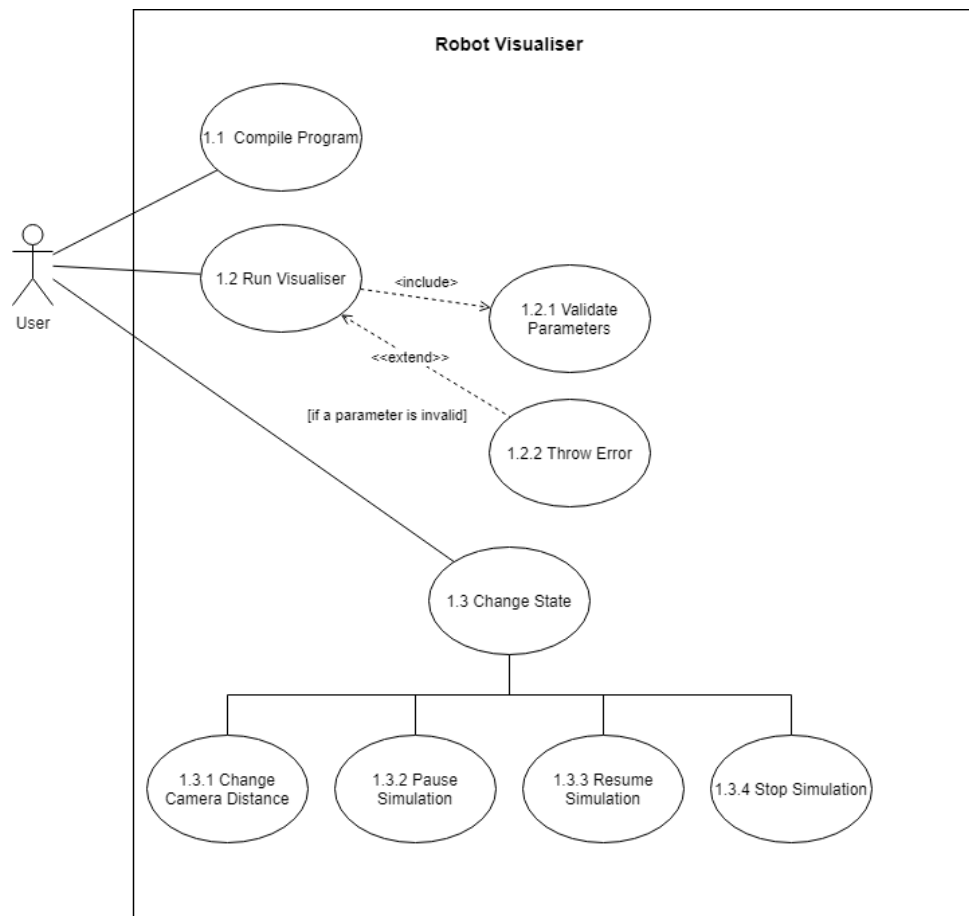See Figure 1 for the Use Case Diagram.

Figure 1: Use Case Diagram

2.4.1.   Start Simulation of the Swarm. Actor: User

Primary Path: The user runs the program and is prompted to enter the name of a robot file and a configuration file (which includes how many robots to simulate in the swarm). The program loads a new window that is the visualiser, and displays the specified number of robots performing their tasks from the same robot file. While the simulation is running, the user may pause the simulation and zoom into the area where the robots are performing for a closer view.

Alternative Path: If the robot file entered is incorrectly formatted or does not exist then the program will throw an error and return the command line help page.

2.4.2.   Run Visualiser. Actor: User

Primary Path: The user starts the visualiser, after that, the user needs to enter 2 parameters, location of the file that defines the robot and the location of the file that defines requirements and configurations for the file viewer. Should one of the parameters entered be invalid or non-existent, an exception will be thrown, requiring the user to enter those parameters again.

2.4.3.   View Simulation. Actor: User

Primary Path: Once the simulation displays the robots performing their tasks in the run-time environment. The user may change the state of the view, by pausing,resuming and stopping the simulation. The user may also zoom in closer to the robots or zoom further away. The simulation will stop once the time limit specified in the configuration file has passed.

3.   Design Overview

Given that the project authors were extending the existing RoboGen [**robogen**] code base, changing some existing aspects of the existing software was considered out of scope due to time constraints and for existing RoboGen users to easily learn to use the RoboViz project.

3.1.   Layered Architecture Diagram

3.2.   Swarm Class

3.3.   SwarmPositionsConfig Class

3.4.   Analysis Class Diagram

See the Analysis Class Diagram in Figure 2

4.   Implementation

4.1.   Overview

The changes made to the existing project will be covered in the same order as they are encountered when a user runs the final executable.

The `robogen-file-viewer` is the executable used to run the simulation such that the swarm can be watched in real time. This executable had to be modified to read in some additional parameters (for example: `swarmSize`, `swarmPositions`) via configuration files, and then to store those parameters in the existing `RobogenConfig` object along with the other configurations. Since the positions of the robots in the swarm has to be specified in a separate file, this also required creating a new file configuration reader (`SwarmPositionsConfig`) to parse the swarm positions. This file was created to be similar to the existing `ObstaclesConfig` and `LightSourcesConfig` objects, and was integrated to the existing `ConfigurationReader`.

Once the configurations for a swarm are read in, error checking is done on those configurations and the program exits if the user requests an invalid combination of options (for example, a negative `swarmSize`). At this point various logging objects are initialised, and these were modified to log the details of every robot in the swarm.

At this stage the robots are initialised, with source files and swarm size specified in the above mentioned configuration files. Each robot's sensors, actuators, and its neural network are also initialised here. This can take some time, and so sufficient log output is printed to keep the user informed.
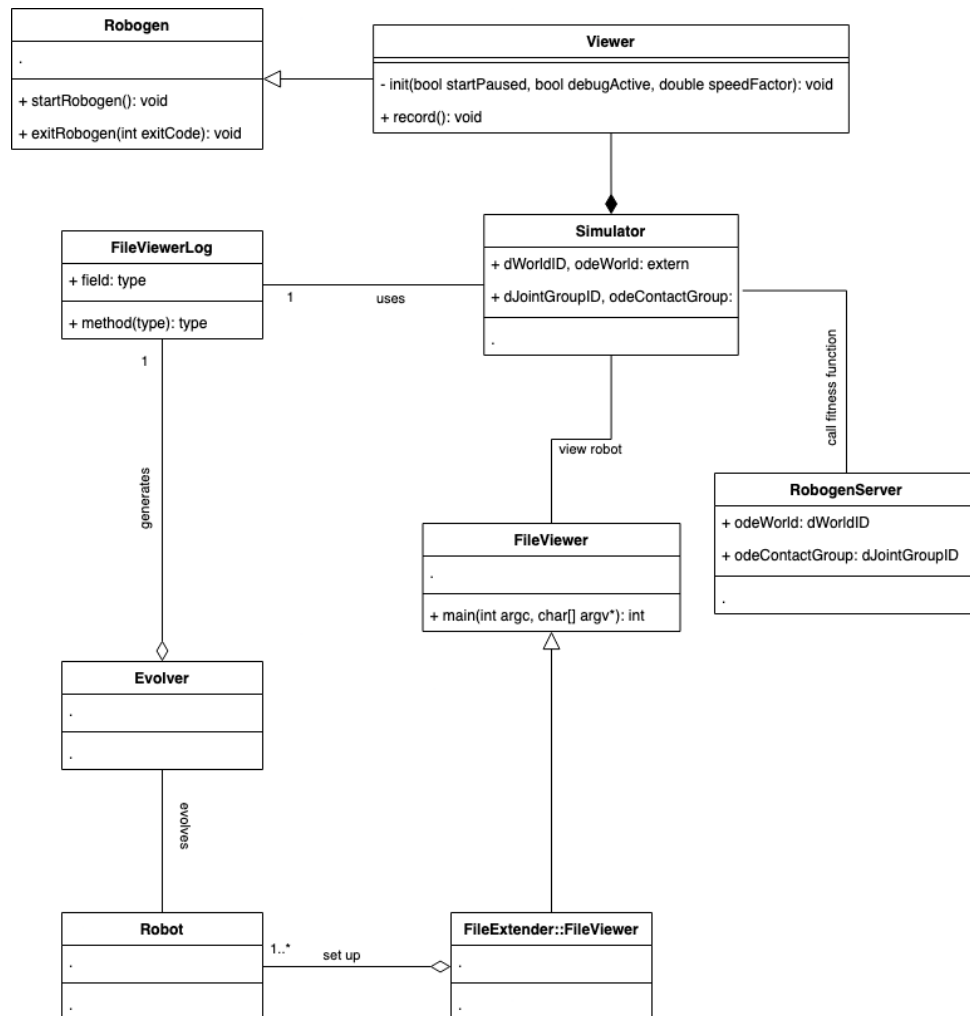
Figure 2: Analysis Class diagram

Now that everything is initialised, the main loop of the simulation begins. The simulation can run once (as is the case for a simple visualisation) or it can run multiple times (as is the case for when a population of swarms are being evaluated by the evolver. There are various protections in place to ensure the swarm is not taking advantage of errors in the physics simulation, which were developed to handle more than one robot. Additionally the evaluation of the individual robot's neural networks had to be extended to work for more than just one robot.

On completion of the simulation, the memory required by the dynamics engine [**ODE**] and the RoboViz swarm is freed.

4.2.   Data Structures Used

4.2.1.   Swarm Class

The Swarm class (`src/Swarm`) is the internal representation of a collection of robots. The members of the swarm are assumed to be attempting to cooperate with one another, and receive a single fitness score at the end of a scenario.

4.2.2.   RobogenConfig Class

The simulator is designed to be flexible, and as such requires a lot of configuration parameters to be specified. These configurations are stored in the `RobogenConfig` object, often named `config` in the code base. However, the `RobogenConfig` object is only responsible for storing the well formatted and easily accessible parameters. The responsibility of parsing those parameters is handed to the `ConfigurationReader` class:

4.2.3.   ConfigurationReader Class

The configuration parameters for the scenario (like the type and size of terrain, the starting positions of the robot(s), the fitness function to use) are parsed by the `ConfigurationReader`. Some of the parameters are specified directly in the configuration file as key-value pairs separated by an equals = symbol, while other parameters have values referencing relative file paths where a list of values can be found. These external parameters are parsed separately by different config classes (for example, `ObstaclesConfig`, `LightSourcesConfig`, and `SwarmPositionsConfig`.

4.2.4.   SwarmPositionsConfig

The x,y,z locations of the individual members of the swarm can be specified in a separate file, and this file is parsed by the `SwarmPositionsConfig` object. The object is instantiated as a member of the `ConfigurationReader`, and these configurations can later be accessed during simulations.

4.2.5.   Scenario

A scenario is the combination of a swarm in rigid body simulation with a fitness function. When a scenario is initialised, the swarm is instantiated into the task environment, and when the simulation starts the swarm will be monitored by the scenario so that it's fitness can be calculated. A custom fitness function can be defined by creating a javascript file with the appropriate callbacks, or one of `racing` (fitness is proportional to average distance from the starting position) or `chasing` (fitness is inversely proportional to distance from the nearest moving light source) can be chosen. These fitness functions are defined in the various files found in `src/scenario/`

4.2.6.   FileViewer Class

The `FileViewer` is the command line entry point for starting a simulation. It takes in two command line arguments (the path of the file defining the robots, and the path of the configuration file), parses those arguments using `boost` program options [**boost**]. This is the file that becomes the executable used to start the simulator for the purpose of viewing a swarm.

4.2.7.   IViewer Class

This is an interface defining common functionality for classes wishing to view the current state of a simulation. For example, the simulation can be viewed in the web browser or on desktop.

4.2.8.   Simulator

The `Simulator` has a single (overloaded) method, `runSimulations` which receives a scenario, a RobogenConfig, and optionally a viewer. This method then initialises the viewer (if applicable), each robot in the swarm, the rigid body simulator, ODE, and sets up the scenario in anticipation of calculating the fitness of the swarm. The simulation is then started up, and run for a duration specified in the `RobogenConfig` object. This duration from starting the simulation in the world until the simulation is destroyed is called a trial. The simulator can be specified to perform multiple trials, or just one.

When the specified number of trials has been completed, the dynamics engine is closed, and all resources are cleaned up.

4.2.9.   robogen.proto

Most objects in the Robogen project can be serialised into Google Protocol Buffer Messages [**protobuf**], which provide a language and platform neutral method for serialising structured data, similar to XML but faster and simpler. The file `robogen.proto` contains definitions of the structure of the data, and was extended to allow the swarm class to be serialised.

4.3.   Overview of the User Interface

4.4.   Most Significant Methods of Each Class

4.5.   Special Relationships between Classes

5.   Program Validation and Verification

5.1.   How the system was tested

The testing frameworks `GoogleTest` and `GoogleMock` were used to run tests on the code base, and these tests were integrated with the existing `CMake` files so that tests are compiled automatically and can be run via the command `ctest`. Testing covered some aspects of the existing code base but due to it's size (164 thousand source lines of code) the primary focus was on the additions made by the RoboViz project in the `Swarm` and `ConfigurationReader` classes. These test are run automatically when the rest of the code base is compiled, so that no changes can get through which break existing functionality.

5.2.   How we know the system works

The RoboViz team has confidence that the existing system works due to the tests that have been put in place to assure us of this. If something were incorrect, the tests would pick it up automatically. Since the test run automatically, there is no scope for human error or judgement which can sometimes cause errors to creep in.

5.3.   Type of testing, and method of test execution

The existing code base was not structured to allow for easy testing. The levels of coupling between different objects was very high and there exist many methods which have a large number of side effects, reducing the atomicity of any tests that would be run on them. The focus of the testing was on unit tests, specifically for the additions to the code base which were made by the RoboViz team. These unit tests were written using the `GoogleTest` testing framework, and were integrated with `CMake` so that they would compile and run automatically.

5.4.   Justification of the chosen testing plan

The testing plan was chosen to be effective yet efficient, given that complete test coverage was infeasible due to the size of the project and the duration of time given to complete the project. The existing test were written to cover the most likely of errors and the most fatal of errors of the existing code base, as well as to cover the newly written code so that problems have the greatest likelihood of causing a test to fail.

5.5.   Use of mock data in the tests

Mock data was used in the form of valid or invalid input files, stored in `examples/test_cases/` as files named like the tests that use them. These data provide common mistakes made when specifying the input files, and the tests are written to check against these common errors.

5.6.   Full detail of the test runs

The full details of the test runs can be found in the Appendix, section A.

5.7.   Usefulness of the system based on the unit tests

The system is considered by the authors to be more useful and more error-save (thereby more usable) than the original project for various reasons. Firstly, the addition of the tests and the setup of the testing framework means that any problems in the future will be found by the existing tests, and adding new unit tests to cover future expansions are made easier. Secondly, the extensive documentation (in the code base in the form of Doxygen comments, in the `CONTRIBUTING` file, and in the commandline-help that gets printed when a user specifies the wrong arguments) mean that future users will easily be able to get started, as we have extensively logged our troubles with the code base (and the solutions to those troubles) via these means.

Table 1: Summary Testing Plan

Process	Technique
1. Class Testing: test methods and state behaviour of classes	Via class-by-class tests in the appropriately named test files
2. Integration Testing: test the interaction of sets of classes	Due to the highly-coupled design of the existing code base and the extensive side effects that occur when classes interact, integration testing had to be covered by the new unit tests.
3. Validation Testing: test whether customer requirements are satisfied	The client expressed satisfaction with the prototype and agreed with our plans for further development after the prototype. The client's requirements were manually considered and found to be satisfied.
4. System Testing: test the behaviour of the system as part of a larger environment	The testing framework was executed on different environments, resulting in the stability of the platform in different situations being established.

6.   Group Contributions

6.1.   Boyd Kane - KNXBOY001

Boyd made changes to the code base, the final report, and was in charge of rebasing the feature git branches onto the master branch. Boyd set up the Doxygen documentation system and added properly formatted documentation comments to the majority of the code base. Boyd set up the GoogleTest testing framework and wrote the unit tests contained in `SwarmTest.cpp` and `ConfigurationReaderTest.cpp`. Boyd changed the Google Protocol Buffer definitions to handle the new Swarm object, and added code to allow the ConfigurationReader to parse the new parameters required by the project brief that did not exist in the original project. Boyd changed the `Simulator` to accept a swarm instead of a collection of robots. Boyd changed the `Scenario` initialisation to accept and initialise a swarm of robots, and to handle the edge cases arising from that. Boyd standardised the console log messages to have a common format, and increased the amount of output so that the program would be more helpful to the user. Boyd changed the `FileViewer` and created the `SwarmPositionsConfig` so that a separate file containing a list of swarm starting positions can be specified.

Table 2: A table of tests. TODO Complete Caption

Test File Name	Test Name	Test Description
SwarmTest	OnInitThenSizeIsZero	Check swarm size is zero on initialisation.
SwarmTest	OnAddRobot ThenSizeIncrements	Check the swarm size variable is incremented when a robot is added to the swarm
SwarmTest	OnAddRobot ThenReturnsCorrectRobot	Check when a new robot is added to the swarm, then that robot is returned by the appropriate `getRobot(i)` call.
ConfigurationReaderTest	DisplaysHelp	Check the ConfigurationReader will display help to the user when they input invalid commandline arguments.
ConfigurationReaderTest	ParsesSwarmSize	Check the ConfigurationReader correctly reads in and parses the swarm size from the configuration file.
ConfigurationReaderTest	ParsesRacingScenario	Check the ConfigurationReader correctly reads in and parses the racing scenario from the configuration file.
ConfigurationReaderTest	ParsesChasingScenario	Check the ConfigurationReader correctly reads in and parses the chasing scenario from the configuration file.
ConfigurationReaderTest	ReadsSwarmPosFile	Check the ConfigurationReader correctly reads in and parses the swarm position file from the configuration file.
ConfigurationReaderTest	ThrowsOnBadSwarmSize	Check the ConfigurationReader throws an error when a bad swarm size and swarmPositionFile combination is used in the configuration file.
ConfigurationReaderTest	ParsesGatheringZone	Check the ConfigurationReader correctly reads in and parses the gathering zone file from the configuration file.
ConfigurationReaderTest	ParsesGatheringZonePos	Check the ConfigurationReader correctly reads in and parses the gathering zone position from the configuration file.
ConfigurationReaderTest	ParsesGatheringZoneSize	Check the ConfigurationReader correctly reads in and parses the gathering zone size from the configuration file.

6.2. Imaad Ghoor - ABCXYZ001

6.3. Jesse Sarembock - ABCXYZ001

7. Conclusion

Here we must summarize everything, and provide a conclusion to the project's initial aims and goals.

8. Bibliography

A. Full Test Runs

```
UpdateCTestConfiguration  from :/home/boyd/roboviz/build/DartConfiguration.tcl
UpdateCTestConfiguration  from :/home/boyd/roboviz/build/DartConfiguration.tcl
Test project /home/boyd/roboviz/build
Constructing a list of tests
Done constructing a list of tests
Updating test list for fixtures
Added 0 tests to meet fixture requirements
Checking test dependency graph...
Checking test dependency graph end
test 1
      Start  1: SwarmTest.OnInitThenSizeIsZero

1: Test command: /home/boyd/roboviz/build/SwarmTest "--gtest_filter=SwarmTest.On
1: Test timeout computed to be: 10000000
1: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
1: Note: Google Test filter = SwarmTest.OnInitThenSizeIsZero
1: [==========] Running 1 test from 1 test suite.
1: [----------] Global test environment set-up.
1: [----------] 1 test from SwarmTest
1: [ RUN      ] SwarmTest.OnInitThenSizeIsZero
1: [       OK ] SwarmTest.OnInitThenSizeIsZero (0 ms)
1: [----------] 1 test from SwarmTest (0 ms total)
1:
1: [----------] Global test environment tear-down
1: [==========] 1 test from 1 test suite ran. (0 ms total)
1: [  PASSED  ] 1 test.
 1/12 Test  #1: SwarmTest.OnInitThenSizeIsZero ...................   Passed
test 2
      Start  2: SwarmTest.OnAddRobotThenSizeIncrements

2: Test command: /home/boyd/roboviz/build/SwarmTest "--gtest_filter=SwarmTest.On
2: Test timeout computed to be: 10000000
2: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
2: Note: Google Test filter = SwarmTest.OnAddRobotThenSizeIncrements
2: [==========] Running 1 test from 1 test suite.
2: [----------] Global test environment set-up.
2: [----------] 1 test from SwarmTest
2: [ RUN      ] SwarmTest.OnAddRobotThenSizeIncrements
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
```

```
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
```

```
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [D] Adding robot to swarm
2: [          OK ] SwarmTest.OnAddRobotThenSizeIncrements (0 ms)
2: [----------] 1 test from SwarmTest (0 ms total)
2:
2: [----------] Global test environment tear-down
2: [==========] 1 test from 1 test suite ran. (0 ms total)
2: [  PASSED  ] 1 test.
 2/12 Test  #2: SwarmTest.OnAddRobotThenSizeIncrements ............   Passed
test 3
```

```
        Start  3: SwarmTest.OnAddRobotThenReturnsCorrectRobot


3: Test command: /home/boyd/roboviz/build/SwarmTest "--gtest_filter=SwarmTest.On
3: Test timeout computed to be: 10000000
3: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
3: Note: Google Test filter = SwarmTest.OnAddRobotThenReturnsCorrectRobot
3: [==========] Running 1 test from 1 test suite.
3: [----------] Global test environment set-up.
3: [----------] 1 test from SwarmTest
3: [ RUN      ] SwarmTest.OnAddRobotThenReturnsCorrectRobot
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
```

```
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
```

```
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [D] Adding robot to swarm
3: [       OK ] SwarmTest.OnAddRobotThenReturnsCorrectRobot (0 ms)
3: [----------] 1 test from SwarmTest (0 ms total)
3:
3: [----------] Global test environment tear-down
3: [==========] 1 test from 1 test suite ran. (0 ms total)
3: [  PASSED  ] 1 test.
 3/12 Test  #3: SwarmTest.OnAddRobotThenReturnsCorrectRobot .......   Passed
test 4
      Start  4: ConfigurationReaderTest.DisplaysHelp

4: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filte
4: Test timeout computed to be: 10000000
4: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
4: Note: Google Test filter = ConfigurationReaderTest.DisplaysHelp
4: [==========] Running 1 test from 1 test suite.
4: [----------] Global test environment set-up.
4: [----------] 1 test from ConfigurationReaderTest
4: [ RUN      ] ConfigurationReaderTest.DisplaysHelp
4: [I] Parsing config file: help
4: Allowed options for Simulation Config File:
4:    --scenario arg                      Experiment scenario: (racing, chasing, or
4:                                        provided js file)
4:    --timeStep arg                      Time step duration (s)
4:    --nTimeSteps arg                    Number of timesteps (Either this or
4:                                        simulationTime are required)
4:    --simulationTime arg                Length of simulation (s)  (Either this or
4:                                        nTimeSteps are required)
4:    --terrainType arg                   Terrain type: flat or rugged
4:    --terrainHeightField arg            Height Field for terrain generation
4:    --terrainWidth arg                  Terrain width
4:    --terrainHeight arg                 Terrain height
4:    --terrainLength arg                 Terrain length
4:    --terrainFriction arg               Terrain Friction Coefficient
4:    --startPositionConfigFile arg       Start Positions Configuration File
4:    --obstaclesConfigFile arg           Obstacles configuration file
4:    --lightSourcesConfigFile arg        Light sources configuration file
4:    --actuationFrequency arg            Actuation Frequency (Hz)
4:    --sensorNoiseLevel arg              Sensor Noise Level:
4:                                         Sensor noise is Gaussian with std dev of
4:                                        sensorNoiseLevel * actualValue.
4:                                        i.e. value given to Neural Network is N(a
4:                                        a * s)
4:                                        where a is actual value and s is
4:                                        sensorNoiseLevel
```

```
4:   --motorNoiseLevel arg            Motor noise level:
4:                                    Motor noise is uniform in range
4:                                    +/-(motorNoiseLevel * actualValue)
4:   --capAcceleration arg            Flag to enforce acceleration cap.Useful f
4:                                    preventing unrealistic  behaviors /
4:                                    simulator exploits
4:   --maxLinearAcceleration arg      Maximum linear acceleration (if
4:                                    capAcceleration. is true
4:   --maxAngularAcceleration arg     Maximum angular acceleration (if
4:                                    capAcceleration. is true
4:   --maxDirectionShiftsPerSecond arg Maximum number of direction shifts per
4:                                    second for testing motor burnout.  If not
4:                                    set, then there is no cap
4:   --gravity arg                    Gravity: either a single z-value for
4:                                    g=(0,0,z) or x,y,z (comma separated) for
4:                                    full g vector. Specified in m/(s^2)
4:                                    Defaults to (0,0,-9.81)
4:   --disallowObstacleCollisions arg Flag to enforce no obstacle collisions.
4:                                    true then any obstacle collision will be
4:                                    considered a constraint violation. (defau
4:                                    false).
4:   --swarmSize arg                  The number of duplicate robots to include
4:                                    in the simulation. Defaults to 1 if not
4:                                    specified
4:   --gatheringZoneSize arg          The size as an 'x,y,z' string of the
4:                                    gathering zone, which is an area highligh
4:                                    in a special color, useful for certain
4:                                    fitness functions
4:   --gatheringZonePosition arg      The center as an 'x,y,z' string of the
4:                                    gathering zone, which is an area
4:                                    highlighed in a special color,  useful fo
4:                                    certain fitness functions
4:   --resourcesConfigFile arg        A config file containing a list of
4:                                    resources, with one resourceper line. Eac
4:                                    line must contain a list of space-separat
4:                                    floating point values defining the resour
4:                                    in the order:x-pos y-pos z-pos x-magnitud
4:                                    y-magnitude z-magnitude unknown unknown
4:   --swarmPositionsConfigFile arg   A configuration file containing the
4:                                    positions of each robot in the swarm. Wil
4:                                    take preference over anything specified i
4:                                    startPositionFile
4:   --obstacleOverlapPolicy arg      Defines the policy for handling obstacles
4:                                    enclosed in the robot's initial axis
4:                                    aligned bounding box (AABB).  Options are
4:                                    'removeObstacles' -- obstacles will be
4:                                    removed, and the simulation will proceed
4:                                    (default).
4:                                    'constraintViolation' -- the simulation
4:                                    will be terminated with a constrain
4:                                    violation.
```

```
4:                                              'elevateRobot' -- the robot will be
4:                                              elevated to be above all obstacles before
4:                                              the simulation begins.
4:
4: [       OK ] ConfigurationReaderTest.DisplaysHelp (0 ms)
4: [----------] 1 test from ConfigurationReaderTest (0 ms total)
4:
4: [----------] Global test environment tear-down
4: [==========] 1 test from 1 test suite ran. (0 ms total)
4: [  PASSED  ] 1 test.
 4/12 Test  #4: ConfigurationReaderTest.DisplaysHelp ..............   Passed
test 5
      Start  5: ConfigurationReaderTest.ParsesSwarmSize

5: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filte
5: Test timeout computed to be: 10000000
5: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
5: Note: Google Test filter = ConfigurationReaderTest.ParsesSwarmSize
5: [==========] Running 1 test from 1 test suite.
5: [----------] Global test environment set-up.
5: [----------] 1 test from ConfigurationReaderTest
5: [ RUN      ] ConfigurationReaderTest.ParsesSwarmSize
5: [I] Parsing config file: ../examples/test_cases/TestParsesSwarmSize.txt
5: [I] Config file '../examples/test_cases/TestParsesSwarmSize.txt' parsed succe
5: [I] Attempting to parse swarmPositions file: '/media/sf_roboviz/build/../exam
5: [D] SwarmPosition file line 1: 0 0 0
5: [D] SwarmPosition file line 2: 3 0 0
5: [D] SwarmPosition file line 3: 0 3 0
5: No startPositionConfigFile provided so will use a single evaluation with the
5: [I] Undefined 'actuationFrequency' parameter in '../examples/test_cases/TestP
5: [       OK ] ConfigurationReaderTest.ParsesSwarmSize (3 ms)
5: [----------] 1 test from ConfigurationReaderTest (3 ms total)
5:
5: [----------] Global test environment tear-down
5: [==========] 1 test from 1 test suite ran. (3 ms total)
5: [  PASSED  ] 1 test.
 5/12 Test  #5: ConfigurationReaderTest.ParsesSwarmSize ...........   Passed
test 6
      Start  6: ConfigurationReaderTest.ParsesRacingScenario

6: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filte
6: Test timeout computed to be: 10000000
6: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
6: Note: Google Test filter = ConfigurationReaderTest.ParsesRacingScenario
6: [==========] Running 1 test from 1 test suite.
6: [----------] Global test environment set-up.
6: [----------] 1 test from ConfigurationReaderTest
6: [ RUN      ] ConfigurationReaderTest.ParsesRacingScenario
6: [I] Parsing config file: ../examples/test_cases/TestParsesRacingScenario.txt
6: [I] Config file '../examples/test_cases/TestParsesRacingScenario.txt' parsed
6: [I] Attempting to parse swarmPositions file: '/media/sf_roboviz/build/../exam
```

17

```
6: [D] SwarmPosition file line 1: 0 0 0
6: [D] SwarmPosition file line 2: 3 0 0
6: [D] SwarmPosition file line 3: 0 3 0
6: No startPositionConfigFile provided so will use a single evaluation with the
6: [I] Undefined 'actuationFrequency' parameter in '../examples/test_cases/TestP
6: [        OK ] ConfigurationReaderTest.ParsesRacingScenario (2 ms)
6: [----------] 1 test from ConfigurationReaderTest (2 ms total)
6:
6: [----------] Global test environment tear-down
6: [==========] 1 test from 1 test suite ran. (2 ms total)
6: [  PASSED  ] 1 test.
 6/12 Test  #6: ConfigurationReaderTest.ParsesRacingScenario ......   Passed
test 7
       Start  7: ConfigurationReaderTest.ParsesChasingScenario

7: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filte
7: Test timeout computed to be: 10000000
7: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
7: Note: Google Test filter = ConfigurationReaderTest.ParsesChasingScenario
7: [==========] Running 1 test from 1 test suite.
7: [----------] Global test environment set-up.
7: [----------] 1 test from ConfigurationReaderTest
7: [ RUN      ] ConfigurationReaderTest.ParsesChasingScenario
7: [I] Parsing config file: ../examples/test_cases/TestParsesChasingScenario.txt
7: [I] Config file '../examples/test_cases/TestParsesChasingScenario.txt' parsed
7: [I] Attempting to parse swarmPositions file: '/media/sf_roboviz/build/../exam
7: [D] SwarmPosition file line 1: 0 0 0
7: [D] SwarmPosition file line 2: 3 0 0
7: [D] SwarmPosition file line 3: 0 3 0
7: No startPositionConfigFile provided so will use a single evaluation with the
7: [I] Undefined 'actuationFrequency' parameter in '../examples/test_cases/TestP
7: [        OK ] ConfigurationReaderTest.ParsesChasingScenario (2 ms)
7: [----------] 1 test from ConfigurationReaderTest (2 ms total)
7:
7: [----------] Global test environment tear-down
7: [==========] 1 test from 1 test suite ran. (2 ms total)
7: [  PASSED  ] 1 test.
 7/12 Test  #7: ConfigurationReaderTest.ParsesChasingScenario .....   Passed
test 8
       Start  8: ConfigurationReaderTest.ReadsSwarmPosFile

8: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filte
8: Test timeout computed to be: 10000000
8: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
8: Note: Google Test filter = ConfigurationReaderTest.ReadsSwarmPosFile
8: [==========] Running 1 test from 1 test suite.
8: [----------] Global test environment set-up.
8: [----------] 1 test from ConfigurationReaderTest
8: [ RUN      ] ConfigurationReaderTest.ReadsSwarmPosFile
8: [I] Parsing config file: ../examples/test_cases/TestReadsSwarmPosFile.txt
8: [I] Config file '../examples/test_cases/TestReadsSwarmPosFile.txt' parsed suc
```

18

```
8: [I] Attempting to parse swarmPositions file: '/media/sf_roboviz/build/../exam
8: [D] SwarmPosition file line 1: 0 0 0
8: [D] SwarmPosition file line 2: 3 0 0
8: [D] SwarmPosition file line 3: 0 3 0
8: No startPositionConfigFile provided so will use a single evaluation with the
8: [I] Undefined 'actuationFrequency' parameter in '../examples/test_cases/TestR
8: [        OK ] ConfigurationReaderTest.ReadsSwarmPosFile (2 ms)
8: [----------] 1 test from ConfigurationReaderTest (2 ms total)
8:
8: [----------] Global test environment tear-down
8: [==========] 1 test from 1 test suite ran. (2 ms total)
8: [  PASSED  ] 1 test.
 8/12 Test  #8: ConfigurationReaderTest.ReadsSwarmPosFile .........   Passed
test 9
       Start  9: ConfigurationReaderTest.ThrowsOnBadSwarmSize

9: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filte
9: Test timeout computed to be: 10000000
9: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
9: Note: Google Test filter = ConfigurationReaderTest.ThrowsOnBadSwarmSize
9: [==========] Running 1 test from 1 test suite.
9: [----------] Global test environment set-up.
9: [----------] 1 test from ConfigurationReaderTest
9: [ RUN      ] ConfigurationReaderTest.ThrowsOnBadSwarmSize
9: [I] Parsing config file: ../examples/test_cases/TestThrowsOnBadSwarmSize.txt
9: [I] Config file '../examples/test_cases/TestThrowsOnBadSwarmSize.txt' parsed
9: [I] Attempting to parse swarmPositions file: '/media/sf_roboviz/build/../exam
9: [D] SwarmPosition file line 1: 0 0 0
9: [D] SwarmPosition file line 2: 3 0 0
9: [D] SwarmPosition file line 3: 0 3 0
9: No startPositionConfigFile provided so will use a single evaluation with the
9: [I] Undefined 'actuationFrequency' parameter in '../examples/test_cases/TestT
9: [E] The swarmSize parameter doesn't equal the length of the swarmPositionsFil
9: [        OK ] ConfigurationReaderTest.ThrowsOnBadSwarmSize (1 ms)
9: [----------] 1 test from ConfigurationReaderTest (1 ms total)
9:
9: [----------] Global test environment tear-down
9: [==========] 1 test from 1 test suite ran. (1 ms total)
9: [  PASSED  ] 1 test.
 9/12 Test  #9: ConfigurationReaderTest.ThrowsOnBadSwarmSize ......   Passed
test 10
       Start 10: ConfigurationReaderTest.ParsesGatheringZone

10: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filt
10: Test timeout computed to be: 10000000
10: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlet
10: Note: Google Test filter = ConfigurationReaderTest.ParsesGatheringZone
10: [==========] Running 1 test from 1 test suite.
10: [----------] Global test environment set-up.
10: [----------] 1 test from ConfigurationReaderTest
10: [ RUN      ] ConfigurationReaderTest.ParsesGatheringZone
```

```
10: [I] Parsing config file: ../examples/test_cases/TestParsesGatheringZone.txt
10: [I] Config file '../examples/test_cases/TestParsesGatheringZone.txt' parsed
10: [I] Attempting to parse swarmPositions file: '/media/sf_roboviz/build/../exam
10: [D] SwarmPosition file line 1: 0 0 0
10: [D] SwarmPosition file line 2: 3 0 0
10: [D] SwarmPosition file line 3: 0 3 0
10: No startPositionConfigFile provided so will use a single evaluation with the
10: [I] Undefined 'actuationFrequency' parameter in '../examples/test_cases/Test
10: [W] parameter gatheringZonePos has been specified but the code for using it
10: [W] parameter gatheringZoneSize has been specified but the code for using it
10: [      OK ] ConfigurationReaderTest.ParsesGatheringZone (2 ms)
10: [----------] 1 test from ConfigurationReaderTest (2 ms total)
10:
10: [----------] Global test environment tear-down
10: [==========] 1 test from 1 test suite ran. (2 ms total)
10: [  PASSED  ] 1 test.
10/12 Test #10: ConfigurationReaderTest.ParsesGatheringZone .......   Passed
test 11
        Start 11: ConfigurationReaderTest.ParsesGatheringZonePos

11: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filt
11: Test timeout computed to be: 10000000
11: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
11: Note: Google Test filter = ConfigurationReaderTest.ParsesGatheringZonePos
11: [==========] Running 1 test from 1 test suite.
11: [----------] Global test environment set-up.
11: [----------] 1 test from ConfigurationReaderTest
11: [ RUN      ] ConfigurationReaderTest.ParsesGatheringZonePos
11: [I] Parsing config file: ../examples/test_cases/TestParsesGatheringZonePos.t
11: [I] Config file '../examples/test_cases/TestParsesGatheringZonePos.txt' pars
11: [I] Attempting to parse swarmPositions file: '/media/sf_roboviz/build/../exam
11: [D] SwarmPosition file line 1: 0 0 0
11: [D] SwarmPosition file line 2: 3 0 0
11: [D] SwarmPosition file line 3: 0 3 0
11: No startPositionConfigFile provided so will use a single evaluation with the
11: [I] Undefined 'actuationFrequency' parameter in '../examples/test_cases/Test
11: [W] parameter gatheringZonePos has been specified but the code for using it
11: [W] parameter gatheringZoneSize has been specified but the code for using it
11: [      OK ] ConfigurationReaderTest.ParsesGatheringZonePos (1 ms)
11: [----------] 1 test from ConfigurationReaderTest (1 ms total)
11:
11: [----------] Global test environment tear-down
11: [==========] 1 test from 1 test suite ran. (1 ms total)
11: [  PASSED  ] 1 test.
11/12 Test #11: ConfigurationReaderTest.ParsesGatheringZonePos ....   Passed
test 12
      Start 12: ConfigurationReaderTest.ParsesGatheringZoneSize

12: Test command: /home/boyd/roboviz/build/ConfigurationReaderTest "--gtest_filt
12: Test timeout computed to be: 10000000
12: Running main() from /home/boyd/roboviz/src/googletest-release-1.11.0/googlete
```

```
12: Note: Google Test filter = ConfigurationReaderTest.ParsesGatheringZoneSize
12: [==========] Running 1 test from 1 test suite.
12: [----------] Global test environment set-up.
12: [----------] 1 test from ConfigurationReaderTest
12: [ RUN      ] ConfigurationReaderTest.ParsesGatheringZoneSize
12: [I] Parsing config file: ../examples/test_cases/TestParsesGatheringZoneSize.t
12: [I] Config file '../examples/test_cases/TestParsesGatheringZoneSize.txt' par
12: [I] Attempting to parse swarmPositions file: '/media/sf_roboviz/build/../exar
12: [D] SwarmPosition file line 1: 0 0 0
12: [D] SwarmPosition file line 2: 3 0 0
12: [D] SwarmPosition file line 3: 0 3 0
12: No startPositionConfigFile provided so will use a single evaluation with the
12: [I] Undefined 'actuationFrequency' parameter in '../examples/test_cases/Test]
12: [W] parameter gatheringZonePos has been specified but the code for using it :
12: [W] parameter gatheringZoneSize has been specified but the code for using it
12: [       OK ] ConfigurationReaderTest.ParsesGatheringZoneSize (2 ms)
12: [----------] 1 test from ConfigurationReaderTest (2 ms total)
12:
12: [----------] Global test environment tear-down
12: [==========] 1 test from 1 test suite ran. (2 ms total)
12: [  PASSED  ] 1 test.
12/12 Test #12: ConfigurationReaderTest.ParsesGatheringZoneSize ...   Passed

100% tests passed, 0 tests failed out of 12

Total Test time (real) =   0.17 sec
```

B.   User Manual

This is the User manual