SIMULATION OF FORWARD AIR CONTROLLER MISSIONS WITH POP-UP TARGETS

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Sandeep Sikharam

In Partial Fulfillment
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

May 2013

Fargo, North Dakota

North Dakota State University
Graduate School

Title

SIMULATION OF FORWARD AIR CONTROLLER MISSIONS WITH POP-UP TARGETS

By

Sandeep Sikharam

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Kendall Nygard
Chair

Dr. Simone Ludwig

Dr. Sumathy Krishnan

Approved:

05-10-2013
Date

Dr. Brian M. Slator
Department Chair

ABSTRACT

Orion is a Java-based framework that is used to develop different battlefield scenarios that involve unmanned air vehicles. There are currently four missions in the Orion framework. They are the track, sweep, patrol, and forward air controller missions.

We focus primarily on the forward air controller (FAC) mission in this paper. The goal of this paper is to add new agents to the FAC mission and to extend its functionality. New elements that will be added to the FAC mission are sensor stations and pop-up targets. The sensor stations have the ability to detect targets or enemies that fall within the sensing area and to recruit an unmanned air vehicle to destroy the detected targets. Pop-up targets are static targets that are not visible and are, therefore, undetectable by unmanned air vehicles and sensors at the start of the mission, and the targets pop-in to the search area at a later time.

ACKNOWLEDGEMENTS

I would like to thank Dr. Kendall Nygard for his continued support, help, and direction. Also, I would like to acknowledge the guidance given to me by others on Dr. Nygard's research team. My sincere thanks go to Dr. Kendall Nygard, Dr. Simone Ludwig, and Dr. Sumathy Krishnan for serving on the committee.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

This paper focuses on the forward air controller mission. The forward air controller missions were used in the Vietnam War. Forward air control (FAC) provided aircraft to ensure the safety of friendly troops and attacked the intended targets. The FAC aircraft flew at low altitudes to maintain aerial surveillance. The low altitudes made the FAC missions dangerous because flying at low altitudes made the aircraft open to enemy fire [6].

Unmanned air vehicles (UAVs) are aircraft that can be flown without pilots on board. These UAVs, also called drones, are constantly monitored and controlled by pilots on the ground at a ground control station [7]. UAVs have gained popularity in modern warfare because of their ability to independently detect and kill targets in an Area of Interest. Technological growth has made it possible to substitute manned aircraft with unmanned aircraft for certain kinds of mission. The aircraft are flown into the Area of Interest with prior knowledge of the area. The aircraft also exhibit certain behaviors, such as patrolling, signaling or communication, collision avoidance, and attacking, while in the field of interest [8].

The objective of this paper is to design, develop, implement, validate, and test a simulation environment with the following characteristics:

a. Sensor stations are added to the Area of Interest in the forward air controller mission.
b. Sensor stations have circulation sensing areas. When an enemy or target falls within this circular sensing area, it is detected by the corresponding sensor station.
c. The sensor stations can signal and recruit killer UAVs to perform a strike and to destroy targets that the stations have detected.
d. Pop-up targets are added to the Area of Interest.

e. Pop-up targets are initially invisible and cannot be detected by hunter UAVs or sensor stations.

f. Pop-up targets are added to the search area randomly and can be detected by hunter UAVs and sensor stations after they are added to the simulation. Hunter UAVs and sensor stations can recruit killer UAVs to perform strikes and destroy these targets.

g. Hunter UAVs are deployed in pairs and perform a search in the Area of Interest. Upon detecting a threat or target, the hunter UAV signals the killer UAV to perform a strike.

h. Sensor stations and hunter UAVs work simultaneously and search for targets, recruiting killer UAVs to destroy targets that have been found.

2. EXISTING ARCHITECTURE

There has been considerable work done on the Orion simulation framework. This paper adds to the research on the command and control of many unmanned air vehicles. Orion is a Java-based simulation framework, and it was built by a research team at North Dakota State University. The framework already includes several scenarios that are meant to be seen as an implementation of real-time battlefield scenarios. The simulation includes several elements, or agents, such as the UAVs (which can be of two types: the hunter UAV and the strike, or killer, UAV), targets (which can also be of two types: static targets or moving targets), and the search area, or the Area of Interest (AOI). Each of these elements has behaviors, and work together in an environment. Joseph Schlecht and Weifing Xu are students who contributed to the simulation framework, along with staff members Doug Schesvold and Jingpeng Tang, and faculty members Kendall E. Nygard and Karl Altenburg [1]. The simulation is also explained in [1][2]. Information regarding agent-based frameworks can be found in [1][3].

The framework consists of four main components. These primary components are as follows.

a. Agents
b. Behaviors
c. Communication
d. Environment

3. COMPONENTS OF THE ORION SIMULATION FRAMEWORK

As explained in the previous chapter, the Orion framework consists of four main components: agents, behaviors, communications, and the agents' environment. Each component is explained in the following sections.

3.1. Agents

Each individual entity that exists in the system is an agent. These entities are independent and have the capability to move, set speed, change direction, and send or receive signals. An abstract class is used to model the basic agent, and therefore, objects of an agent class cannot be created. Each agent class has a run method that executes the associated agent's behavior. All agents in the system are derived from the basic agent [1].

3.2. Behaviors

Each agent in the system has some behavior associated with it. The run method of each agent executes the agent's behavior. There is a finite number of behaviors associated with each agent, and therefore, behaviors can be thought of as a finite state machine. The executed behavior determines the agent's state, and at any point of time, an agent is in precisely one of several available states [1].

3.3. Communication

The communication component, in turn, consists of four main components. They are as follows:

a. Transmitters
b. Receivers

c. Signals

d. Messages

The transmitters and receivers are used to represent devices. These devices can be sensors, actuators, and radio transmitters. These devices are capable of sending and receiving signals, and can be added to agents. The system can have one of three different kinds of signals, although other signals could still be added. The signals contain the message being transmitted, and all the agents in the system communicate via the environment agent [1].

3.4. Environment

All agents in the system exist in the environment. It is necessary for each agent to register with the environment, so that the environment is cognizant of the agent's existence. The environment should also accommodate all agent behaviors and other global behavior in the system. The primary role of the environment is to manage inter-agent communication. All agents in the system communicate via the environment. Inter-agent communication occurs as follows:

a. A message is sent to the environment by the agent. The agent provides information about gathering a signal at a specific location.

b. The environment passes this information to the appropriate agents.

The signal strength decreases as an inverse-square function. The environment is implemented as a Java thread, and it also performs periodic checks to detect collision occurrences in the system [1].

4. FORWARD AIR CONTROLLER SCENARIO

The existing functionality of the forward air controller mission is explained here. The mission includes a search area, or the Area of Interest (AOI), hunter UAVs, strike or killer UAVs, and static targets. The Area of Interest is a rectangular area where potential threats may exist. The hunter UAVs, which are deployed into the search area in pairs, scan the area and look for potential threats, or targets. Once a deployed hunter UAV discovers a target, on hunter remains in close contact with the target, while the other hunter ascends to a higher altitude and acts as an airborne radio repeater. The killer UAV orbits at a higher altitude and waits until signaled to perform a strike. The killer UAV responds to the hunter UAV's request to perform a strike. The killer UAV, once recruited for a strike, is guided to the target location with the help of the controlling UAV. After the strike is performed and the target is destroyed, the hunter UAVs continue to scan the area and conduct battle damage assessment (BDA). The hunter UAVs can recruit killer UAVs if further strikes are necessary in the area. Figure 1 shows an image depicting the forward air controller mission [4].

Below is a step-by-step indication of the hunter-killer interaction in the forward air controller's mission:

a. Two hunter UAVs are deployed to the Area of Interest [4].
b. The two deployed hunter UAVs scan and search the Area of Interest for threats or potential targets [4].
c. A hunter UAV finds a target [4].
d. The hunter UAV recruits a killer UAV and signals it to perform a strike [4].

e. The hunter UAV and killer UAV identify each other. They confirm their communication and strike capabilities [4].
f. The hunter UAV provides information regarding the type of target it has discovered to the killer UAV [4].
g. Information regarding the target's location is also provided to the killer UAV by the hunter UAV. The hunter also describes the locations of friendlies in the area [4].
h. The hunter UAV gives the killer UAV permission to conduct a strike.
i. The targets are marked [4].
j. The killer UAV conducts a strike and destroys the target [4].
k. The hunter UAVs continue to scan the area and conduct battle damage assessment (BDA) [4].
l. The communication between the hunter UAV and killer UAV is broken and they continue to perform other tasks [4].

Figure 1 shows two hunter UAVs patrolling the Area of Interest and looking for potential threats or targets. The targets are scattered at random locations across the Area of Interest. There is also a killer UAV orbiting at a higher altitude and waiting to be recruited by the hunter UAVs to conduct a strike.

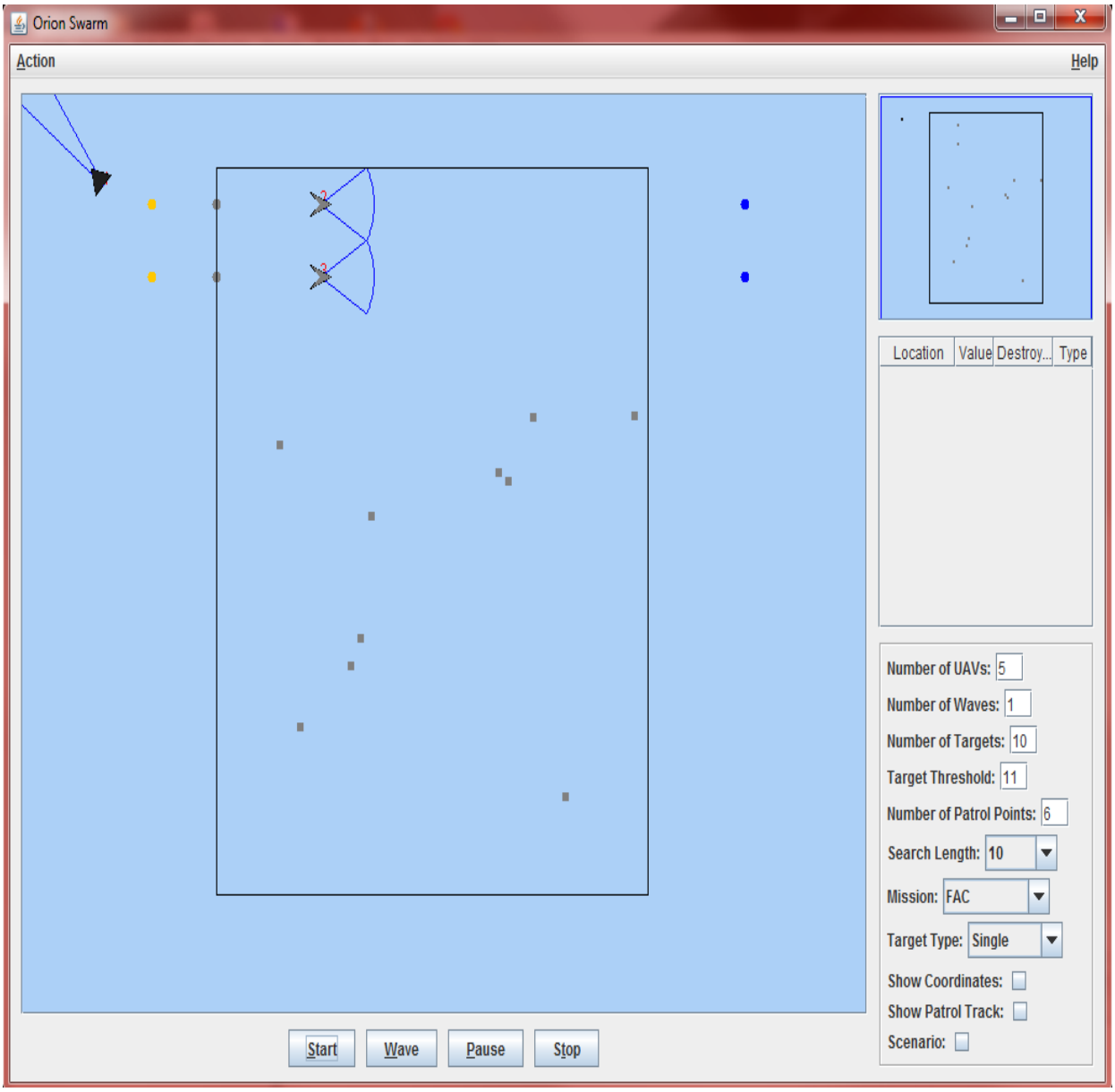

Figure 1. The forward air controller mission.

5. SIMULATION FRAMEWORK

The research contributes to previous work done to develop an agent-based framework. This framework is used to simulate unmanned air vehicles as virtual agents and is knows as Asynchronous Search and Attack System (ASAS). UAV agents are added to this framework by extending its object-oriented structure. The UAV agents are simulated as small, low-cost, dispensable UAVs. Each agent in the framework is assumed to have limited functionality in order to obtain a fairly precise model. The computational power of UAVs, as well as the memory and communication capacity is also assumed to be limited. The sensor and actuators, which are devices built into the UAVs, are also assumed to be affected by noise and failures [4].

There exists a tight sensor-actuator coupling between modules, and the coupling determines control for agents in the system. Agents are triggered with sensations in order to add behavior to them. Agents use different states and could act differently to similar sensations in different states [4].

6. HIERARCHICIAL CONTROL FOR FORWARD AIR CONTROLLER

The forward air controller mission in the Orion framework utilizes a real-time hierarchical structure to control UAVs performing hunter and killer strike missions with communication relays. Bayesian Decision Analysis is used for high-level control, and message passing and state machines are used to determine low-level control [4].

The forward air controller mission includes different types of UAVs: the hunter YAV and the killer, or strike, UAV. Each UAV has different operational capabilities. The hunter UAVs, which are deployed into the search area in pairs, scan the area and look for potential threats, or targets. Once a deployed hunter UAV discovers a target, one hunter remains in close contact with the target while the other hunter ascends to a higher altitude and acts as an airborne repeater. The killer UAV orbits at a higher altitude and waits until signaled to perform a strike. The killer UAV, once recruited for a strike, is guided to the target's location with the help of the controlling hunter UAV. After the strike is performed and the target is destroyed, the unter UAVs continue to scan the area and conduct battle damage assessment (BDA). The hunter UAVs can recruit a killer UAV if further strikes are necessary. The complexity of each individual UAV is minimized through task differentiation and specialization [4].

7. OBJECTIVES

The goal of this paper is design, develop, implement, validate, and test a simulation environment with the following characteristics.

7.1. Objective 1

To add sensor stations to the forward air controller mission. Sensor stations exhibit the following characteristics.

a. Sensor stations are added to the Area of Interest as static elements.
b. Each sensor station has a circular sensing area.
c. The sensor stations are deployed at random locations in the Area of Interest.
d. Sensor stations can detect targets that fall within its sensing area.
e. Sensor stations can signal killer UAVs and recruit them to perform a strike to destroy targets that have been detected.
f. Hunter UAVs are deployed in pairs and perform a target search. They can recruit killer UAVs to perform strikes when targets are detected.
g. Sensor stations and hunter UAVs search for targets simultaneously and recruit killer UAVs to destroy targets that have been found.
h. The number of sensor stations included in the mission is entered by the user.

7.2. Objective 2

To add pop-up targets to the forward air controller mission. Pop-up targets exhibit the following characteristics:

a. Pop-up targets are added to the simulation environment.

b. Pop-up targets are static and are initially invisible in the simulation, and, therefore, cannot be detected.
c. Pop-up targets are added to the environment randomly after the simulation has begun.
d. These targets are added to the Area of Interest at random locations.
e. Once added to the environment, they can be detected by hunter UAVs and sensor stations.
f. Killer UAVs are recruited by sensor stations and hunter UAVs to destroy these targets.
g. The number of pop-up targets added to the simulation is entered by the user.

8. THE NEW FORWARD AIR CONTROLLER MISSION – SENSOR STATIONS AND POP-UP TARGETS

While the previous forward air controller mission includes the search area, hunter UAVs, strike UAVs, and static targets, the new forward air controller mission includes sensor stations and pop-up targets in addition to the already existing elements. Because every agent or element needs to be registered with the environment, the new sensor stations and pop-up targets are registered with the environment as well. Registering with the environment is necessary so that the environment is aware of the presence of all agents in the system.

The new forward air controller mission works as follows. A pair of hunter UAVs are deployed to the search area. These hunter UAVs scan through the search area looking for threats. While the hunter UAVs patrol the search area, the killer UAV orbits at a higher altitude and waits to be recruited by a hunter UAV. Sensor stations are also deployed at random locations in the search area. These sensor stations have a circular sensing area and are capable of discovering targets. The hunter UAVs and sensor stations search for threats simultaneously, thereby reducing the overall mission time and cost. Once a hunter UAV discovers a threat, on hunter UAV remains in close contact with the target while the other hunter UAV ascends to a higher altitude and acts as an airborne radio repeater. The hunter UAV then signals the killer UAV to perform a strike. In a similar fashion, sensor stations that discover a threat also signal the killer UAV to perform a strike. Upon being recruited by a hunter UAV or a sensor station, the killer UAV moves in and performs a strike. The forward air controller mission also includes pop-up targets. They are static targets that are initially non-existent in the system and show up in the environment at a later time (pop-in) after the simulation has begun. These pop-up targets cannot be detected or discovered by hunter UAVs or sensor stations until they pop-in to the system and,

therefore, cannot be destroyed. Once these targets show up in the system, they can be detected by both sensor stations and hunter UAVs, and the killer UAVs will be able to perform a strike on these targets.

A step-by-step indication of the hunter-sensor-killer scenario is as follows:

a. Sensor stations are deployed at random locations in the Area of Interest.
b. Two hunter UAVs are also deployed as pairs in the Area of Interest.
c. The sensor stations and hunter UAVs search for targets. The sensor statins are static and can find targets within their sensing area. Hunter UAVs perform a sweep search.
d. A sensor station or hunter UAV finds a target.
e. The sensor station or hunter UAV signals the killer UAV to perform a strike.
f. The sensor station or hunter UAV provides information regarding the type of target and also provides information regarding the target's location.
g. The killer UAV receives permission from the sensor station or hunter UAV to perform a strike.
h. The targets are marked and destroyed by the killer UAVs.

Figure 2 depicts the new forward air controller mission.

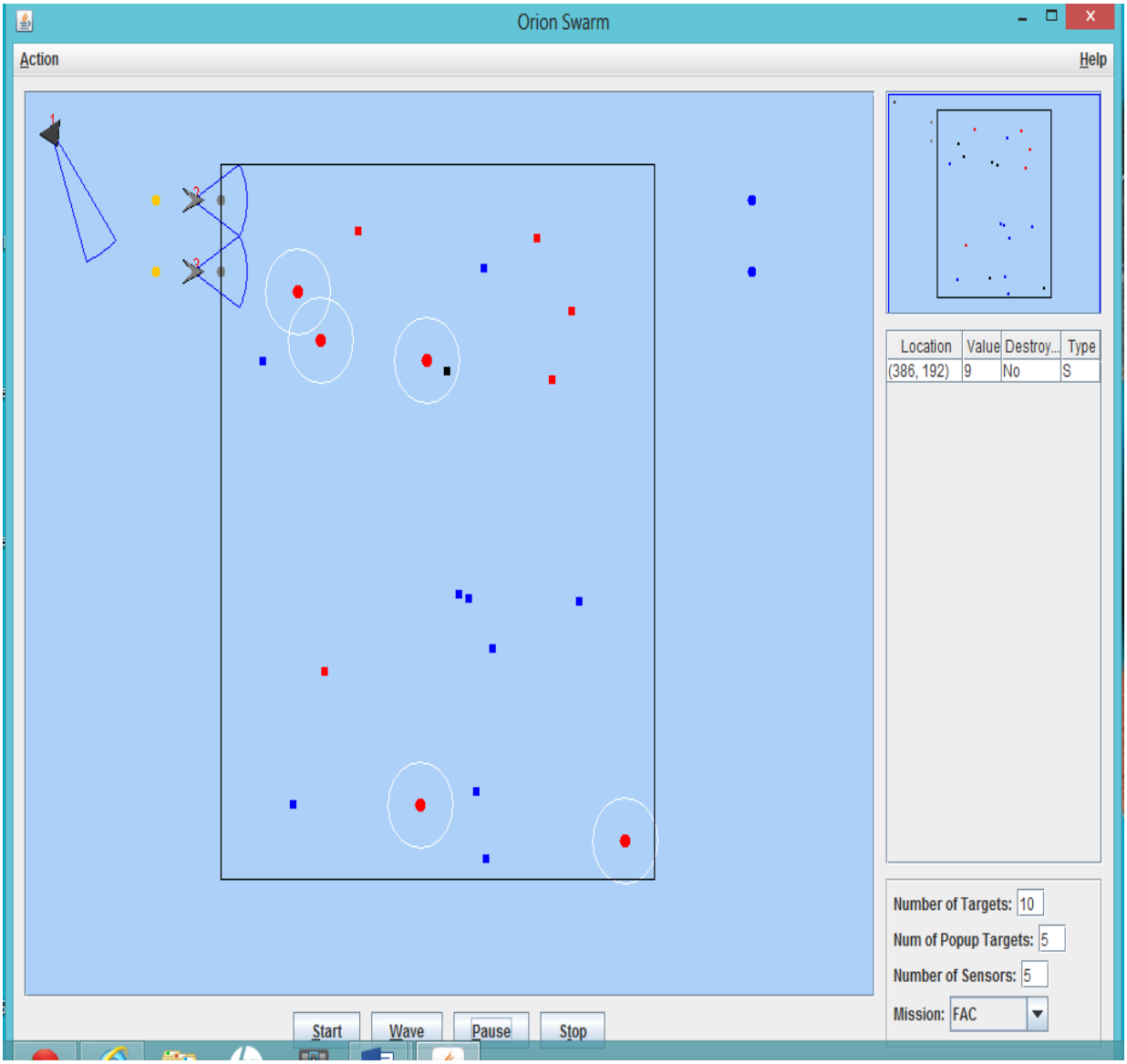

Figure 2. The forward air controller mission with sensor stations and pop-up targets.

8.1. Bayesian Decision Analysis

The forward air controller mission uses the Bayesian Decision Analysis model for high-level decision making. The sensor stations and hunter UAVs utilize the Bayesian Decision theory. The Bayesian Decision theory is applied in situations of certainty, risk, and uncertainty. Due to the complex and dynamic nature of a battlefield, decision making becomes more difficult.

The Bayesian Decision tree assists in decision making my choosing the best decision tree from a set of alternatives [4].

When a sensor station or hunter UAV detects a target, the sensor station and hunter UAV use the Bayesian Decision tree to make one of two choices. The two choices are as follows:

a. To continue gathering more information about the target or make a decision without more investigation [4].
b. To recruit the killer UAV and perform a strike, or continue to search without performing any actions [4].

The Bayesian Decision tree equation is as follows:

$$p(s_i \mid e_j, z_k) = [(p(s_i)p(z_k \mid e_j, s_i)]/[p(z_k \mid e_j)],$$

In the above equation,

s – State of nature

e – Available experiment which provides new information

z – Message that can be returned from an experiment

a – available course of action that can be taken

8.2. Sensor behaviors

The sensor station behavior is assumed to be limited in order to simplify the implementation. Sensor stations have similar behavior to unmanned air vehicles. While the sensor stations are assumed and programmed to be static, and do not perform sweep searches in the Area of Interest, they are capable of detecting threats or potential targets in the search area. Sensor stations have a circular sensing area unlike the UAVs. Therefore, any targets that fall

within the range of the sensor stations' sensing radii and are at any angle from the sensor station can be discovered. Once a sensor station discovers a threat, it communicates with the killer UAV which waits at a higher orbit until recruited. The sensor station and killer UAV go into a communication handshake to identify each other. The killer UAV then receives clearance from the sensor station to conduct a strike.

The step-by-step sensor station and killer UAV mission is described below.

a. The sensor station is deployed at a random location in the Area of Interest.
b. The sensor station is static and looks for threats, or targets, within its sensing area.
c. The sensor station discovers a target.
d. The sensor station provides information regarding the type of target it has discovered.
e. The sensor station communicates with the killer UAV that is loitering and waiting to be recruited, and provides a description of the type of target and the target location.
f. The sensor station then provides clearance to the killer UAV.
g. The killer UAV moves into the search area and conducts a strike.
h. The communication between the killer UAV and the sensor station is broken.

Figure 3 shows an image of a sensor station in the Area of Interest. The red circle represents the sensor station and the white circle outlines the sensing area for the sensor station.

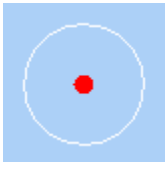

Figure 3. A sensor station.

Figure 4 is an image of a sensor station in the Area of Interest, there is a target in the station's sensing area. The target is discovered. The sensor station then signals the killer UAV to perform a strike on this target.

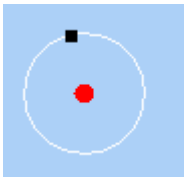

Figure 4. A threat detected by a sensor station.

Figure 5 is an image of the destroyed threat after the sensor station recruits the killer UAV to conduct a strike that has been performed.

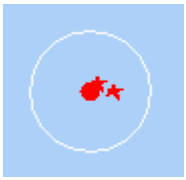

Figure 5. The destroyed threats in the sensor station's sensing area.

The image shows that two targets were discovered by the sensor station and later destroyed by the killer UAV.

8.3. Pop-up target behavior

Pop-up targets are static and are placed at random locations in the Area of Interest. The end-user can select the number of pop-up targets that need to be included in the forward air controller mission. The idea behind having pop-up targets is that a friendly in the Area of Interest could become a threat at a later time. The pop-up targets are initially invisible in the simulation and are not registered with the environment. Therefore, they are undetected by sensor stations and hunter UAVs. They cannot be destroyed because they are not detected by sensor stations and hunter UAVs and their locations are unknown. After the simulation has begun, these targets

show up, or ‘pop-up’, in the search area. Once these targets pop-up in the simulation, they can be detected by sensor stations and hunter UAVs. The sensor stations and hunter UAVs can then recruit killer UAVs to perform strikes and destroy these targets.

8.4. Registering sensor stations with the environment

The default number of sensor stations that are added to the environment is five. The end-user can change this number by entering a value for the number of sensors in the input field provided in the graphical user interface. The location of each sensor station is random. The sensor stations are deployed at random locations in the search area or the Area of Interest. The X and Y coordinates of each sensor stations are determined as follows.

```
X = (int) Math.round((searchArea.width)*Math.random()+searchArea.x);
Y = (int) Math.round((searchArea.height)*Math.random()+searchArea.y);
```

Each sensor is also assigned radio receivers and transmitters. The receivers and transmitters are necessary for the sensor stations to interact with the environment and also with other agents in the environment.

```
sensor = new CircleSensor(X, Y, value, typeOfSensor);
```

The primary radio receiver for the sensor station is set as follows.

```
RadioReceiver primaryRadioReciever = new RadioReceiver(0,360);
sensor.setPrimaryRadioReceiver(primaryRadioReceiver);
```

The primary radio transmitter for the sensor station is set as follows.

```
RadioTransmiitter primaryRadioTransmitter = new
RadioTransmitter(0,360,15*UAV.LADAR_FOOTPRINT);
```

The primary optical receiver for the sensor station is set as follows.

```
OpticalReceiver primaryOpticalReceiver = new
OpticalReceiver(0,170,1.5*UAV.LADAR_FOOTPRINT);
sensor.setPrimaryOpticalReceiver(primaryOpticalReceiver);
```

The primary LADAR receiver for the sensor station is set as follows.

```
LADARReceiver primaryLADARReceiver = new LADARReceiver(0,360);
sensor.setPrimaryLADARReceiver(primaryLADARReceiver);
```

The primary LADAR transmitter for the sensor station is set as follows.

```
LADARTransmitter primaryLADARTransmitter = new LADARTransmitter(0,360,
UAV.LADAR_STRENGTH);
sensor.setPrimaryLADARTransmitter(primaryLADARTransmitter);
```

Sensor behavior is then added to each sensor station and each one is registered with the environment as follows.

```
env.registerAgent(sensor);
```

Because the sensor stations are static throughout the mission, there are no fuel requirements for them. There is also no speed (slow, cruise, or speed) associated with any of the sensor stations.

8.5. Registering pop-up targets with the environment

The number of pop-up targets added to the mission is determined by the user. Although the default number of pop-up targets is five, this number can be changed by the end-user. The graphical user interface provides an input field where the user can enter the number of pop-up targets to include in the mission. The location of each target is random, and is determined by using the height and width of the search area. The X and Y coordinates of each target are determined as follows.

```
X = (int) Math.round((searchArea.width)*Math.random()+searchArea.x);
```

```
Y = (int) Math.round((searchArea.height)*Math.random()+searchArea.y);
```

The X and Y values evaluated from the previous statements determine the location, or position, of the pop-up target. Each target is also set with a receiver and transmitter. Targets require receivers and transmitters so that they can communicate with the environment and other agents in that environment. The primary receiver for each target is assigned as follows.

```
popupTarget = new PopUpTargets(X,Y,0,n,typeOfTarget);
LADARReceiver primaryLADARReceiver = new LADARReceiver(0,360);
popupTarget.setPrimaryLADARReceiver(primaryLADARReceiver);
```

The primary transmitter for each target is assigned as follows.

```
LADARTransmitter primaryLADARTransmiitter = new
LADARTransmitter(0,360,UAV.LADAR_STRENGTH);
primaryLADARTransmitter.setPaintable(false);
popupTarget.setPrimaryLADARTrransmitter(primaryLADARTransmitter);
```

The typeOfTarget variable determines the type of target which can be one, two, or three. In order to simplify implementation, we assume that all targets, including the pop-up targets, in the forward air controller mission are static. After each target is assigned a random location in the Area of Interest, and is also assigned transmitters and receivers, the targets are registered with the environment. Each target must be registered with the environment so that the environment is aware of each target's presence. Targets are registered with the environment as follows.

```
env.registerAgent(popupTarget);
```

8.6. Additions to the Orion Framework

There are several code changes made for the Orion Framework in order to add sensor stations and pop-up targets and to implement their functionality. Figure 6 below shows that the orion.swarm.circlesensor package has been added to the framework.

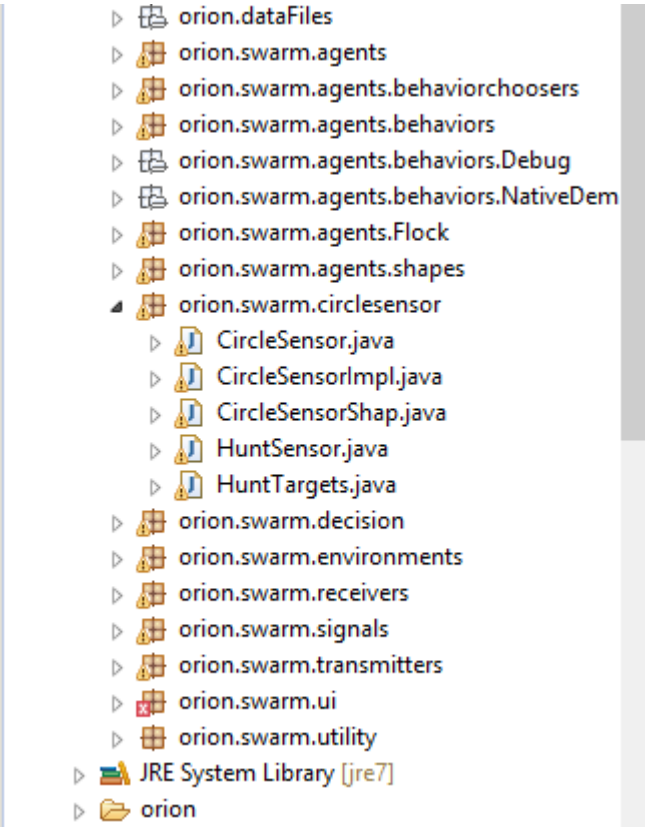

Figure 6. Added orion.swarm.circlesensor to the Orion Framework.

Figure 7 shows that PopupTargets.java and Sensor.java have been added to the orion.swarm.agents package.

Figure 8 shows that HuntSensorTargets.java has been added to the orion.swarm.behaviors module.

Figure 9 shows the additions to the orion.swarm.agents.behaviors module. AvoidCircleCollision.java, MovingTargetAvoidCircleCollision.java, SensorSearch.java,

SensorTandemSearch.java, TargetCircleIdentification.java and UAVAvoidCircleCollision.java have been added. Several other changes have been made to the framework in order to design and implement the sensor stations and pop-up targets.

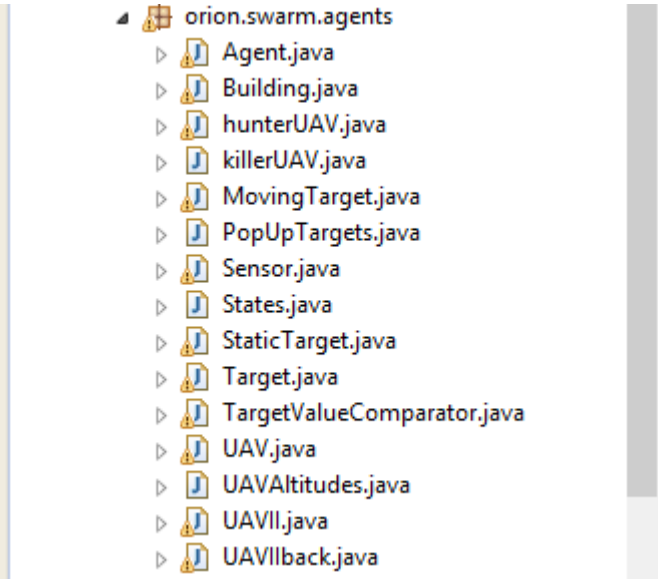

Figure 7. Additions to the orion.swarm.agents package.

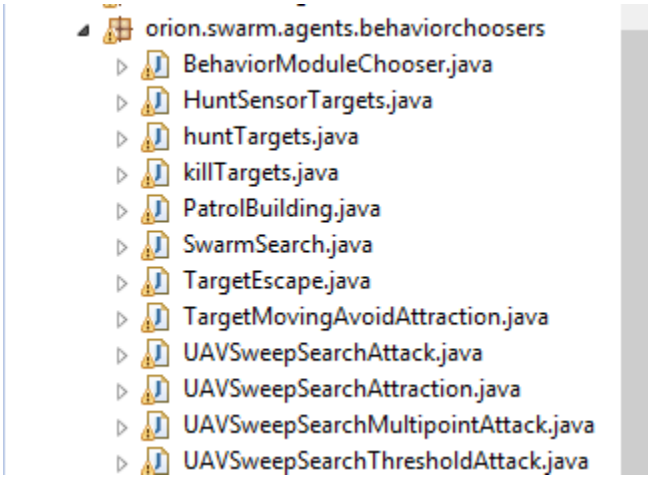

Figure 8. Additions to the orion.swarm.agents.behaviorchoosers package.

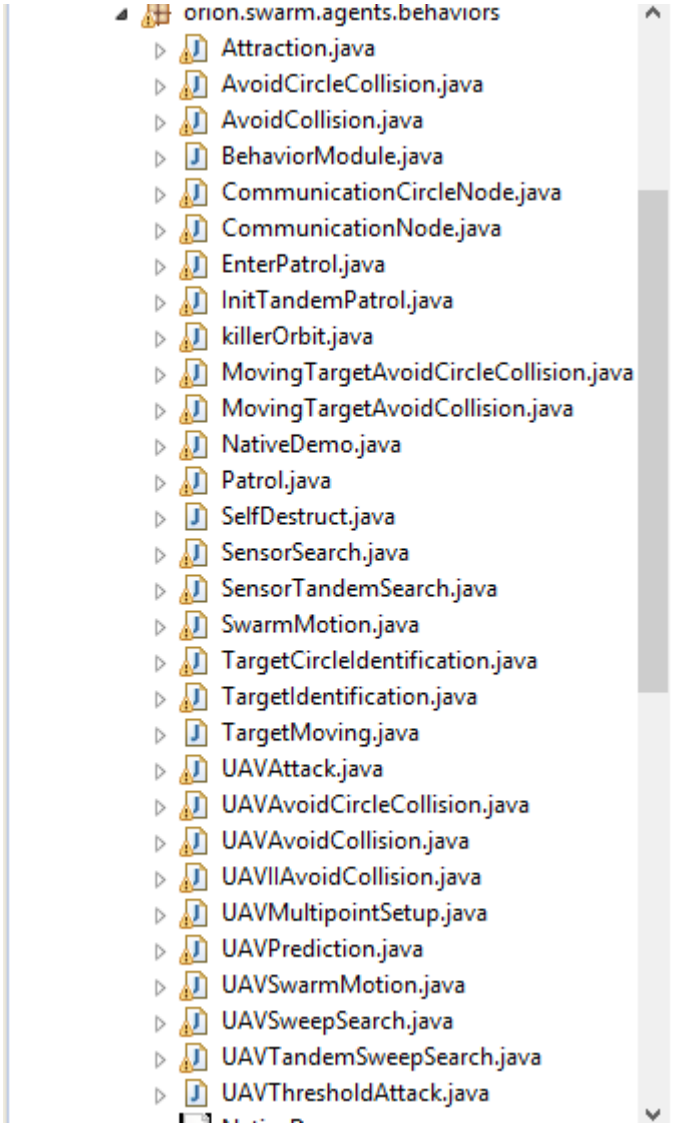

Figure 9. Additions to the orion.swarm.agents.behaviors package.

8.7. An illustration of the forward air controller mission

8.7.1. Start screen

Figure 10 shows the start screen for the forward air controller mission. As seen in the figure, the start screen has a number of fields for the user to enter information, depending on the forward air controller mission's requirements. At the bottom right of the screen, the user is allowed to enter values for the following fields.

a. The number of static targets to include in the mission.

b. The number of pop-up targets to include in the mission.

c. The number of sensor stations to include in the mission.

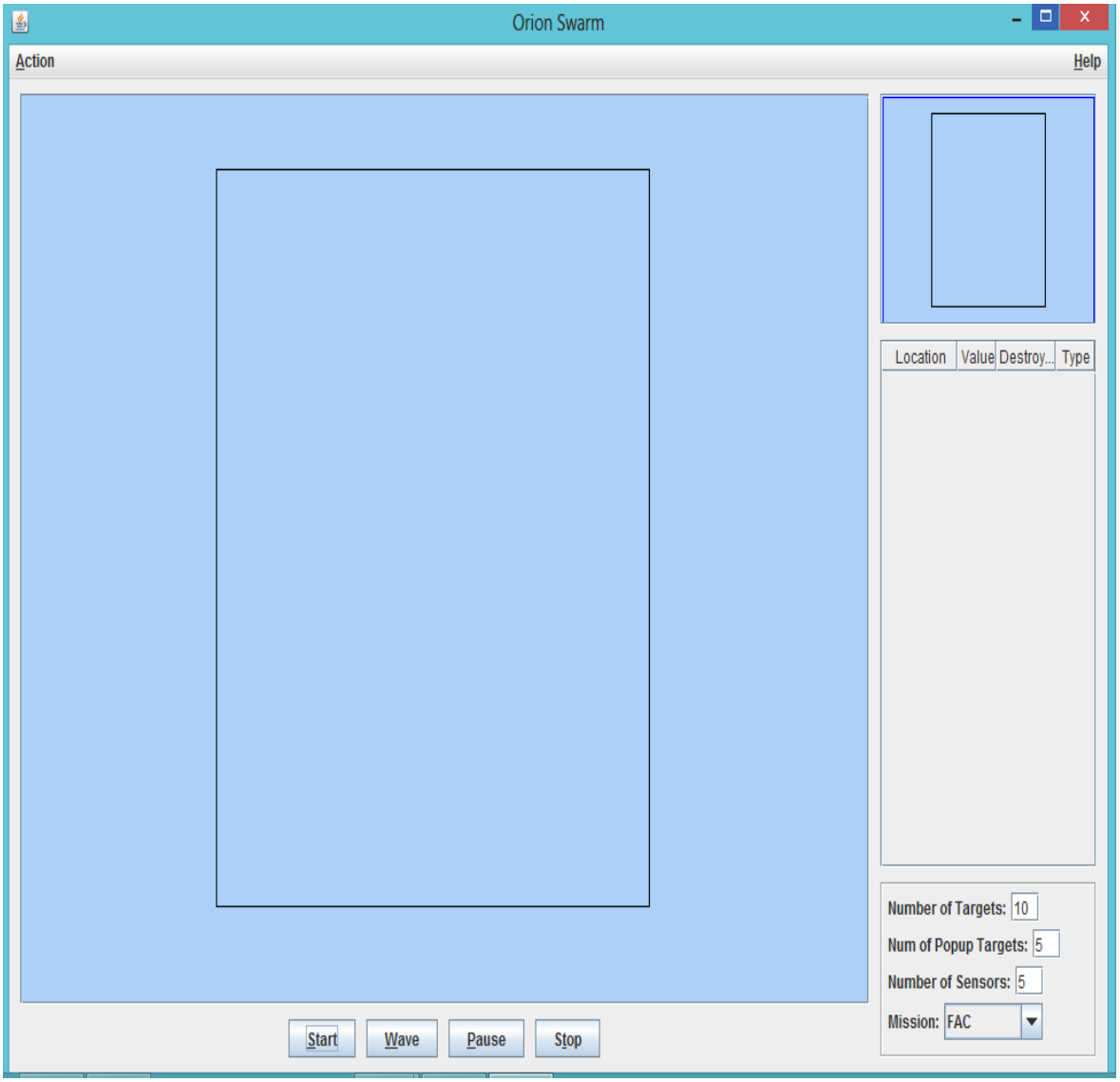

Figure 10. Start screen.

The mission field in the graphical user interface is always set to forward air controller (FAC) and cannot be changed. The start screen also shows four buttons at the bottom. They are as follows.

a. Start – to begin the mission.

b. Wave – to add a new wave of UAVs and sensor stations to the mission.

c. Pause – to pause the simulation.

d. Stop – to end the mission.

Figure 10 also shows a table on the right side of the screen. This table provides statistical information about the targets that are detected by the sensor stations and hunter UAVs. It gives information regarding the target's location, and the type of target (static or dynamic – we only use static targets for this mission). The also indicates whether the detected target is or is not destroyed.

8.7.2. Target deployment

The mission begins when the Start button is clicked. Figure 11 shows that the static targets, which are deployed in the Area of Interest at the beginning of the mission, are placed at random locations in the search area. The static targets are represented by the blue boxes in the search area. When the mission begins, two hunter UAVs are also deployed to search the Area of Interest for threats. A killer UAV is also deployed but waits at a higher orbit until it is recruited by hunter UAVs to perform a strike. The number of static targets to include in the mission is entered by the end-user in the input field provided in the graphical user interface. The default number of targets is ten.

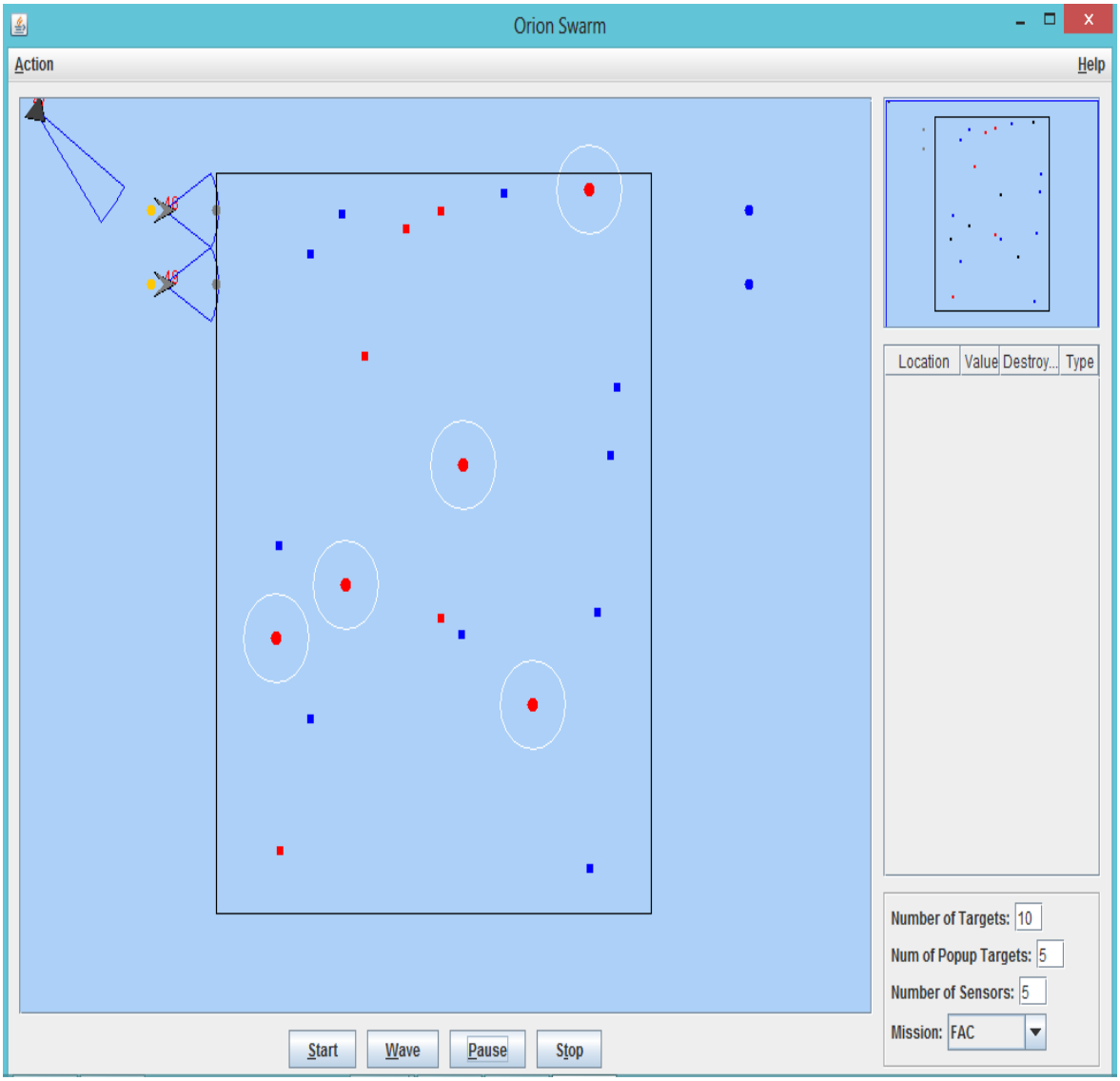

Figure 11. Target deployment in the Area of Interest.

8.7.3. Sensor station deployment

Figure 12 shows that the sensor stations are also deployed at random locations in the search area when the mission begins. The red ovals in the graphical user interface represent the sensor stations. Each sensor station has a circular sensing area around it and is represented by the white ovals. The number of sensor stations to include in the mission is entered by the end-user in

the input field provided in the graphical user interface. The default number of sensor stations is five.

Figure 12. Sensor stations deployed in the Area of Interest.

8.7.4. Pop-up target deployment

Pop-up targets are deployed at random locations in the search area. These targets are not present at the start of the simulation, and are later added to the environment at random times. These targets are represented by the red boxes in Figure 13. The number of pop-up targets

included in the mission is determined by the user. The end-user can enter the number of pop-up targets to add to the mission by entering a value in the input field provided. The default number of pop-up targets is five.

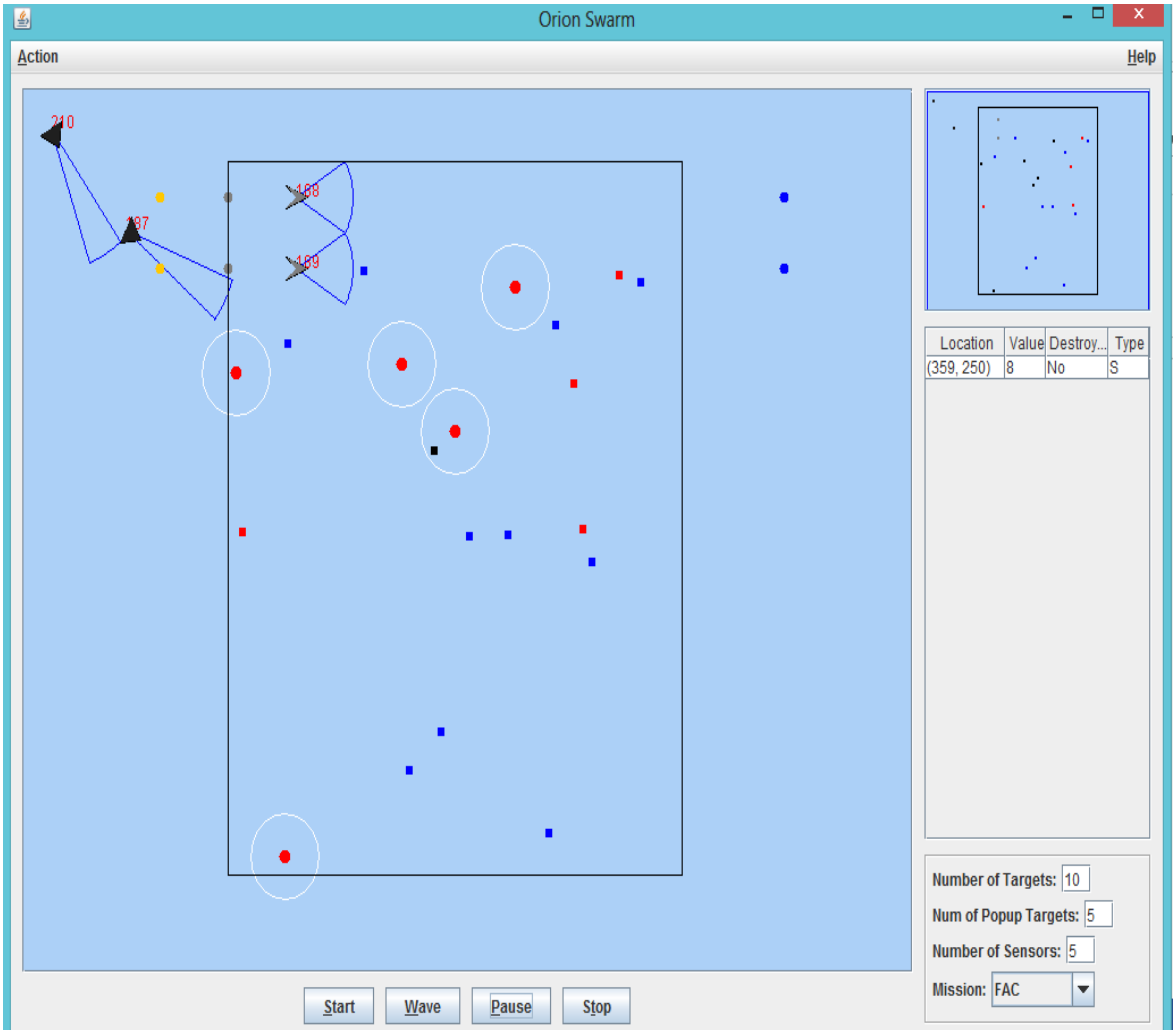

Figure 13. Pop-up targets deployed in the Area of Interest.

8.7.5. Search for targets in the mission

Figure 14 shows the hunter UAVs performing a sweep search in the Area of Interest. The hunter UAVs are deployed in pairs and search the Area of Interest. Their objective is location and find threats and to signal the killer UAV to perform a strike, destroying the targets that are

found. The sensor stations also search for and detect targets that are in their sensing areas. The sensor stations are static, but they can detect targets in their sensing area and can recruit killer UAVs to destroy these targets. Hunter UAVs and sensor stations can detect both static targets and pop-up targets.

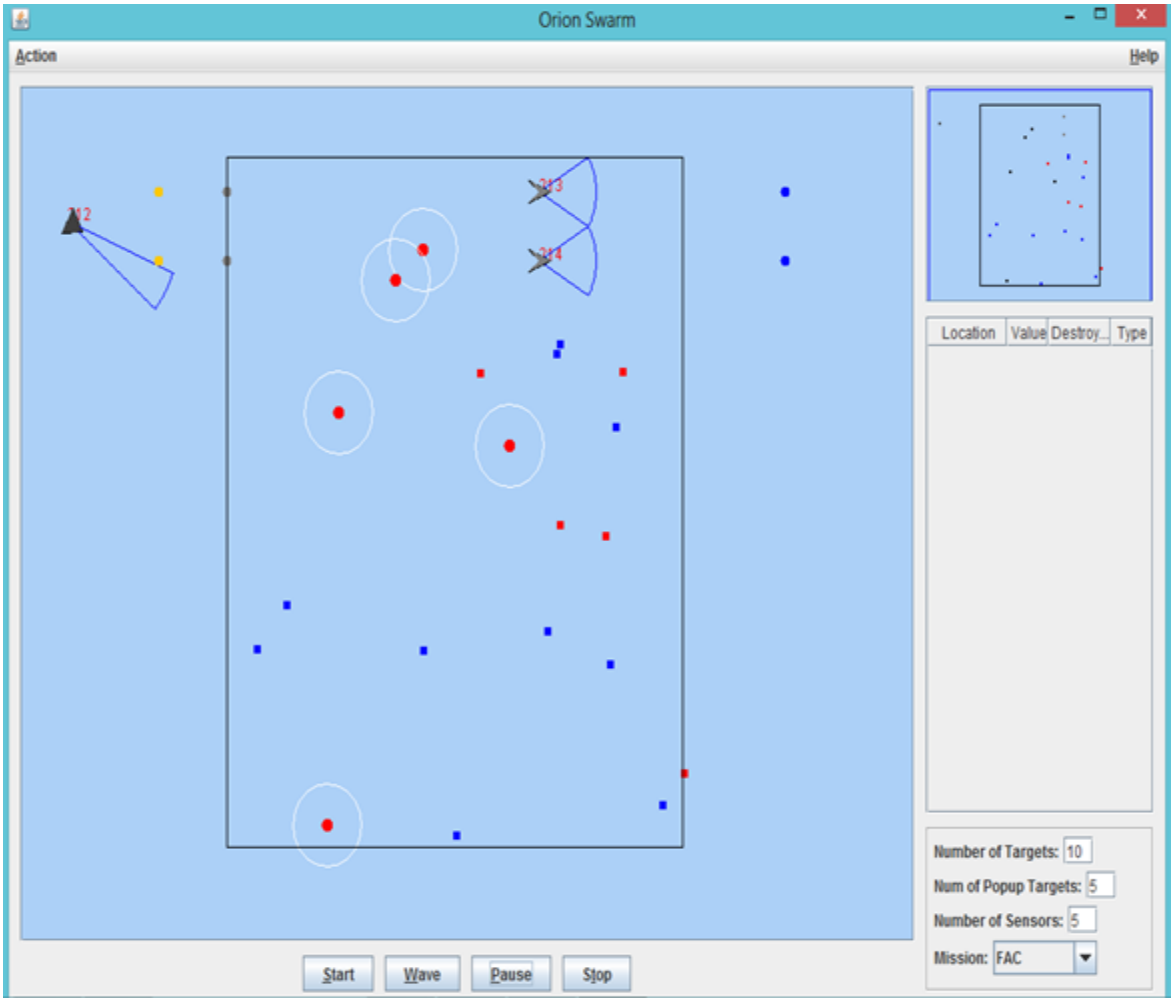

Figure 14. Hunter UAVs and sensor stations search for targets.

8.7.6. Target detection and destruction

Figure 15 shows that some targets have been detected by sensor stations and hunter UAVs. When a target is in a sensor station's sensing area, it can be detected by the corresponding sensor station. Upon detection of a target, the sensor gets the target's exact

location. The detected target changes from blue to black in the case of a static target or from red to black in case of a pop-up target. The sensor station then recruits a killer UAV to perform a strike on the detected target.

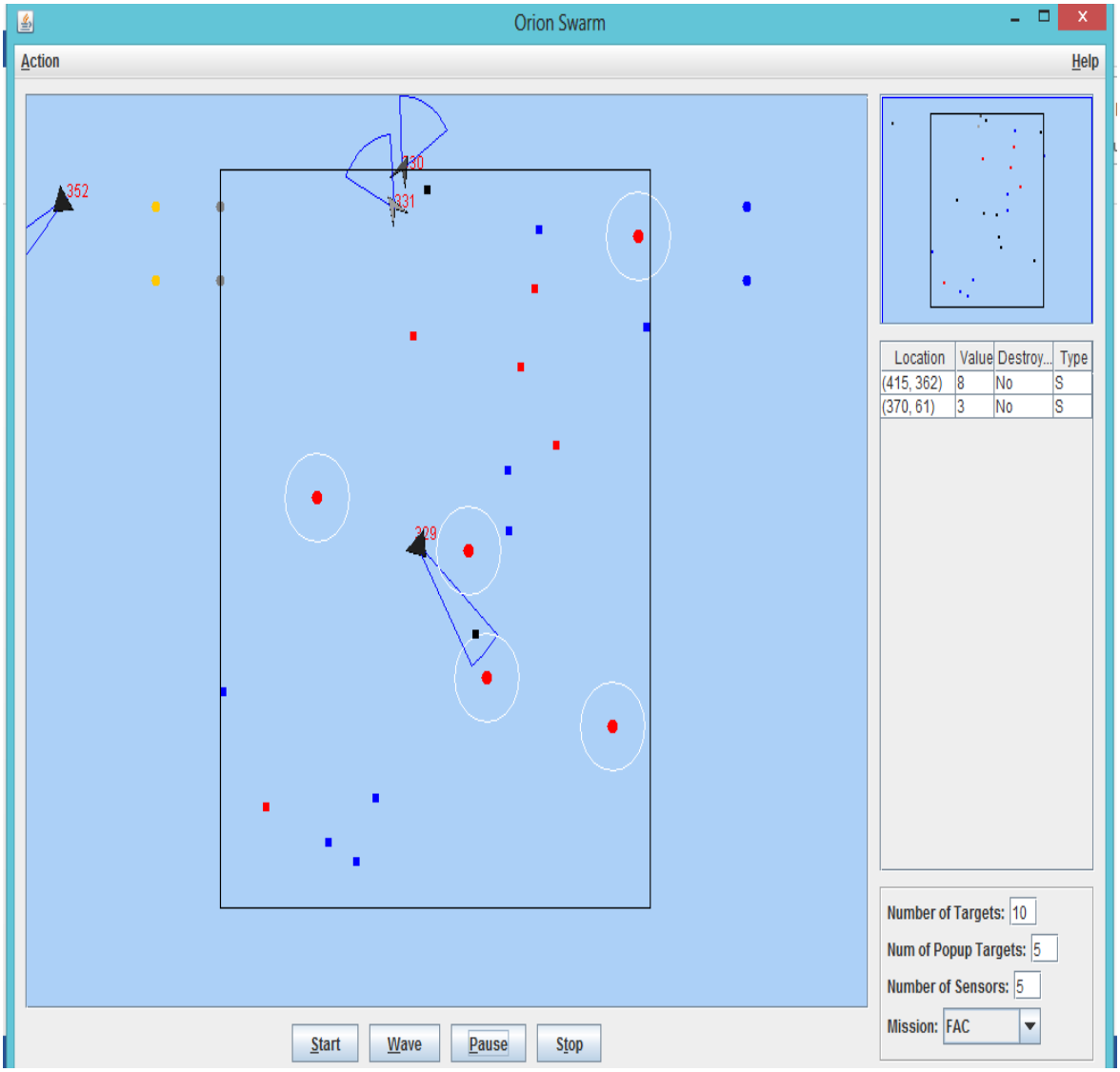

Figure 15. Targets are detected by sensor stations and hunter UAVs.

Figure 16 shows a killer UAV moving in to destroy a target that has been detected by a sensor station. The figure also shows that the hunter UAVs have detected a target. The target turns black upon detection. One hunter UAV stays in close contact with the target while the other

acts as a radio repeater. The hunter UAV in close contact with the target then signals the killer UAV to perform a strike and destroy the targets.

Figure 16 shows a killer UAV recruited by the hunter UAVs moving in to the search area to perform a strike. As can be seen in the figure, the table on the right side of the screen provides information about the targets that have been detected by sensor stations and hunter UAVs. The table provides information regarding the location coordinates for each detected target,

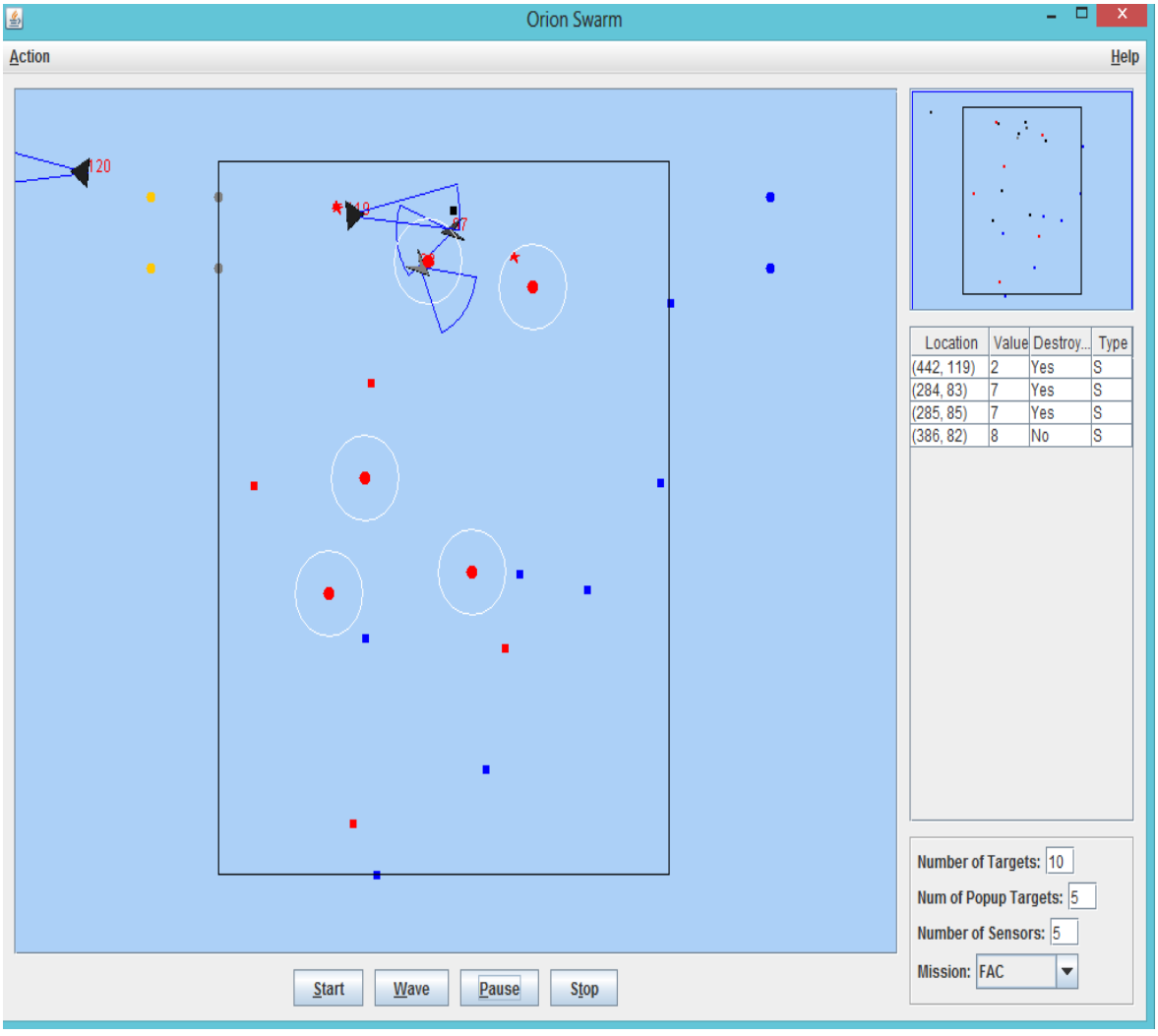

Figure 16. Killer UAV moves in to perform a strike.

information regarding its type (static or dynamic), and also provides information on whether the target has been destroyed.

Figure 17 shows targets that have been destroyed. Destroyed targets are represented by the red explosions in the graphical user interface. The number of killer UAVs required to destroy a specific target may vary depending on the type of target that is to be destroyed. As can be seen in the figure, targets that have been detected by sensor stations are destroyed, and some targets detected by the hunter UAVs are destroyed. The hunter UAVs and sensor stations continue to search for targets until the mission is complete. The mission is completed when the hunter UAVs complete searching the Area of Interest.

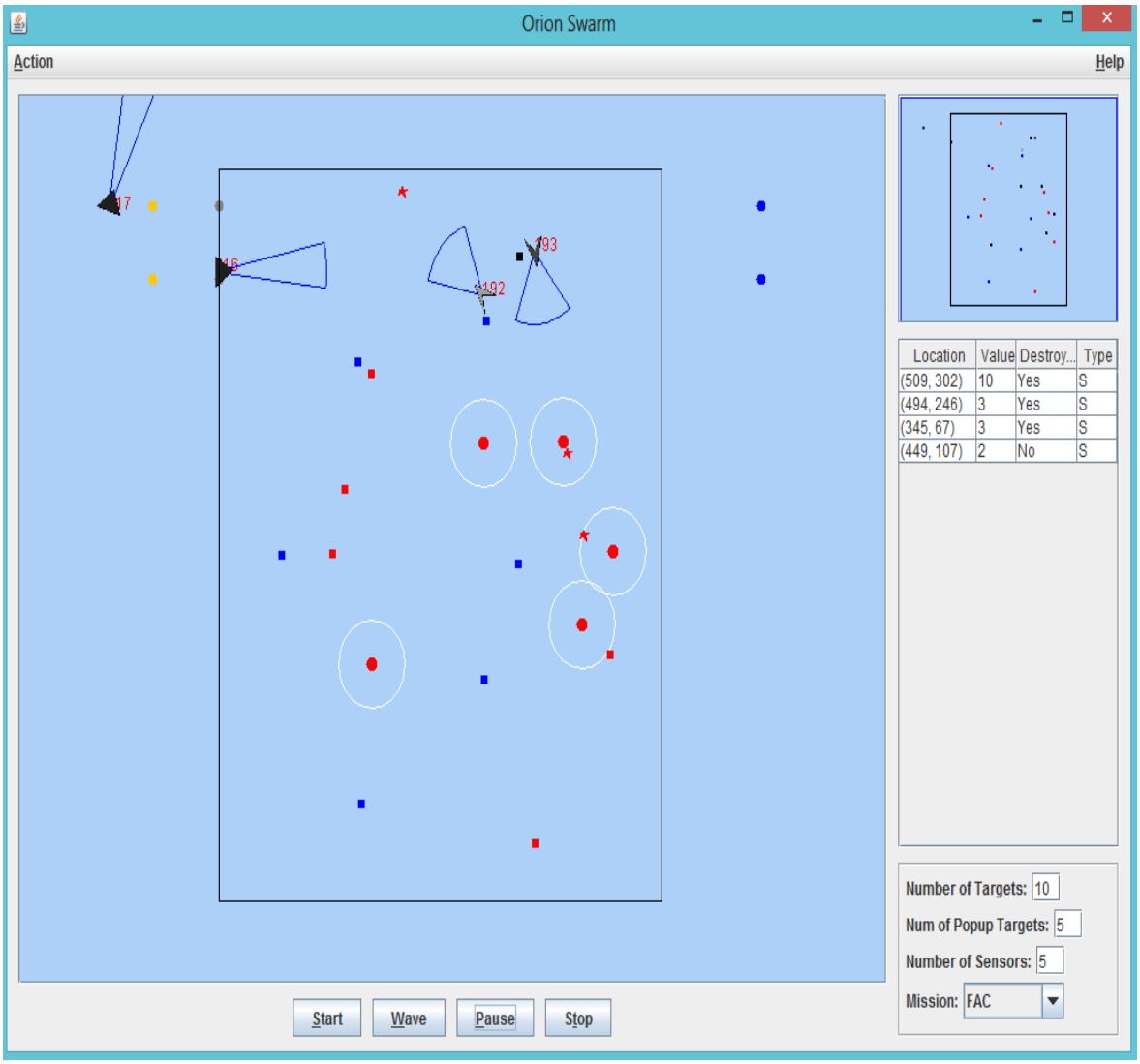

Figure 17. Detected targets are destroyed.

8.7.7. Mission completion

Figure 18 shows the completed mission. While the sensor stations are still in place in the search area, the hunter UAVs self-destruct after searching the Area of Interest. The figure shows all the targets that have been destroyed. There are also some targets, in black in the graphical user interface, that have been detected but have not been destroyed. The table on the right hand side provides information on all the targets that are detected, their locations, their types, and indicates whether they have been destroyed.

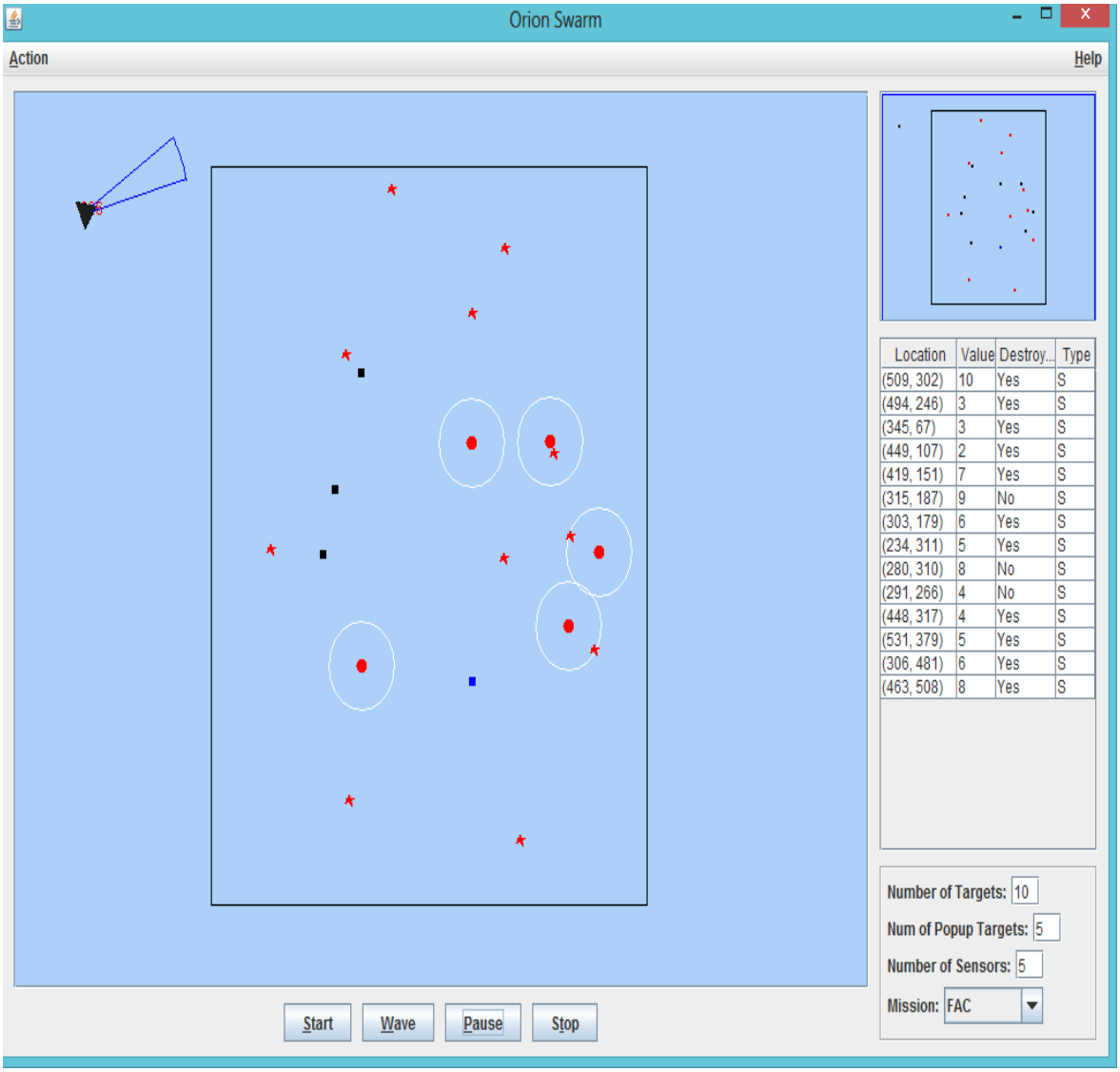

Figure 18. Mission complete.

9. TESTING AND RESULTS

The number of hunter UAVs used in all test cases is two. Other parameters such as the number of sensor stations, number of static targets, and number of pop-up targets are changed. The number of targets destroyed at the end of the mission in each test is recorded.

9.1. Test 1

Figure 19 shows the distribution of targets (static and pop-up) and the sensor stations in this test case. It shows the completed mission for the parameters given in Table 1.

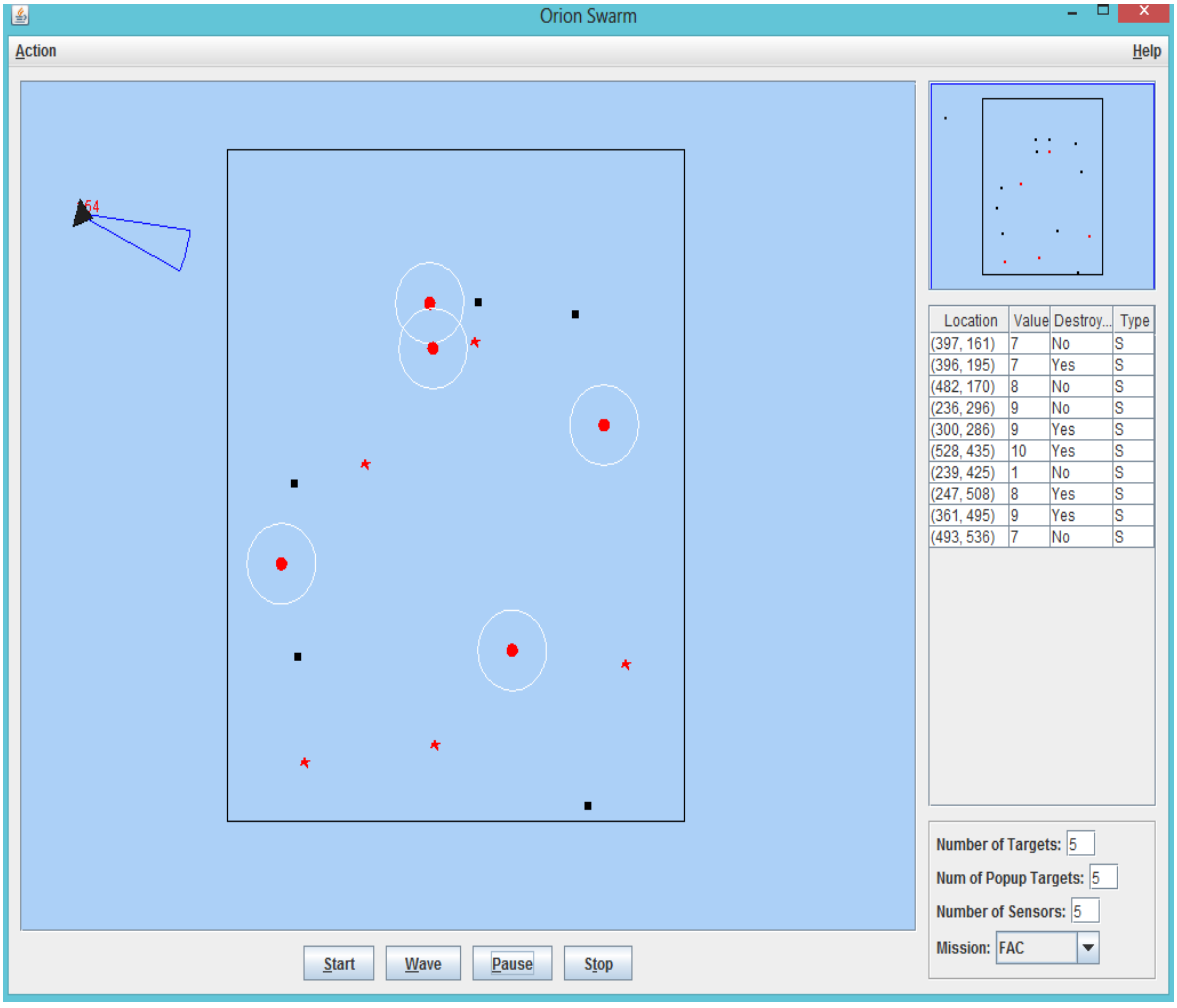

Figure 19. Testing and results – 1.

Table 1 provides the list of parameters used in this test case.

Table 1. Testing and results - 1

Number of sensor stations	5
Number of static targets	5
Number of pop-up targets	5
Number of targets destroyed	5

9.2. Test 2

Figure 20 shows the distribution of targets (static and pop-up) and the sensor stations in this test case. It shows the completed mission for the parameters given in Table 2.

Table 2. Testing and results – 2

Number of sensor stations	5
Number of static targets	10
Number of pop-up targets	5
Number of targets destroyed	9

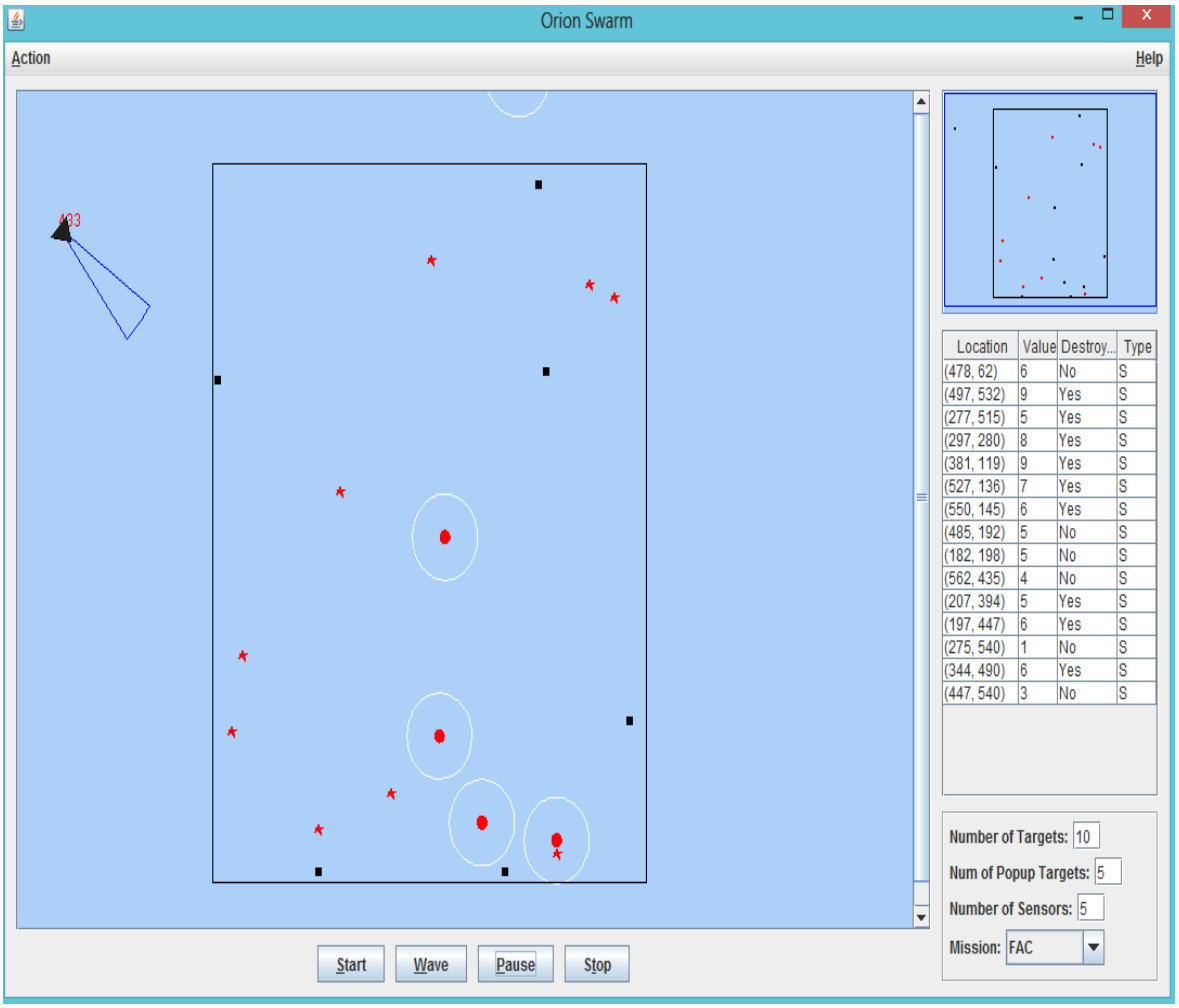

Figure 20. Testing and results – 2.

9.3. Test 3

Figure 21 shows the distribution of targets (static and pop-up) and the sensor stations in this test case. It shows the completed mission for the parameters given in Table 3.

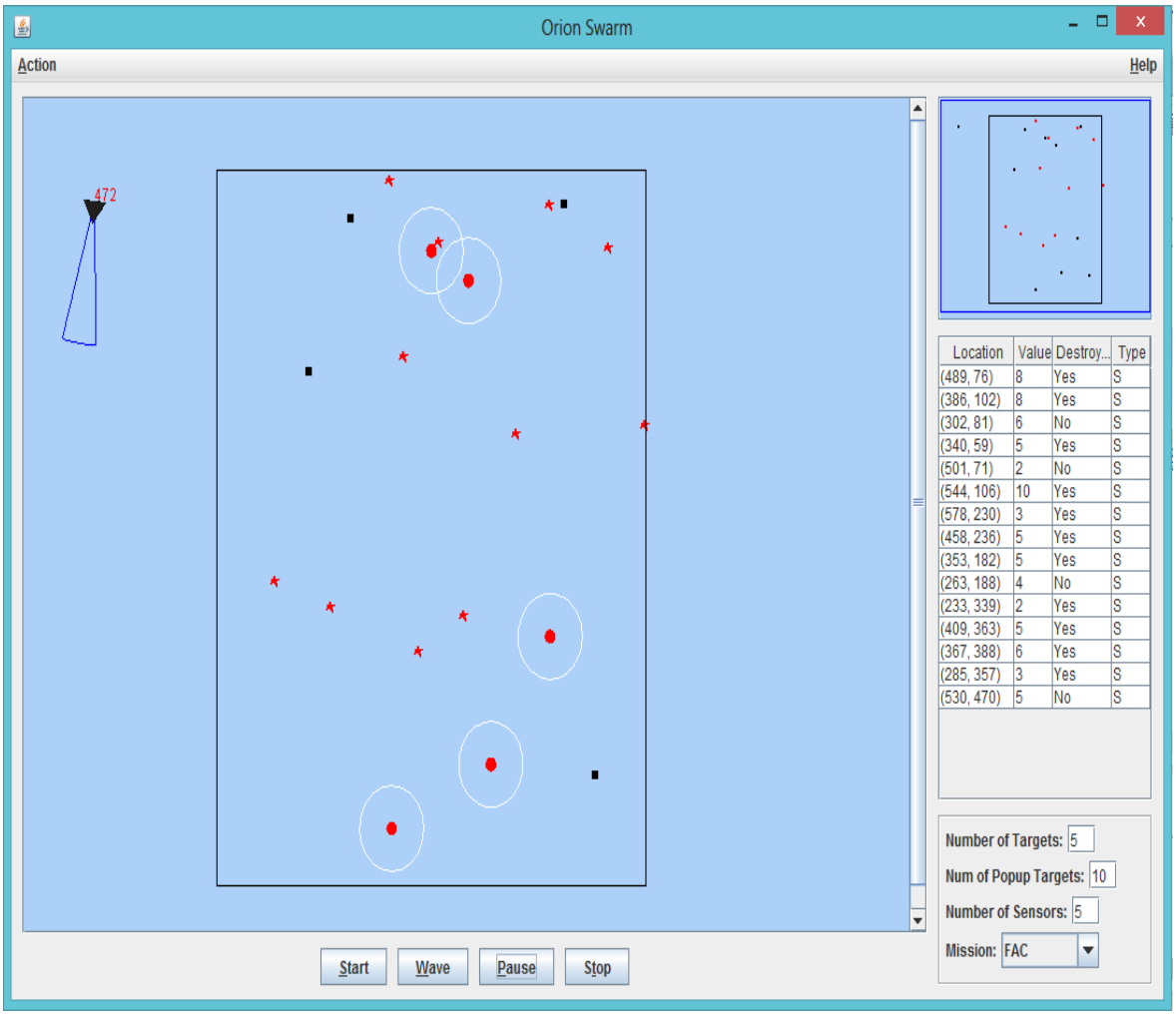

Figure 21. Testing and results – 3.

Table 3. Testing and results – 3

Number of sensor stations	5
Number of static targets	5
Number of pop-up targets	10
Number of targets destroyed	11

9.4. Test 4

Figure 22 shows the distribution of targets (static and pop-up) and the sensor stations in this test case. It shows the completed mission for the parameters given in Table 4.

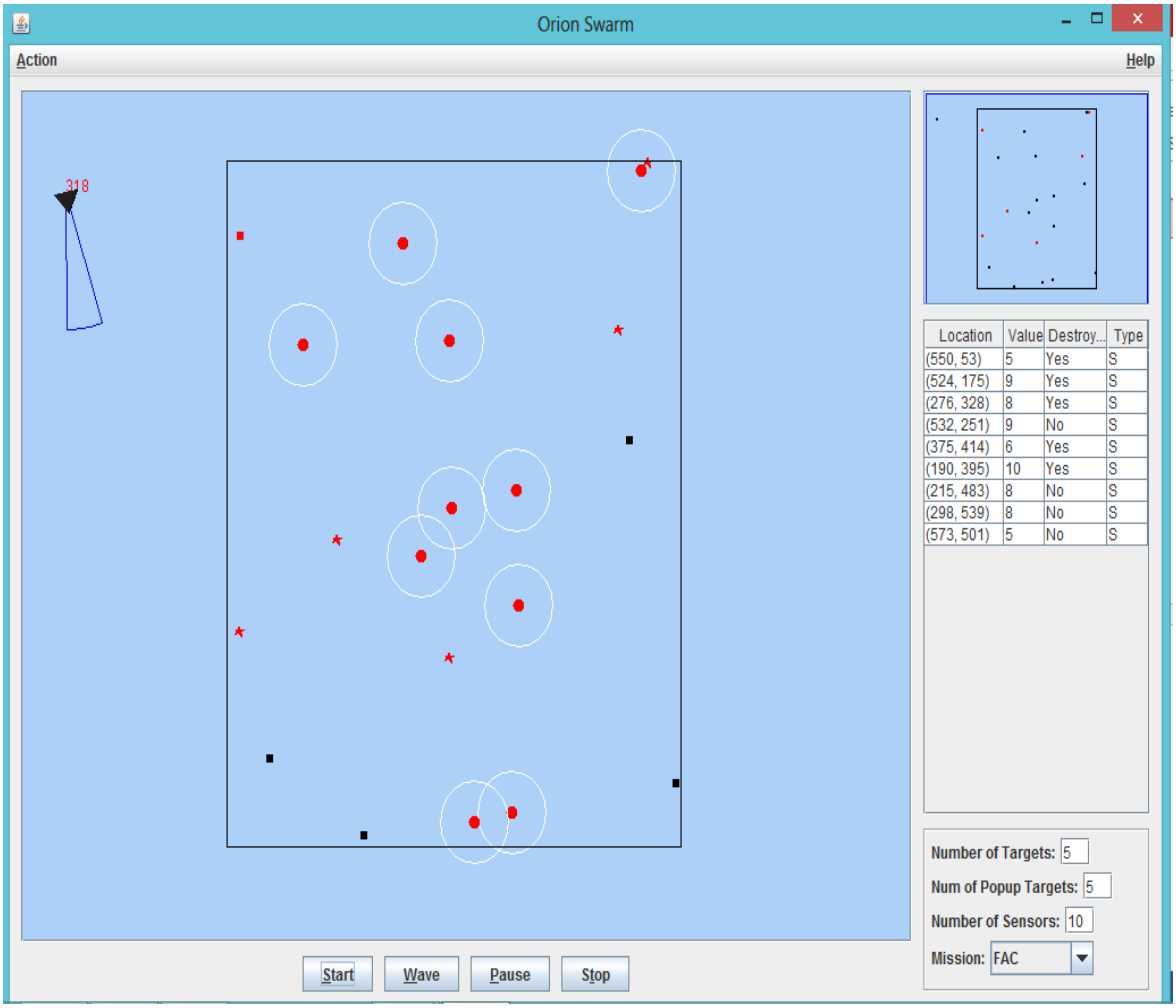

Figure 22. Testing and results – 4.

Table 3. Testing and results – 4.

Number of sensor stations	10
Number of static targets	5
Number of pop-up targets	5
Number of targets destroyed	11

9.5. Test 5

Figure 23 shows the distribution of targets (static and pop-up) and the sensor stations in this test case. It shows the completed mission for the parameters given in Table 5.

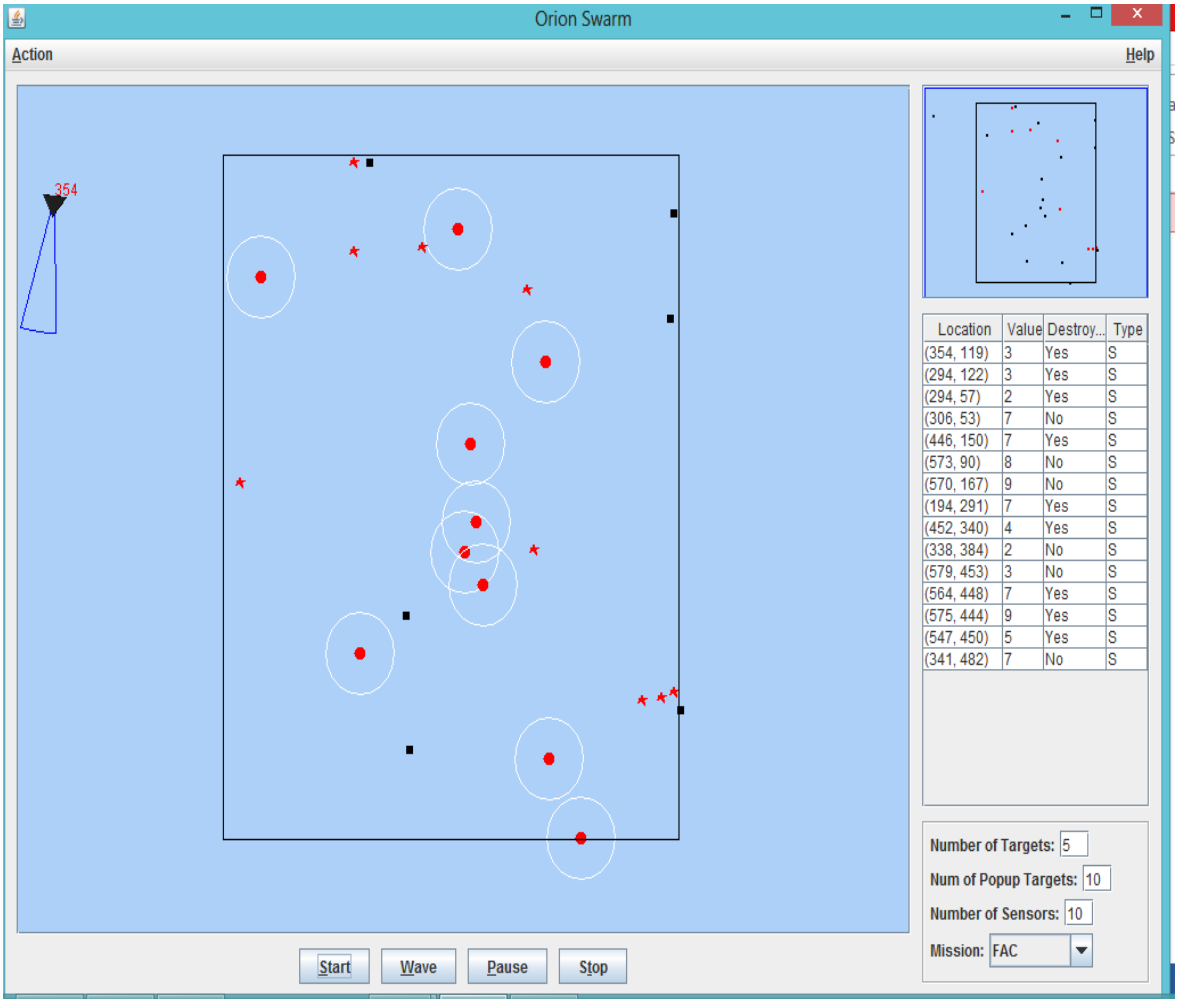

Figure 23. Testing and results – 5.

Table 5. Testing and results – 5.

Number of sensor stations	10
Number of static targets	5
Number of pop-up targets	10
Number of targets destroyed	11

10. CONCLUSION

Using the Java-based simulation framework, new agents have been added to the forward air controller mission and its functionality has been extended. The new agents have been designed, developed, implemented, validated and tested in the Orion simulation environment. The new agents include sensor stations with circular sensing radii, and pop-up targets. The sensor stations can detect a threat or target within their sensing area and can summon killer UAVs to perform a strike when a threat, or target, is found. The sensor stations use the Bayesian Decision Tree to decide between recruiting a killer UAV to strike a target immediately, and continuing to search and gather more information about the target. The hunter UAVs and sensor stations act simultaneously, thereby reducing the total cost and time of the forward air controller mission. Pop-up targets have been added to the mission, and are initially unregistered with the environment are not visible in the Area of Interest. They cannot be detected by sensor stations or hunter UAVs until they are registered with the environment at a later time. Once registered, these targets become visible (‘pop-up’) in the Area of Interest. The sensor stations and pop-up targets have different behaviors associated with them, and the run method in each agent is used to execute these behaviors. The new simulation environment has been validated and tested.

REFERENCES

[1] H. Mukhami, “Flocking behavior in purposeful UAV swarms”, Department of Computer Science, NDSU, Fargo, 2005.

[2] K. Altenburg, J. Schlecht, and K. E. Nygard, “An agent-based simulation for modeling intelligent munitions”, in Proceedings of the Second WSEAS Int. Conf. on Simulation, Modeling and Optimization, Skiathos, Greece, 2002.

[3] N. Huff, K. Ahmed, and K. E. Nygard, “An agent-based framework for modeling UAVs”, in Proceedings of the 16th International Conference of Computer Applications in Industry and Engineering, Las Vegas, NV, November, 2003.

[4] J. Tang, “Distributed control systems with incomplete and uncertain information”, Department of Computer Science Technical Report, North Dakota State University, Fargo, 2008.

[5] K. Altenburg, J. Schlecht, and K. E. Nygard, “An agent-based simulation for modeling intelligent munitions”, Athens, Greece, 2002: Advances in Communications and Software Technologies, pp.60-65.

[6] “Forward air controllers in the Vietnam War”, URL: http://en.wikipedia.org/wiki/Forward_air_controllers_in_the_Vietnam_War, retrieved on 04-23-2013.

[7] “Unmanned aerial vehicle”, URL: http://en.wikipedia.org/wiki/Unmanned_aerial_vehicle, retrieved on 04-23-2013.

[8] V. Narayanan, “A sweep search and strike air controller mission simulator”, Department of Computer Science, NDSU, Fargo, 2007.