QUANTITATIVE COMPARISON OF TWO APPROACHES

TO AGENT COOPERATION

by

JOSE RAMON ESPINO

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2005

ACKNOWLEDGEMENTS

I would like to take this opportunity to offer my sincere thanks and appreciation to all the persons who provide their undying support and fill me with a tremendous strength. First and foremost, I thank my parents, Ramon and Lucila, who continually and unconditionally support me in my professional and academic endeavors; my brother and sister, Benjamin and Irma, who are my best friends and confidantes, both of which have repeatedly listened to my frustrations about research. I would also like to thank my aunts, Maria and Angela, for their continued interest in my research and my cousin Marcos for his advice: “you should finish”.

This thesis would not have been possible without the great help of Dr. Ryes, my thesis advisor. He provided direction, advice, and motivation to continue work when I least felt like it. His suggestions and comments are invaluable. Last of all, I would like to thank Dr. McGrory for his interest and the motivation he provided.

July 23, 2005

ABSTRACT

QUANTITATIVE COMPARISON OF TWO APPROACHES

TO AGENT COOPERATION Publication No. ______

José Ramon Espino, M.S.

The University of Texas at Arlington, 2005

Supervising Professor: Alexander A. Reyes

Multi-agent systems (MASs) are characterized by collections of autonomous agents that interact with each other in simple ways, but the collection of agents as a whole is characterized by emergent behavior (EB) which will have properties that individual agents do not.

Engineers want to design artificial MASs for a variety of reasons including military operations. The idea is for the EB to be carrying out the mission itself. In this scenario, the agents are relatively inexpensive and expendable. A large body of work exists in MAS research for military applications and a variety of different designs have been proposed.

Because there is no uniform framework for communicating quantitative research results dealing with agent cooperation, it is difficult for engineers to make well-informed design decisions. We demonstrate an example of how to qualitatively and more importantly, quantitatively compare two approaches to agent cooperation. The example includes a formal experiment design and data analysis. Our example MAS deals with cooperative suppression of enemy air defense (C-SEAD). We consider two approaches to agent cooperation: state-based and Artificial Physics (AP). In this case, our example indicates that the state-based approach is better.

In the corpus of research work, ideas for mobile robot MAS cooperative control, etc. are usually "validated" by simple, ad hoc simulations. We are not convinced these simulations actually validate the ideas proposed because they lack an appreciation of system-level constraints. E.g., you cannot engineer an unmanned aerial vehicle (UAV) that has three-hour endurance yet weighs less than one pound—there is no viable source of power that can do that. If one is engineering a MAS for the real-world, we think the validation of each candidate design requires high-fidelity simulation of all important aspects of the MAS—movement, communication, resource consumption, etc. This thesis demonstrates the need for such simulations.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Table Page

LIST OF EQUATIONS

Equation Page

CHAPTER 1

INTRODUCTION

1.1 Multi-Agent Systems

A multi-agent system (MAS) [3] is a collection of autonomous agents that interact with each other in simple ways, but the collection of agents as a whole is characterized by “emergent behavior”. Emergent behavior (EB) [3] allows the MAS as a whole to perform large-scale functions, even though the individual agents are unaware of the EB. Examples of EB in animal societies include foraging in ant colonies, and schooling or flocking in fish or birds.

It is tempting to attempt to engineering a MAS that possesses the EB you want—design a family of relatively simple sensors, routers, mobile robots, etc., but design them in such a way that their interactions enable packets to cross a communication network (i.e., Internet), map a geographic region, patrol a border, search for victims trapped in collapsed buildings, etc.

1.2 Difficulty in Comparing Research Results

A number of different approaches to agent cooperation have been proposed. The problem for the engineer who wants to design an artificial MAS is that published research results are not amenable to quantitative comparison. If the engineer wants to quantitatively compare two approaches to agent cooperation, he must define a precise framework for quantitative comparison, build quantitative models within the framework of the two agent cooperation methods, design an experiment, run the experiment, and analyze the results.

This thesis research was motivated by the observation that in the corpus of research work for mobile robot MAS cooperative control, ideas are usually "validated" by simple, ad hoc simulations. We are not convinced these simulations actually validate the ideas proposed because the simulations lack an appreciation of system-level constraints. E.g., you cannot engineer a UAV that has three-hour endurance yet weights less than one pound—there is no viable source of power that can do that. If one is engineering a MAS for the real-world, we think the validation of each candidate design requires high-fidelity simulation of all important aspects of the MAS—movement, communication, resource consumption, etc. This thesis demonstrates the need for such simulations.

1.3 Quantitative Comparison of Approaches to Cooperation

This thesis provides an example of how an engineer can quantitatively compare approaches to autonomous UAV cooperation. Such quantitative comparisons help engineers select the approach to UAV cooperation that is most appropriate to the application at hand or the MAS to be engineered. Please note that the results of such comparisons are applicable only within the confines of the application at hand or the

MAS to be engineered. The thesis of the method is to design and conduct a valid and reliable experiment. The results of the experiment will quantitatively indicate which approach to autonomous UAV cooperation is best for the application at hand. In the following sections we present a brief description of the process.

1.3.1 Phase 1—Application Assessment and Specification

The experimenter must become familiar with all applicable system and subsystem requirements, architectural design decisions, allocation of required system functions to subsystems, etc. Keep in mind all the things the MAS is required to do. Therefore make a list of all the physical phenomena you may be required to model, e.g., movement with defined degrees of freedom/equations of motion, imaging (visual, IR, UV, radar, sonar), radio communication, radio direction finding, energy consumption, heat dissipation, emissions (chemical, electrical, magnetic, EM), state transition, stochastic processes, etc. You may have to develop mathematical models for these phenomena.

1.3.2 Phase –Assess what is Available to Work With

The experimenter must become familiar with approaches to autonomous UAV cooperation (i.e., research the problem). The most useful research results will be those that are formally defined using standard mathematical notations and backed up by performance curves. Note the fundamental mathematical framework used to describe each approach. Approaches that operate in continuous time will usually be described via differential equations. Approaches that operate in discrete events will usually be described via Queuing Theory, Markov Models, or some other system of difference equations. The least useful research results will be those that are informally documented with lots of textual hand-waving, imprecise box-and-arrow diagrams, and reference to simulation programs to which you have no access. It would be judicious to carefully consider approaches that will require you to invent a lot of details to fill in the gaps between the published description and your application at hand. Also note that as soon as you discover that applying a particular approach to autonomous UAV cooperation will impact the requirements you have been given (e.g., the particular approach requires radio direction finding, but your MAS has no requirements for such), you must inform project management of the risk.

1.3.3 Phase 3—Factorial Experiment Design

The basic method is to design and conduct a valid and reliable experiment. We suggest a factorial experiment design, because the results of the comparison will be the same no matter how many time you run the experiment (i.e., there are no effects of learning).

Designing a factorial experiment requires you to define a parameterized model of your MAS and that you chose specific values to test for each parameter. Each parameter is a “factor” and each value to be tested for each factor is a “level”. Every combination of factors and levels represents a candidate MAS design. The goodness of each candidate design will have to be reliably, unambiguously, and quantitatively evaluated, as we will discuss later when we talk about the “utility function”.

One of the most important factors in your parameterized/“factorized” MAS model is the approach to autonomous UAV cooperation. Each level of this factor corresponds to one of the approaches to autonomous UAV cooperation you decided to evaluate.

Another example factor will be the mission or scenario in which the candidate MAS design is to be evaluated. You have to be a little careful to distinguish the factors that define a candidate MAS design (e.g., approach to autonomous UAV cooperation) from the factors that help us understand the utility of each candidate MAS design (e.g., factors related to missions or scenarios).

The next thing you have to specify is a "utility function" (a "figure of merit" or "fitness function"). The purpose of the utility function is to reliably and unambiguously assign a numeric value of "goodness" to each candidate MAS design. Note that if your evaluation missions or scenarios are disparate, you may have to define a utility function that is appropriate for each mission or scenario.

Next you have to choose a statistical tool that is appropriate for the experiment design. Each combination of factors and levels give rise to an experiment. The utility function will assign a number to the outcome of each experiment. The purpose of the statistical tool is to do the right thing with all these numeric experiment outcomes. You have to choose the right statistical tool that will group the experiment outcomes the correct way. The most important groupings are the ones that are distinguished by the factor "approach to autonomous UAV cooperation". You have to choose a statistical tool that will hopefully show that the differences in corresponding means can be attributed to the approach to autonomous UAV cooperation and not to chance or some potentially conflating detail.

One useful way of identifying potentially conflating details is to make a reasonable effort to bring “everything” within the same mathematical framework. This may not be possible given the resources available to you (e.g., not enough time). By “everything”, I mean the essential physical phenomena manifested by the MAS and the essential phenomena required by the approaches to autonomous UAV cooperation that are of interest to you. For example, see if you can adequately represent the essential phenomena of your MAS and each approach to autonomous UAV cooperation entirely via differential equations. You will most likely discover that you need a different mathematical framework (e.g., a system of differential equations versus a system of difference equations) to represent your MAS for each approach to autonomous UAV cooperation.

The next step is to implement your experiment. This is probably the most time-consuming part of the entire process. It usually requires you to implement your factorized MAS model using a number of simulation or analysis tools. If you have time on your hands, you can verify that each system of equations is validated by its corresponding implementation.

The final step is to apply your chosen statistical tool on the numeric outcome of each experiment and present the results.

1.3.4 Thesis Organization

This thesis is organized as follows: chapter 2 presents a comparison of related works that are relevant to the thesis. In chapter 3 we present the experiment design and in chapter 4 an analysis of the experiment's data. Finally, chapter 5 presents our conclusions.

CHAPTER 2

RELATED WORKS

In chapter 2 we present research that is relevant to the thesis. In section 2.1 we provide a discussion of each of the approaches to cooperation while in section 2.2 we present a qualitative comparison of related works. Section 2.3 presents the different qualities to consider in MAS design and we conclude chapter 2 with a discussion of the difficulty in executing a quantitative comparison of related works.

2.1 Relevant Approaches to Agent Cooperation

The body of work is large and varied and we divide it into six approaches for agent cooperation which are reactive, state-based controls; Artificial Physics; indirect communication through the environment (i.e., "stigmergy"); systems with a communication manager; neighborhoods; functions of attraction and repulsion. We limit our consideration to related work to those works that have been demonstrated or applied to problems involving UAVs (unmanned aerial vehicles).

2.1.1 Reactive, State-Based Control

Reactive, state-based control uses a FSM with states and transitions to govern the behavior of agents in the system. The state represents the current role or behavior of the agent and transitions from state are caused by receiving an event. The event triggers the agent to transition to another state. Actions may be associated with either the state or the transition, but independent of the model type of FSM that is used (Moore or Mealy) the system is deterministic. Two of the works ([11] and [5]) discussed in section 2.3 adopt a reactive, state-based control approach to role coordination and dynamic classification and behavioral states, respectively.

2.1.2 Artificial Physics

Artificial physics or "physicomimetics" is a concept introduced by [16] where the agents in the MAS as well as targets are represented by idealized particles that posses charge. "Agents sense and react to virtual forces" according to [16] which are physics-like. The interaction between the charges are governed by Newtonian-based physics and according to the researchers in [16] the "system acts as a molecular dynamics ($\vec{F} = m\vec{a}$) simulation." Artificial physics also includes a frictional force that is used for self-stabilization and mass for the agents so that momentum can be modeled. The scientist or engineer who wants to apply the AP approach has the ability to edit the static force function. They can do this in such a way as to make the agents/particles arrange themselves into regular, lattice-like formations.

2.1.3 Indirect Communication

Other approaches for agent cooperation exist but eschew direct communication and are varied in their approach. The use of pheromones (both attractive and repulsive) has been modeled in [10] and [5] and widely explored. In pheromone-based MASs, agents consider the environment a grid which may be two or three dimensions. As agents move throughout the environment they deposit pheromones that other agents interpret. The pheromones can aggregate, saturate, and dissipate.

2.1.4 Communication Manager

We also reviewed the work found in [1] which provides a hierarchical, communication manager approach for agent cooperation where agents are divided in groups with distinct responsibilities. The experiment contains a centralized path planner and trajectory generator that dictates where each agent will navigate and also computes the paths that agents will take. Each group of agents has a leader that is responsible for managing a group of formation controllers. Formation controllers in turn manage groups of agents.

2.1.5 Neighborhood

A particle-based system described in [14] proposes a novel approach based on particle systems where each particle has been "replaced by an entire geometrical object consisting of a full local coordinate system." In this system each particle not only has state, but the geometric representation gives rise to orientation. There is no centralized control for the motion of the particles instead each particle has a computational process that is responsible for computing the trajectory of the particle. Additionally, it is capable of processing local perception of the environment. It is through the interaction "of the relatively simple behaviors" of each of the particles that the aggregate motions results.

2.1.6 Functions of Attraction and Repulsion

Another approach to agent cooperation that is similar to Artificial Physics is adopted in [4]. The similarity stems from the use of equations to control the agents, but the set of attraction and repulsion functions is not based on a molecular or Newtonian system. Instead it is defined directly by [4] based on previous experimental work and does not contain the concepts of friction and momentum. The approach in [4] assumes "synchronous motion and no time delays."

2.2 Qualitative Comparison of Related Work

The research carried out in the UAV field is large and varied. The following section presents a qualitative comparison of the works that are related to the research problem of this thesis. In particular we examine related work that focuses on stated-based model where a finite state machine is used as a controller and equation-based systems for UAV control. We also review the different media for agent cooperation and emergent behavior that may arise. Where applicable we also review the fidelity in radio, sensor, and motion, the mission the UAVs engage in, and the physical geography and territory that the mission takes place in. The geography and territory also lead us to review the bounded nature of the space and the units of measure in the space, or lack thereof. A summary of the issues considered in the qualitative comparison is found in Appendix B.

2.3 Qualities to Consider in MAS Design

We summarize the qualities of each related work in a table where each related work is a row in the table. The columns of the table represent the qualities that we considered in the research. The columns include whether a state machine is used for control; the state transition triggers; equation-based methods of control; the medium for agent cooperation; emergent MAS behaviors; the fidelity of the sensors; communication fidelity; motion; geographic considerations for the system; and finally the units of measure. For each consideration, we present a definition and clarification.

2.3.1 Finite State Machine Controller

The classic definition of a finite state machine (FSM) is best summarized in [9] as "[a] model of computation consisting of a set of states, a start state, an input alphabet, and a transition function that maps input symbols and current states to a next state. Computation begins in the start state with an input string. It changes to new states depending on the transition function." The work of [5] uses the Moore machine model while [11] adopts a Mealy machine model. In both cases, the FSM is deterministic.

2.3.2 FSM State Transition Trigger

Regardless of the type of FSM used as a controller, a FSM must transition to do something useful. A state transition trigger is responsible for causing the machine to move from one state to the next. Signals from agents are used as triggers in [5] while [12] adopts a more complex approach and uses a combination of "current context, constraints, and preferences" for a state transition trigger. In such a scenario a controller or supervisor monitors the current state of the ongoing simulation and can make a probabilistic determination that instructs UAVs to change their state so that the mission objectives are sufficiently satisfied.

2.3.3 Real Variable, Equation-Based Control

Another interesting method of UAV control is real variable, equation-based control. This type of control is based on a set of equations where the state variables of the system change continuously over time (e.g. as water flow over a dam). The equations may be modeled several ways: physics-based forces [16]; a combination of force, acceptance, and path planning equations [1]; set of equations for particle systems [14]; functions of attraction and repulsion [4]. We discuss each of these equation-based systems in section 2.2.4.

2.3.4 Medium for Agent Cooperation

One major goal of multi-agent systems is cooperative interaction towards accomplishing a task or missions. Different methods for cooperation exist and are catalogued in the literature. We present several of the media proposed.

The concept of cooperation via pheromones is taken directly from the natural observation of insects. According to the Merriam-Webster Dictionary, a pheromone is "a chemical substance that is produced by an animal and serves especially as a stimulus to other individuals of the same species for one or more behavioral responses". In an artificial MAS, pheromones could be implemented by simple, battery-powered, disposable radio beacons dropped by agents. Pheromones are emitted by the agents, have different flavors that cause the agents to react differently. The pheromones may be attractive [5] or repulsive. They may also exhibit behavior such as aggregation, dissipation, and saturation. One paper that uses a combination of repulsion, aggregation, and dissipation is [11].

Another approach that has been proposed is artificial forces which are described by [16] as "…artificial (or virtual) because although we are motivated by natural physical forces, we are not restricted to them. Although the forces are virtual, agents act as if they were real." In this framework, agents contain sensors that compute the force that the agent should react to based on the presents of other agents.

Another method of interaction of agents relies on indirect communication between the agents via a centralized, communication manager. Agents have their own managers for trajectory planning, target management, and interception, but synchronization activity takes place through the communication manager. This is the approach adopted by [1] where "the primary role of the communication manager is for synchronization." A very different, but effective approach is found in [14] were interaction occurs via a neighborhood which is a "spherical zone of sensitivity" who's center lies around the body of a "boid." An inverse exponential function of distance governs how much sensitivity a boid feels.

One of our two models uses simulated radio packets as medium for agent communication. Any agent action that is visible to the outside world corresponds with the transmission of packets. Agents have a limited range of reception and their interaction varies with radio proximity, noise, interference, etc.

Finally, [4] proposes the uses of functions of attraction and repulsion. The authors "assume synchronous motion and no time delays, i.e., all the members move simultaneously and know the exact position of all the other members…motion dynamics evolve in continuous time." The motion of the agent is the sum of all of the forces, attractive or repulsive, that are due to all the other agents.

2.3.5 Emergent MAS Behaviors

Emergent behavior is characterized by [10] as "system behavior that is more complex than the behavior of the individual components" and some of the behavior is not necessarily a result of the design. Several of the works presented in this chapter exhibit emergent behavior such as cooperative search, formations, cooperative attack, flocks, and aggregation and stabilization. During cooperative search, a group of UAVs cover an area which may be of fixed or unfixed size. Another type of emergent behavior is formations such as cubes or hexagonal lattices. Cooperative attack is an emergent behavior where two or more UAVs synchronize their attack in hopes of that the attack will be more likely to succeed.

Flocking is an emergent behavior that is based on the behavior exhibited by birds and is characterized by a dense area of interaction where aggregate action occurs [14]. The last type of emergent behavior presented is aggregation, the formation of a swarm, which may or may not be cohesive. Of special interest is the idea that aggregation leading up to swarming is the result of the interplay between attractive and repulsive forces. Appendix B lists the applicable emergent behaviors for the reviewed works.

2.3.6 Sensor Fidelity

Sensor fidelity refers to the accuracy of the sensors' details used in the experiment and ranges in accuracy from the idealized to high fidelity. An idealized sensor is implemented simply as a function within the works found in [11], [16], [1], and [14]. A different approach with medium sensor fidelity is that of [5] where the sensors are modeled as receptor cones controlled by a stochastic function that for locating targets based on "distance, elevation, and the amount of time spent flying over a given terrain cell." At the far end of the spectrum is high fidelity where their may be multiple types of sensors each with adaptive models, different models for communication including protocols and links. Our OPNET model adopts a high level of sensor fidelity.

2.3.7 Communication Fidelity

Much like sensor fidelity, communication fidelity refers to the accuracy of the communication details and ranges in accuracy from the idealized to high fidelity. Most of the works ([11], [5], and [1]) rely on idealized communication. Some works ([16], [14]) do not consider communication at all. The only work with high communication fidelity is our OPNET model. High communication fidelity explicitly models: packet transmission (definition of receive groups, transmission delay, propagation delay, noise, and error), antenna models, data rates, packet formats, bandwidth, frequency, and power.

2.3.8 Motion Fidelity

Every work we have reviewed is concerned with the motion of the UAVs which presents the issue of how explicitly motion is modeled, that is motion fidelity. There are six areas of motion that we examined in particular. The definitions are taken from [8].

- Pitch: The "up and down movement of the nose of the aircraft" which in this case a UAV.
- Yaw: The "side to side movement of the nose of an aircraft."
- Roll: The "up and down movement of the wings of an aircraft."
- Thrust: The force generated in the opposite direction of the accelerated gas.
- Drag: An "aerodynamic force that opposes an aircraft's motion through the air."
- Kinematics: "The study of motion exclusive of the influences of mass and force. It includes displacement, velocity, and acceleration without regard for the forces acting on a body."

Appendix B provides a summary of the motion fidelity of the related works.

2.3.9 Mission

UAVs may engage in one or more types of missions such as search, situational awareness, imaging, or suppression. A search mission is characterized by a group of UAVs attempting to cover the largest amount of a geographical territory. Situation awareness is similar to search, but also includes "finding, locating, identifying, and tracking potential targets" or other objects of interest [11]. Another type of reconnaissance like mission is imaging which involves constructing a formation of

UAVs equipped with sensors, collecting data from reflection and transmission, and inducing images via mathematical techniques [11]. Finally, some of the surveyed UAVs engage in suppression where a target is destroyed by one or more UAVs.

2.3.10 Geography Bounding

Missions are performed in an area which can vary greatly with respects to the number of dimensions that are modeled. The simplest form of modeling treats a geographical area as a two-dimensional area with borders of known size [11]. A further enhancement involves the mapping of a two-dimensional shape to a three-dimensional shape (e.g., parallelepiped) as is done in [5]. Three-dimensional areas have also been used in UAV simulations and may be of a bounded and of fixed size [1] or unbounded [16]. Additional work has been performed on generalizing the area into n-dimensions by [4].

2.3.11 Geography and Territory

The geography and territory of the simulation may be physical or simulated. Most of the research involves simulated environments, although [16] has used their artificial physical framework to build UAVs that have been deployed in a physical geography and territory.

2.3.12 Units of Measure

All of the related works present units of measure for their geography. In the situation where the units of measure are simulated, the related works speak of a dimensionless unit. Works which refer to physical units of measure will use dimensional units such as meters and seconds.

2.4 Difficulty in Quantitative Comparison of Related Works

In section 2.1 we presented six general approaches to agent cooperation and in section 2.3 we presented qualities to consider in MAS design. The related works presented six different experiments each with its own set of tools that cannot be run directly by other researchers. Additionally, each experiment set up is involved and different with each experiment, and each experiment produces different results that are difficult to compare. If a researcher were to attempt to compare the works, they would have to set up the experiment in the exact, same way, thus introducing redundant effort.

CHAPTER 3

EXPERIMENT DESIGN

In this chapter we present the design of the experiment. The experiment consists of a simulation of cooperative UAVs on a mission to jam air defense radars (ADRs). UAVs are deployed with different starting configurations. In the experiment, we record the time that the ADRs are jammed.

The UAV simulation is structured as follows below.

- There are four ADRs in the territory.
- UAVs begin the simulation in a predefined configuration and with a predefined number of UAVs. All communication occurs between UAVs and ADRs, and there is no centralized control.
- At the start of the simulation the UAVs may or may not be within jamming range of an ADR and the UAVs disperse from each other in an attempt to locate one or more ADRs.
- When a UAV discovers an ADR it attempts to jam the ADR. Other UAVs may also jam the ADR if they are within range of the ADR.
- The time that an ADR spends in a jammed state is recorded and the simulation ends after 1800 seconds (30 minutes) of simulated time have elapsed.
- Each run is recorded and the ADR_{denial} metric is computed.

We defer discussion of the models at this point and discuss it in sections 3.4.1 and 3.4.2 for the state-based and particle-based models, respectively. In the following section, 3.1 we address the independent variables of the experiment and in section 3.2 we discuss the dependent variables. The discussion of the experiment design concludes with section 3.3 which presents the analytical model used for comparison.

3.1 Description of the Example MAS

To ground our quantitative comparison of approaches to agent cooperation, we invented an example MAS. The MAS performs a cooperative "suppression of enemy air defense" (C-SEAD) mission. Each agent in the C-SEAD MAS is a unmanned aerial vehicle (UAV) with the size, weight, speed, and endurance of "low-cost autonomous attack system" (LOCAAS), being developed by Lockheed Martin Missiles & Fire Control in Grand Prairie, Texas. Each agent in our hypothetical C-SEAD MAS uses radio jamming to deny enemy air defense radars (ADRs) time to operate in the clear. The idea is to design a MAS that can perform C-SEAD via EB.

The first approach to agent cooperation is reactive, state-based system where the controller for each agent is modeled as a finite state machine (FSM) that has a finite number of states and transitions. Actions are represented by traversing a transition from one state to another and the state machine is deterministic. Some MAS systems use the same controller for each agent while other systems use hierarchical controllers such as a coordinator (master) and subordinate.

The second approach to agent cooperation in our experiment is "Artificial Physics" (AP). AP is a simple approach to cooperative control in which UAVs are made to move as if in response to artificial electrical or gravitational forces between them and their targets/goals. Each agent and goal/target is assigned a charge potential and a set of "gravitational" equations govern the movement and geometric arrangement of the agents.

3.2 Independent variables/factors

Agents are arranged in square configuration at the start of the simulation by considering the initial x-coordinate and y-coordinate. The agent locations are computed by considering the x-y-coordinates to form a centroid of a square. Agents are arranged in a diamond pattern around the centroid with angles measuring 45, 135, 225, and 315 degrees from the centroid, respectively (i.e. arranged in a "diamond" shape). The length of the diamond's sides is 500 meters. Figure 3.1 depicts a map of the territory in which the simulation takes place.

Figure 3.1 OPNET Territory

The initial starting conditions are computed by considering the cross products of the x-coordinate and y-coordinate entries found in Table 3.1

Table 3.1 Agents' Centroid Coordinates for Initial Positions

x coordinate (km)		y coordinate (km)
1230.0		1430.0
1232.5		1432.5
1235.0		1435.0
1237.5		1437.5
1240.0		1440.0
1242.5		
1247.5		
1250.0		
1252.5		
1255.0		
1257.5		

3.3 Dependent variables

The experiment considers a single dependent variable which is the total duration during which ADRs are jammed. The Cartesian coordinates refer to the position of an agent during any given time of the simulation and at that time interval, each of the ADRs will be jammed or not. The primary measure that we are interested in is the total time that an ADR is jammed, namely the ADR_{denial}. ADR_{denial} is described as the total duration during which each ADR is "jammed" and is calculated as the sum of the durations of all jammed states. Equation 3.1 provides a formal definition of the ADR_{denial} metric where the ADR_{denial} is computed as follows. There are four UAVs in the simulation as denoted by UAV_i (where i has the value 0 to 3) and each of those UAVs may be within jamming distance (R_{jam}) multiple times during the simulation. We must sum all of the time intervals where each UAV is in jamming range of each ADR (ADR_i) range and then divide the resulting sum by four times the simulation run. ADRdenial is equal to 100% when all four ADRs are jammed for the entire duration of the simulation run.

Equation 3.1 Definition of ADR_{denial} Metric

$$ADR_{denial} = \frac{\sum_{i=0}^{i=3} D_{within}(R_{jam}, ADR_i, UAV_0, UAV_1, UAV_2, UAV_3)}{4T_{mission}}$$

3.3.1 Example of ADR_{denial} Calculation

Consider Figure 3.2 which shows the results of a fictitious simulation of 10 seconds where agents attempt to suppress an ADR that emits packets at a maximum rate of 1-Mbps. During the time interval [2,3] the ADR does not emit packets because it is jammed; likewise, the same situation holds during the interval of [5,6]. The interval *t* [6,10] is considered in the calculation of our metric, but it does not increase the value of the metric because the ADR is not in a jammed state (the throughput is not zero). The sum of the duration of the two intervals is two seconds and in this case we state that the ADR_{denial} is 0.2.

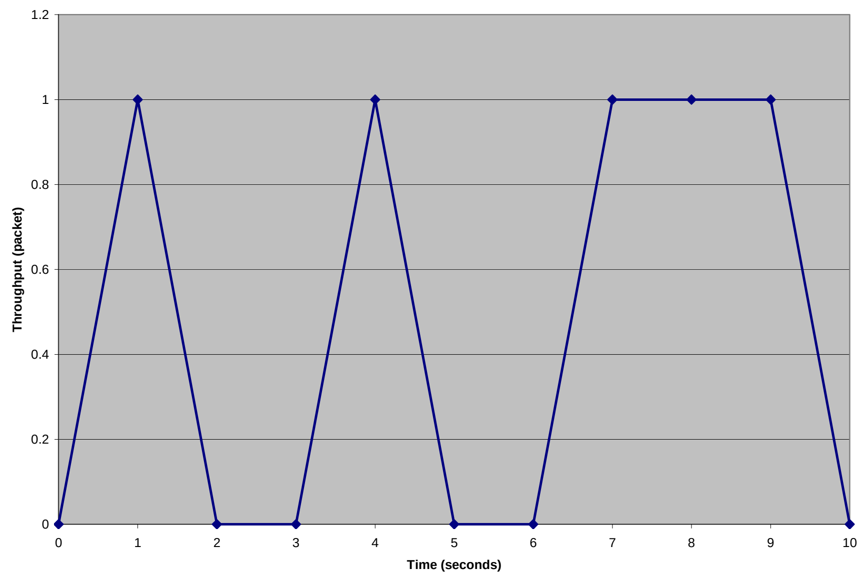

Figure 3.2 Sample ADR_{denial} Calculation

3.3.2 Starting Position of Agent and ADR_{denial}

Agents are configured in a starting position before the simulation begins. Once the simulation starts, agents begin to move away from each other with the expectation that they will encounter an ADR. After an agent encounters an ADR, it begins to broadcast jamming packets. If there are other agents that are within transmission range, they will also attempt to locate the target. Throughout all of the activities (moving away from each other, locating a target, and cooperatively jamming the target) each agent changes its position with respect to time. We will not record the trajectories of each individual agent but we will record the ADR_{denial} that each starting configuration of agents yields. Consider the example simulation presented in Table 3.1. The results depicted in the example represent a simulation with a given initial, starting configuration. It is straightforward to see that given a Cartesian coordinate system and a finite number of starting configurations, there will be one ADR_{denial} metric for each starting configuration.

3.4 Approach to Agent Cooperation

Two approaches to agent cooperation are compared in the experiment—the state-based model and an electro-static force (ESF) model that is based on the Artificial Physics found in [16]. We discuss the state-based model and the FSM that is used as a controller for the agents in section 3.4.1. The ESF model is discussed in depth in section 3.4.2 including the modeling of particles, limitations, system of equations used in the model, and the tuning parameters.

3.4.1 OPNET State-Based Model

The OPNET model represents our implementation of a state-based model. In the following sections, we present the type of FSM that is used in our state-based model, the controller used in the UAVs and ADRs, and the actions that UAVs take in their attempt to jam ADRs.

Our state-based model is a Mealy machine in the sense that it produces output for each transition and is deterministic. States may have one or more transitions that are labeled with an action and these transitions ingress and egress from the states. Each transition can be traversed when the labeled condition (logic statement) is true. Additionally, states have the concept of pre-condition and post-condition which are one or more actions that are executed before entering a state and immediately after leaving a state, respectively.

Consider the example state-based machine in Figure 3.3 which represents a system that sends a packet (we are not interested where) when it receives a signal to send the said packet and then enters a wait state until the packet is consumed. After the packet is consumed, it returns to the original state that it started in and waits for another signal before producing yet another packet.

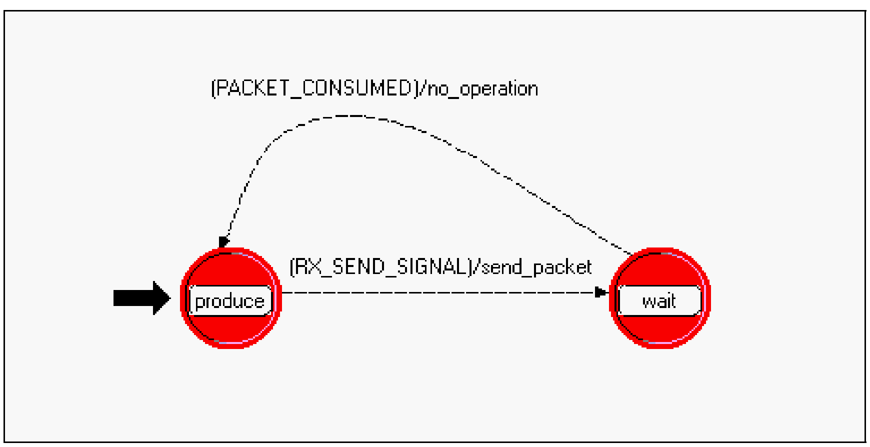

Figure 3.3 Sample Controller

The stated-based system has two states, two transitions, and the initial state of the system is the *produce* state. After the simulation begins the system maintains its state until it receives an event at which point it evaluates the condition associated with the single transition *send_packet* that egresses from *produce*, which in this case is *RX_SEND_SIGNAL*. If the condition *RX_SEND_SIGNAL* evaluates to true, the system moves across the transition, executes the actions associated with *send_packet* and enters the wait state. If there are any post-conditions defined for *produce* they will be executed before the transition; likewise, if there are any pre-conditions they will be executed before entering the *wait* state. The system used as an example is quite simple and straightforward—the controller for the experiment that we present below is not.

We implemented the state-based system using OPNET Modeler which is a discrete-event simulation tool for modeling network and wireless systems. Additionally, OPNET has advanced capabilities in radio modeling, packet transmission, and allows for customized code insertion into the simulation. OPNET was a natural choice because of its state-based nature as well as its advanced modeling capabilities. In the remaining following paragraphs we discuss the geographical layout of the ADRs, how an ADR is modeled, and present the state-based controller for a UAV. Finally, we conclude the discussion of the state-based system by addressing the limitations that exist in such a state-based system when implemented via OPNET.

The sole purpose of an ADR in a defense scenario focuses on radar use to determine if an incoming enemy exists and at what distance and speed the threat approaches. The ADR accomplishes this radar. Furthermore, the ADR is only as effective as the capabilities of the radar—if the radar stops functioning then the ADR becomes useless. Consider the configuration of ADRs in Figure 3.4 which depicts the arrangement of four ADRs (reflector0, reflector1, reflector2, reflector3) in the experiment.

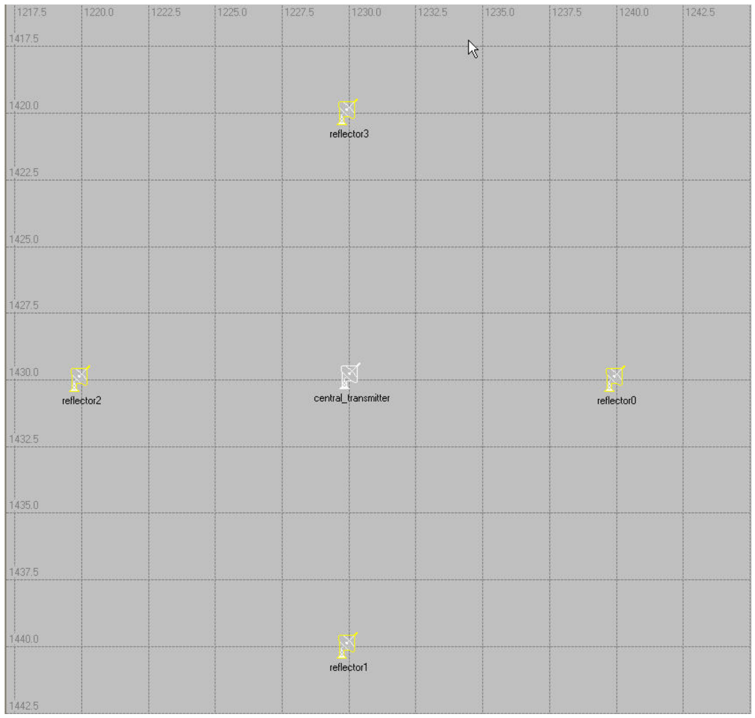

Figure 3.4 Geographical ADR Configuration

Our coordinate system contains the x-axis along the horizontal top edge of the picture and it increases from left to right which the y-axis is found on the vertical left edge of the picture and it increases from top to bottom. The first ADR is located at 1240.0 km and 1430.0 km for the x-coordinate and y-coordinate, respectively. The radar found at the center labeled *central_transmitter* serves produces packets according to probability distribution function (PDF) and transmits the packets to the reflectors who upon reception transmit the packet out on a different channel and with a different packet format. It is important to note that the *central_transmitter* exists only the make the ADRs work more realistically in the simulation. UAVs do not know of about the *central_transmitter* and do not interact with it.

In Figure 3.5 we present the state model for the central transmitter which contains three states and four transitions. The *central_transmitter* begins in an initial state and once the simulation starts an event is raised and the *central_transmitter* evaluates the START condition which is true whenever the first event is raised. After evaluating the start condition the *central_transmitter* generates a packet and enters into a holding pattern—a packet is generated according to a probability density function (PDF) and then the *central_transmitter* pauses. At the end of the simulation, OPNET raises a STOP event which causes the STOP condition to evaluate to true and the simulation ends.

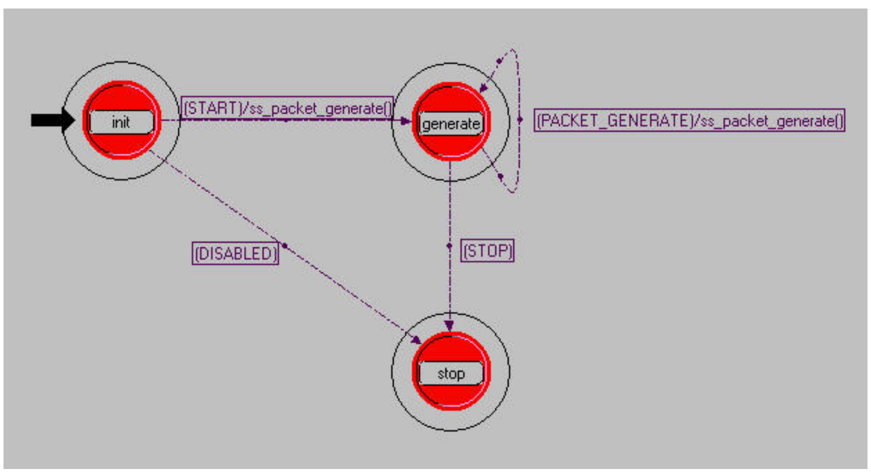

Figure 3.5 Central Transmitter State Model

This scenario imitates a radar transmission. We designed the system so that UAVs are capable of jamming packets that are transmitted by an ADR, but cannot jam the signal from a *central_transmitter* because we wanted to measure the time that an ADR spends in a jammed state.

In our experiment, UAVs begin the simulation at a specified geographical location and begin to move away from each other after the simulation begins in hopes of locating one or more ADRs. Once an ADR is located by an UAV, the UAV broadcasts packets that inform other UAVS within range that they should change their direction and cooperate in jamming the ADR. From this general idea we refined the experiment to consider four states for a UAV.

- State *initial*: represents the UAV before the start of the simulation.
- State *diffuse*: represents the state of an UAV while it is moving away from the other UAVs while it searches for an ADR.
- State *attack*: represents the state an UAV enters once it locates an ADR and begins its attempt to jam the ADR. UAVS can only perform jamming activities from this state.
- State *follow*: represents the state an UAV enters when a fellow UAV has notified it that an ADR has been located.

Previously we mentioned that UAVs have the ability to move by dispersing and then converging when an ADR is found and can transmit various types of packets. In our experiment, we have defined three packet types.

- Jamming Packet Format (*jamming_packet)*: represents the packet sent when an UAV is in the *attack* state and notifies other eligible UAVs to enter their *attack* state to do so.

- Anti-collision Packet (*anticollision_packet)*: represents the packet sent when an UAV is moving away from other UAVs which is done typically at the start of the simulation.
- Follow Packet (*follow_packet*): represents the packet sent when an UAV wishes to indicate that other UAVs should follow it in preparation of an attack.

In Figure 3.6 we present the state-based controller that is found within each of the UAVs and continue our discussion by presenting the actions (transitions) and conditions that are evaluated and ultimately lead to a state change.

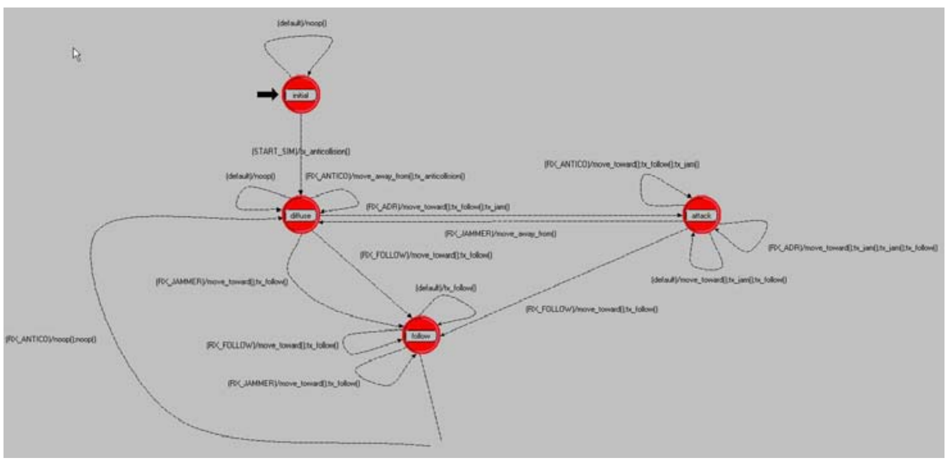

Figure 3.6 OPNET Controller

A transition in the OPNET implementation consists of a condition and an associated action. The condition must evaluate to true for the action to take place. In Figure 3.6 the condition is listed as the first part of the label and the action is listed as the second part of the label and the condition is full capitalized except for the “default” condition, a catch-call condition. Our experiment has five conditions which are summarized below.

- START_SIM: The condition asserts that beginning of the simulation has begun.
- RX_ANTICO: The condition asserts that a packet has been received by the UAV and that the format of the packet is an *anticollision_packet*.
- RX_ADR: The condition asserts that a packet has been received by the UAV and that the format of the packet was sent by on of the ADRs.
- RX_FOLLOW: The condition asserts that the UAV has received a packet with the *follow_packet* format.
- default: The condition does not serve as an assertion. It is the condition that holds true if none of the other conditions for the given state hold true.

Our state-based controller has six basic actions that may appear as a sole action or may be chained together to form composite action. We present each basic option below with a brief discussion and then follow the discussion with a presentation of the overall actions of the controller.

- *noop*: Represents the no-operation action which is an action that is executed but does nothing aside from responding to its invocation. It is analogous to a stubbed function.
- *tx_anticollision*: Represents the action where the UAV transmits an anti-collision packet.
- *tx_follow*: Represents the action where the UAV transmits a packet of type *follow_me* which signals other UAVs to follow the issuer.
- *move_toward*: Represents the action where the receptor UAV moves toward the entity responsible for transmitting the received packet.

- *move_away_from*: Represents the action where the receptor UAV moves away from the entity responsible for transmitting the received packet.
- *tx_jam*: Represents the action where the UAV transmits a packet in an attempt to jam an ADR.

The UAVs are in the *initial* state before the simulation begins and during the first few moments during the initialization and start up. During this period the UAVs may receive interrupts that are signaling simulation initialization and start, but the interrupts are ignored until the START_SIM condition becomes true at which point the UAV emits a *tx_anticollision* packet and promptly enters the *diffuse* state. The start simulation event is received asynchronously by the UAVs and the transmission of the anti-collision packet is responsible for initiating the actions of the UAVs.

After entering the *diffuse* state the UAV pauses and waits for an interrupt to arrive before evaluating the conditions that are applicable in the *diffuse* state. It is during the *diffuse* state that the agents will attempt to locate ADRs by moving away from the centroid of the UAVs every time that the condition RX_ANTICO evaluates true. The movement takes place via the *move_away_from* function, is followed by the emission of an anti-collision packet via the *tx_anticollision* function, and the UAV returns to the *diffuse* state. If a UAV is in the *diffuse* state and it cannot assert that one of the conditions is true, it simply return to the state it was in.

UAVs may leave the *diffuse* state when RX_ADR, RX_FOLLOW, or RX_JAMMER are true which corresponds to receiving a packet transmitted by an ADR, reception of a packet indicating that the receiving UAV should change course and follow the issuing UAV, or reception of a jamming packet transmitted by one of the other UAVs. When an UAV asserts that it received a transmission from an ADR (RX_ADR), it proceeds to the *attack* state by moving toward the ADR. In addition to moving towards the ADR, the UAV transmits a packet that indicates other UAVs should follow it (assuming that other UAVs are in range to receive the packet) and it transmits a jamming packet in an attempt to jam the ADR. Jamming packets are only effective within a one-kilometer radius from the transmitting UAV.

The remaining two transitions, the default transition and the transition when an anti-collision packet is received, are of a simpler nature than the previously discussed transitions action. When the UAV receives an anti-collision packet the condition RX_ANTICO evaluates true it moves away from the UAV that was the originator of the packet and it transmits an anti-collision packet continues to provide feedback for other UAVs to disperse away from the centroid. The default transition executes the *no_op* function which is empty and no action takes place. We will not discuss the default transitions that exist in the *follow* and *attack* states because they have the same semantics.

A second state in the controller is the *follow* state in which agents reside when they are following another UAV in a potential attack on an ADR. In Figure 3.6 the follow state shows four transitions the first and foremost of importance occurs when the condition RX_ANTICO is true which occurs when the UAV receives an anti-collision packet. It moves toward the originator of the packet and it transmits a follow packet in attempt to signal other UAVs to join it moving away from the originator. The UAV does so in hopes of dispersing the potentially clustered UAVs and improving the chance to locating additional targets. After executing a move, the UAV performs no further actions since it executes *no_op* functions.

There are two additional conditions that are considered in the *follow* state, RX_FOLLOW and RX_JAMMER. The first of these conditions causes the UAV to move towards the originator of the packet because the receiving UAV believes that it has received a transmission from a UAV that has found a target. When the condition RX_JAMMER becomes true the UAV also moves toward the originator of the packet because it is follow a UAV which is attempting to jam an ADR. Both conditions cause the transmission of packets (*follow_packet* format).

Of all of the states in the controller, the attack state naturally is the most important since it will be responsible for jamming an ADR and ultimately increasing the ADR_{denial} metric. The state has five transitions two of which transition away from *attack* to *diffuse* and *follow* when RX_ADR and RX_FOLLOW are true, respectively. If RX_ADR evaluates to true then the UAV has received a signal from an ADR which may not necessarily be the ADR that it is transmitting jamming packets to. The action of the UAV is to move toward the originator since it is potential target and then to transmit follow packets in hope that other UAVs will join in the attack. If RX_FOLLOW evaluates to true, then the UAV has received a packet from an UAV that has located a target. The receiving UAV moves toward the originator of the follow packet and it transmits an additional follow packet in hopes of gaining additional support for the attack.

As previously mentioned, there are three transitions that do not transition away from *attack*. The default action deserves mention only in that it continues the attack by moving toward the target ADR and transmitting follow and jamming packets in an attempt to recruit other UAVs and further jam the ADR. When the condition RX_ADR evaluates to true the ADR continues attacking by increasing its proximity to the target, transmission of packets that jam the ADR, and transmission of follow packets in an attempt to recruit other UAVs.

OPNET has a vast library of simulation, allows for the introduction of highly-complex, customized code, and maintains high-fidelity in its representation of signals yet it has limitations particularly where motion is concerned. The models that are built and execute in OPNET state-based models, enhanced FSMs whose actions take place when the model moves from one state to another. The movement takes place when an event is raised by the tool and the condition on the transition is true. During the processing of the event and execution of the actions associated with the transition time stands still because the tool considers the evaluation of the conditions and execution of the actions to be instantaneous, that is, OPNET is a discrete-event simulator. Our UAVs move during this period and their movement is crude—their old position is instantly replaced by the new position—there is no continuous change in motion. Additionally the speed of movement is reached instantaneously and we can only calculate and average speed by considering the total distance moved and the total time of the simulation.

3.4.2 Electro-Static Force (ESF) Model

In section 3.4.2 we present a discussion of the ESF model and we open the discussion with a review of the (AP) approach. After an introductory look at (AP) we present the particles (UAVs and ADRs), the inherent limitations of the ESF model, the system of equations that govern the ESF model, and we close with a discussion of the tuning parameters used in the ESF model.

Previously in the Related Works section, we presented a review of the (AP) framework found in [16] where the entities in the system are modeled as particles and "agents sense and react to virtual forces" that are based on natural physics. The work of [16] has generated results and was particularly intriguing to us on several different fronts. For an experimenter's point of view, we needed a theoretical model that would be used to compare the experimental model (state-based, OPNET model). Artificial Physics readily provides a framework that has already been verified and serves as a starting point for creating a model to verify the experiment.

Artificial Physics use of "natural physics laws" where the system behaves as a particle system was also attractive because of the number of tools, such as MATLAB, that readily model such behavior. The equations that govern the system are simple and MATLAB readily handles their calculation. Additionally, a system modeled in terms of equations is also attractive because it readily generates numbers. We decided to implement an electro-static force (ESF) model that was based on (AP).

In our ESF model both the ADRs and UAVs in the system are modeled as particles and are arranged in the same configuration as the state-based model. ADRs are represented by very massive particles that move negligible amounts during the simulation and UAVs are represented by lightweight particles (with respect to the ADRs).

The AP model we developed does not feature friction or thrust. Thus the UAV particles tend to accelerate and decelerate in an unrealistic manner. To prevent unbounded accelerations and hence numerical integration problems in the Matlab ode45 solver, we imposed a maximum ESF. The maximum force corresponds to the centripetal force on a LOCAAS agent making a 2 g's level turn.

We modeled ADRs as massive particles so that we could simulate them as non-mobile entities as closely as possible. ADRs are modeled as particles with a positive charge of 11 C and with a mass of 1,000,000,000 kilograms.

On the other hand UAVs were modeled as very lightweight particles with respect to ADRs. UAVs were modeled with a negative chare of 0.4 C and with a mass of 45.36 kilograms (i.e., the mass of LOCAAS, upon which our UAVs are based).

There are two particular limitations in the ESF model—the lack of friction and the highly idealized representation of the UAVs and ADRs. The first limitation of the ESF model is with regards to the lack of friction (i.e., resistance to movement, aerodynamic drag) in the system. The particles that represent the UAVs are in constant motion during the simulation with multiples forces of attraction and repulsion acting on them. Friction is not modeled in the system and the UAVs offer no resistance to their change in motion when they encounter one another or an ADR. If we want to design a system that is more representative of a physical system, we would need to account for friction and a thrust force to counteract friction.

Another limitation of the ESF model is inherent to modeling the UAVs and ADRs as particles—the system is highly idealized. ESF does not consider the physical attributes of a UAV such as the fuel source, communication, and mechanics of the UAV itself. The same is true for the ADRs—with ESF, the system consists of particles interacts with each other motivated by a system of equations. Explicit modeling of fuel source, communication, etc. would be mandatory if the system were to be used to accurately model physical UAVs.

Our ESF model is based on the physics equation for Coulomb's Law which is shown in Equation 3.2.

In this equation, force (F) is calculated by considering the product of the charge of the two particles divided by the distance between the two particles. The value constant that appears in the equation is readily recognized as the permittivity constant.

Equation 3.2 Coulomb's Law

$$F = \frac{1}{4\pi\varepsilon_0}\frac{q_1 q_2}{r^2}$$

Equation 3.3 Permittivity Constant

$$\varepsilon_0 = 8.99\times10^9 \frac{N\cdot m^2}{C^2}$$

In this equation, force (F) is calculated by considering the product of the charge of the two particles divided by the distance between the two particles. The value constant that appears in the equation is readily recognized as the permittivity constant.

In our ESF model we use a very similar equation (Equation 3.4) save for the notation of the constant. The value of r_{self} is the coordinate location of the UAV (or particle) under consideration and the value r_{other} is the coordinate location of the other UAV (or particle) and their difference is the distance between the two particles. The numerator contains three terms *Const*, q_{self}, and q_{other} which are analogous to the Coulomb's Law in the fact that that we consider a constant and the charge of the two particles (UAV or ADR). It is important to also note that the ESF equation is sensitive to the position of the UAV under consideration because we consider the sign of the force to account for the direction that an agent will move in the Cartesian coordinate system. Consider the situation where UAV_1 has and x and y coordinate that is greater than UAV_2, we want the UAV_1 to experience a force that pushes it away from UAV_2 which in this case would be an increase in both the x and y value of its coordinates. In this situation, we consider the force to be positive. Likewise, when UAV_1 has an x and y coordinate that is less than UAV_2, we want the UAV_1 to experience a force that pushes it away from UAV_2, but such a force must decrease the x and y position. In this situation, we consider the force to be negative.

Equation 3.4 ESF Force Equation

$$ESF(r_{self}, r_{other}, q_{self}, q_{other}) = \begin{Bmatrix} k_{ESF} \dfrac{q_{self} \cdot q_{other}}{(r_{self} - r_{other})^2}, r_{self} - r_{other} \geq 0 \\ -k_{ESF} \dfrac{q_{self} \cdot q_{other}}{(r_{self} - r_{other})^2}, r_{self} - r_{other} < 0 \end{Bmatrix}$$

It is important to note that that subtraction in the denominator implies the distance calculation listed below.

Equation 3.5 Distance Formula

$$r = \sqrt{\left(y_{self} - y_{other}\right)^2 + \left(x_{self} - x_{other}\right)^2}$$

In Equation 3.6, we present the constant used in the ESF model which maintains the same units as Coulomb's Law but we have expanded the Newton to the base quantities that represent it (mass and time) to simplify calculation.

Equation 3.6 ESF Constant

$$k_{ESF} = \frac{1}{4\pi(8.85\times10^{-12})}\cdot\frac{kg\cdot m^3}{A^2\cdot s^4}$$

Our ESF contains more than two particles, in fact, there are eight particles and each particle exerts a force on the rest of the particles. We can account for all of the forces by performing a vector summation of all the forces acting on a given particle. Consider a given particle, *i*, the total force acting on it is given by the equation shown in Equation 3.7 where "i" is the "self" particle.

Equation 3.7 Force Summation on the ith Particle

$$F_i = \sum_{j=1, i\neq j}^{n} ESF(r_i, r_j, q_i, q_j)$$

The only external forces on particles are electrostatic forces. Applying Newton's Second Law we write that the sum of the external forces acting on a particle is equal to the mass of the particle time the acceleration of the particle. In the ESF model, we found that we had to artificially limit the maximum force acting on a particle to be 2600 N. This force corresponds to the amount to centripetal force that must act on a LOCAAS UAV when it makes a 2-g level turn. Without restricting the force, MATLAB's ODE solver function ran into numerical integration difficulty when the forces became astronomically large.

Equation 3.8 Applying Newton's Second Law

$$\vec{F}_i = m_i \vec{a}_i = \sum_{j=1, i \neq j}^{n} \overrightarrow{ESF}(r_i, r_j, q_i, q_j)$$

We then re-write the acceleration found in Equation 3.8 according to the definition of acceleration as force divided by mass. But force is simply the change in velocity with respect to time and we further re-write our equation by considering that velocity is the change of position, in this case *r*, with respect to time. This produces Equation 3.9.

Equation 3.9 Acceleration

$$\vec{a}_i = \frac{\vec{F}_i}{m_i} = \frac{d\vec{v}_i}{dt} = \frac{d^2\vec{r}_i}{dt^2}$$

Using Cartesian coordinates x and y we have the necessary information to derive the force that exists along the coordinates x and y. Consider Equation 3.9 and substitute our x and y coordinates. We can derive the equations for the force that acts on the x and y coordinates as depicted in Equation 3.10 and Equation 3.11, respectively.

Equation 3.10 Acceleration in X Direction

$$a_{i_x} = \frac{d^2 x_i}{dt^2} = \frac{\sum_{i \neq j}^{Particles} ESFx(q_i, x_i, y_i, q_j, x_j, y_j)}{m_i}$$

Equation 3.11 Acceleration in Y Direction

$$a_{i_y} = \frac{d^2 y_i}{dt^2} = \frac{\sum_{i \neq j}^{Particles} ESFy(q_i, x_i, y_i, q_j, x_j, y_j)}{m_i}$$

The complete state of the ESF model is uniquely determined by its "System State Variables". The system state variables for the ESF model are the position and speed of each particle. The equations that completely describe the ESF model are as follows.

"x" represents the horizontal Cartesian position coordinate of the particle.

"y" represents the vertical Cartesian position coordinate of the particle.

"u" represents the horizontal Cartesian velocity coordinate of the particle.

"v" represents the vertical Cartesian velocity coordinate of the particle.

"adr" represents an ADR particle.

"uav" represents an UAV particle.

"0" represents the first particle of a type.

"3" represents the fourth particle of a type.

"y" represents a state variable.

"ydot" represents the time derivative of a state variable.

y1 = xadr0

y2 = yadr0

y3 = uadr0

y4 = vadr0

…

y29 = xuav3

y30 = yuav3

y31 = uuav3

y32 = vuav3

ydot1 = uadr0 = y3

ydot2 = vadr0 = y4

$$ydot3 = \frac{\sum_{adr0 \neq j}^{Particles} ESFx(q_{adr0}, x_{adr0}, y_{adr0}, q_j, x_j, y_j)}{m_{adr0}}$$

$$ydot4 = \frac{\sum_{adr0 \neq j}^{Particles} ESFy(q_{adr0}, x_{adr0}, y_{adr0}, q_j, x_j, y_j)}{m_{adr0}}$$

…

ydot29 = uuav3 = y31

ydot30 = vuav3 = y32

$$ydot31 = \frac{\sum_{uav3 \neq j}^{Particles} ESFx(q_{uav3}, x_{uav3}, y_{uav3}, q_j, x_j, y_j)}{m_{uav3}}$$

$$ydot32 = \frac{\sum_{uav3 \neq j}^{Particles} ESFy(q_{uav3}, x_{uav3}, y_{uav3}, q_j, x_j, y_j)}{m_{uav3}}$$

These 32 equations are solved numerically using MATLAB's "ode45" solver function, subject to initial conditions for y1 though y32.

Our discussion turns next to the constant values that are necessary to initialize the ESF model. These values are used to give each agent and ADR charge and mass in the manner of [16] and are used to make the ESF model behave like the OPNET model. The constant values were derived through empirical results and tuning. We used the Matlab "Genetic Algorithm & Direct Search" toolbox to discover the tuning parameters.

Consider the particles that represent agents. We want the agents to repel each other and to be extremely light in mass with comparison to the ADRs. We adopted a charge of negative charge for all of the agents of -0.4 Coulombs and a mass of 45.36 kg (This equals 100 pounds, which is the mass of a LOCAAS UAV) which allows the agents to be readily attracted to ADRs and repel each other mildly.

In the ESF model ADRs should be stationary relative to other ADRs and UAVs. In an effort to capture this requirement we chose a mass value of one-billion kilograms (10^9 kg) and a positive electric charge of 11 Coulombs which ensured a negligible amount of movement for the ADRs.

In the state-based model implemented in OPNET the jammed state of an ADR could be measured readily and directly because the tool had built in support for gathering such statistics, but our ESF model implemented in MATLAB did not provide for the concept of a "jammed state." We directly measured the distance at which ADRs became jammed in the OPNET model which was at 1.05 km and adopted the same distance for the ESF model. In the ESF model we consider an ADR to be jammed when one or more particles that represent an UAV are at a distance of 1.05 km or less. The ADR is in a jammed state during the time interval in which the jammed distance holds and the ADR_{denial} metric is the sum of the time spent in a jammed state divided by the four times the total simulation time.

Velocity is not a system state variable in our OPNET model. Hence there is no way to set an initial velocity for UAVs in the OPNET model. Velocity is indeed a system state variable in our ESF model. Hence it is necessary to provide an initial velocity to each particle for each simulation run.

In our ESF model, we decided to make the initial speed of each UAV particle equal to the speed at which the LOCAAS UAV flies, 100 m/s. For the initial speed of 100 m/s, we varied the initial direction of motion through the angles 0, 45, 90, 135, 180, 225, 270, and 315 degrees. We also ran cases for the ESF model in which the initial speed was set to 0 m/s. Thus there were 10 values of initial UAV velocity run for the ESF model.

The agents in the ESF will have a tendency to move forever because they are an ideal particle system. We adopted the termination criteria of time just as we did in the state-based model. Our ESF simulation terminates after 1800 seconds.

3.5 F Statistic

Our experiments yields a significant amount of data—ten simulation runs with 60 data points per simulation—and it is necessary to test the data and determine if the observed differences are significant. A statistic tool that is well suited for this type of comparison in the F statistic which is capable of comparing two or more groups and determining whether the observed difference in the mean of the groups is significant. In our case we are interested in determining if the mean ADR_{denial} rate is different between the different simulation runs.

Consider an analysis situation where there are *s* groups of equal size *n* and each group has a sample mean and a standard deviation as depicted in Equation 3.12 and Equation 3.13, respectively.

Equation 3.12 Sample Mean

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Equation 3.13 Standard Deviation

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}$$

The F statistic is calculated using the "within groups" variance and the between groups variance. We first show the formula for the F statistic in Equation 3.14 which has the "between groups variance" as the numerator and the "within groups" variance as the denominator.

Equation 3.14 F Statistic Formula

$$F = \frac{s_{bet}^2}{s_{wit}^2}$$

In the calculation of the F statistic the "within groups" variance represents the variance of the sample means. For a comparison of *s* groups each with *n* observations the "within groups" variance is calculated according to Equation 3.15.

Equation 3.15 Within Groups Variance

$$s_{wit}^2 = \frac{1}{s}\sum_{i=1}^{s}\sigma_i^2$$

The calculation of the "between groups" variance is more involved and it consists of the computation of the standard deviation of the *s* observed sample means (refer to Equation 3.12 and Equation 3.13). Once the standard deviation of the sample means ($\sigma_{\bar{x}}$) has been computed, the "between groups" variance is computed according to Equation 3.16.

Equation 3.16 Between Groups Variance Formula

$$s_{bet}^2 = n \cdot \sigma_{\bar{x}}^2$$

The equation that we presented in our discussion so far, calculates the F statistic which alone does not mean anything. An F statistic only has meaning when it is compared to the critical F value ($F_{critical}$) so that the null hypothesis may be accepted or rejected. $F_{critical}$ is computed by considering the degrees of freedom (DOF) present in the numerator and denominator values used to calculate the F statistic and the level of confidence. It is common to adopt a confidence level of 95% for an experiment such as the one we have carried out.

Equation 3.17 Degrees of Freedom

$$n_N = s - 1$$
$$n_D = n_N \cdot (n - 1)$$

Finally, the critical F value is found by consulting a table of critical F values such as the one found in [6] and considering the DOF present in the numerator, denominator, and the confidence level.

3.6 Potentially Conflating Factors

3.6.1 Real-World Limitations and Simplifying Assumptions

The stated-based model and AP model both share many similarities—they are both experiments where the effectiveness of UAVs is measured by the ADR_{denial} metric.

While UAVs in both models engage in the same types of activities it is important to note that there are potential conflating factors in the experiment especially with regards to the range of the sensors and range of communication.

Consider the ESF model that is based on the AP work of [16]. It uses a force equation that is based on Coulomb's Law where the force is inversely proportional to the square of the distance between two particles. It is noteworthy that even though the distance may be very large, there is still a force exerted between the particles. The ESF model uses force to represent sensors and communication and it is this representation that causes the sensors and communication in the ESF modeled to be unlimited. In our ESF model although two UAVs may be on opposite ends of the territory, they are still "communicating" because they exert an influence via force on each other.

ESF stands in stark contract to OPNET which explicitly addresses such limitations. OPNET contains a sophisticated radio transceiver pipeline that models transmission delay, link closure, channel match, transmitting antenna gain, propagation delay, receiving antenna gain, receiver power, background nose, interference, error allocation, bit error rate, signal to noise ration, and error correction if applicable. In our state-based simulations UAVs are only capable of communicating with other UAVs across relatively short distances, may become isolated from other UAVs, and may cease to participate in the simulation because they have moved outside of the reception range. Of equal importance is the fact that because OPNET explicitly models communication, a UAV is only effective in suppressing an ADR at a close range.

Based on the related work presented in CHAPTER 2 we used OPNET Modeler [13] to design, implement, and evaluate a stated-based controller that operated inside each UAV. The controller that allowed each UAV to engage in activities such as dispersing (moving away from each other), following, and attempts at cooperative attack via jamming. OPNET Modeler allowed us to model communication between UAVs as well as ADR transmission in great detail as it provides high-fidelity simulation tools for modeling of packet radio transmissions. Using OPNET, it was possible to use built-in features to detect ADRs, the jammed states of ADRs, and communication between UAVs. Additionally a vast array of statistical tools and metrics were available. While OPNET presents such rich features for radio packet transmission it is does a poor job of simulating movement and cannot do so in a continuous fashion and instead can only represent it as incremental, instantaneous changes in position because OPNET actions take place during an event that does not consume time.

Our AP model was implemented using MATLAB [7] which is a commercial tool for numerically solving a given system of equations. In this model MATLAB was used to solve the equations that represented the interactions between the UAVs and ADRs. While MATLAB is very capable in solving the equations and readily providing us with numerical data for our model it did not represent communication between UAVs with a high degree of fidelity as did OPNET. In particular, UAVs and ADRs interacted with each other across distances that would have been out or range in the OPNET simulation. Additionally in the AP model there was no way to represent jamming radio transmission explicitly and we relied on an empirically derived distance to calculate ADR_{denial} metrics. When the distance between an UAV and an ADR was one kilometer or less the ADR was considered jammed.

Both the OPNET model and the MATLAB model were tuned perform the C-SEAD mission as best as possible. Consider the OPNET simulation at the start of our research. We did not know at what distance an UAV would effectively jam and ADR and we derived the optimal values for transmission power and distance through empirical investigation. Likewise, this distance was used in the AP model. Additional empirical measures were made for the simulation time and to discover what the minimal actions (dispersion, follow, jam) for the simulation were.

CHAPTER 4

EXPERIMENT ANALYSIS

We carried out an experiment that compared two different models for UAV cooperation in a C-SEAD mission. The C-SEAD mission was comprised of a simple goal—the UAVs should attempt to locate and suppress ADR that are in the area and should attempt to do cooperatively. UAVs started the simulation in a predetermined configuration and with initial coordinates and dispersed during the beginning of the simulation in an attempt to located ADRs. If an UAV located an ADR, it attempted to jam it. In some instances the jamming behavior was cooperative.

Both the stated-based and AP models generated data that was used to calculate ADR_{denial} metric which measured how effective the UAVs were in their mission. We present a summary of the data for the ESF Model, the stated-based model, a comparison of the surfaces generated by the ADR_{denial} metric, and an F-Statistic analysis in the following sections. One of our great interests was to answer the following questions. Are the differences in the mean ADR_{denial} for the models due to chance or are the differences significant? We address the question in with a statistical analysis using an F-Test.

4.1 ADR_{denial} ESF Model

Figure 4.1 presents a three-dimensional graph of the ADR_{denial} value which is presented on the z axis. The *AcenterX0* represents the initial x coordinate and is found along the x axis and *AcenterY0* represents the initial y coordinate, and is found along the y axis. In Figure 4.1 the units along the x and y axes are in kilometers and the z axis assumes a dimensionless number because it represents ADRdenial, the percentage of the simulation time in which the one or more ADRs was jammed.

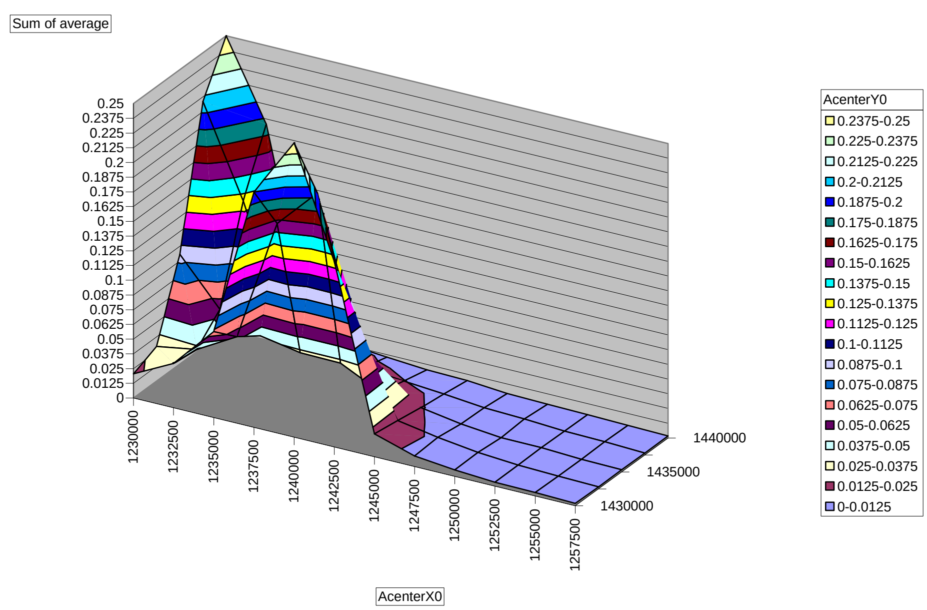

Figure 4.1 ESF ADR_{denial} Graph

There are two points of particular interest in Figure 4.1—the two peaks where the ADR_{denial} value reaches 25%. Both of these peaks correspond to starting locations where the UAV were immediately next to an ADR (the x-y coordinates {1235, 1430} and {1230,1440}. We expect such a result when UAVs start next to an ADR. At the beginnings of the simulation the UAVs would repel each other a minimal amount, but the attraction to the ADR would be a much greater force. The UAVs would spend most of the time around the ADR, but would not approach other ADRs and at most we would expect a 25% suppression of ADR activity.

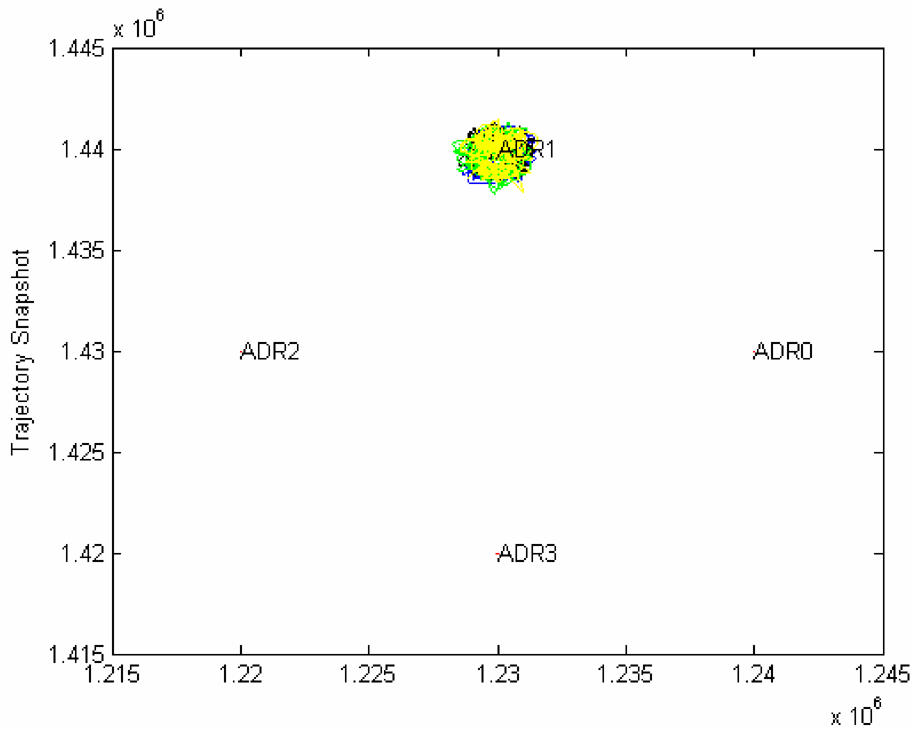

Figure 4.2 ESF Model Trajectories When UAV X0, Y0 Lies Over ADR1

There is another additional point of interest which is the trough that occurs between the two peaks. The trough is easily explained when we consider the fact that as the UAVs increase their distance from the ADRs the force that attracts on them will decrease. This decrease will reduce the amount of time that the UAVs spend within jamming distance. The trough results from an increase in distance between one of the ADRs (x-y coordinate {1235, 1430}) and a gradual decrease in distance from the second ADR (x-y coordinate {1230, 1440}).

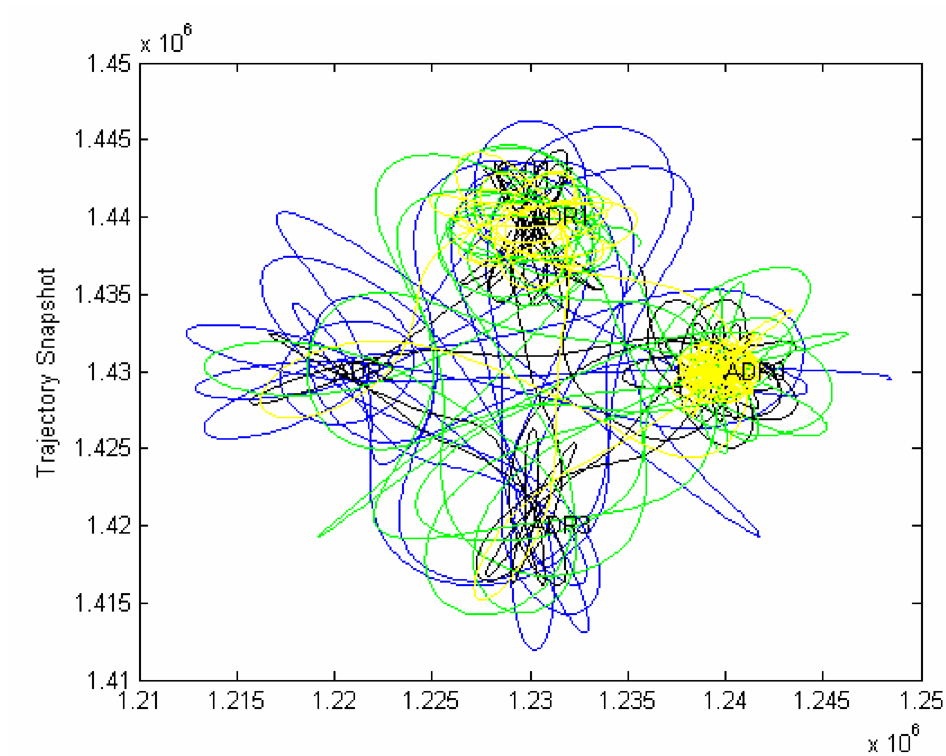

Figure 4.3 Typical ESF Model Trajectory when UAVs are Launched from {1235, 1430}

4.2 ADR_{denial} State-Based Model

Figure 4.4 presents a three-dimensional graph that summarizes the results of the state-based model. The units are presented directly in kilometers.

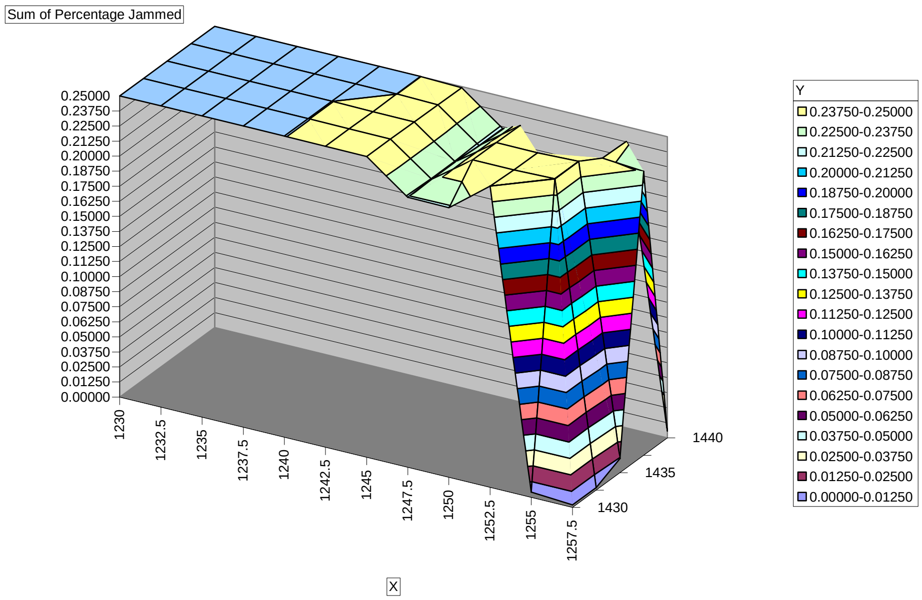

Figure 4.4 State-Based ADR_{denial} Graph

It is interesting to note that the ADR_{denial} value is constant in the region bounded by the x coordinates in the interval [1230, 1242.5] and y coordinates in the interval [1430, 1440]. In this portion of the experiment we do not observe a decline in ADR_{denial} value with increasing distance between UAVs and ADRs. We can reasonably explain this behavior by considering that the maximum distance between a UAV and an ADR which occurs when the UAV is at {1242.5, 1440} and ADR_1 is at {1230, 1440}. The distance between the two objects is approximately 7.1 kilometers. Also consider the fact that the ADR packets arrive at the average rate of 27 packets per second and the UAV moves at 0.01 kilometers per ADR packet that is received. We calculate an approximate speed of 972 km/h in Equation 4.1 and from this value it is straightforward to calculate that the time required by a UAV to reach the ADR_1 is about 0.02 seconds. Such a low value readily translates into a value about 0.25 for ADR_{denial}.

Equation 4.1 Speed Calculation

$$\frac{27\,packet}{s}\cdot\frac{60s}{1\min}\cdot\frac{60\min}{1hr}\cdot\frac{0.01km}{packet}\approx 972km/h$$

Once we move beyond the region bounded by [1240, 1242.5] along the x axis and [1430, 1440] along the y axis the distance becomes significant. Once UAVs begin the simulation with coordinates greater than {1252.5, 1430}, the ADR_{denial} value drops of rapidly. This situation is also explainable if we consider the actions that the UAVs take during the start of the simulation which is to disperse. At these distances UAVs will tend to travel along a vector away from the centroid for some time before they come close enough to an ADR to receive an ADR packet and they will have to travel an added distance back toward the ADR. A second phenomenon must be taken in account. As the distance increases the UAVs that move along the vectors that are 45-degrees and 315-degrees from the centroid may disperse to a distance where they are out of range of both the ADRs and other UAVs. When this happens the ADR_{denial} value will be reduced by 25% if one UAV is lost and 50% if two are lost. This is particularly noticeable toward the right hand side of Figure 4.4.

4.3 Comparison of the two-surfaces

We were very interested in comparing both of our results. In Appendix C we have included a listing of each of the initial starting positions as well as well as the ADR_{denial} value associated each position for both the state based and ESF models. Additionally we have included the mean and standard deviation for each of the simulation runs. The values for the state-based implementation in OPNET were obtained by exporting each simulation run to an Excel and running a batch program that calculated the ADR_{denial} metric according to our definition. For the ESF models the ADR_{denial} values were obtained via MATLAB. Below we present the F-Test calculation using the values cited in Appendix C.

We know that in order to calculate the F-Statistic according to Equation 3.14. we must calculate the "between groups" variance and the "within groups" variance. Using the values from Appendix C we have the "between group" variance and the "within group"in Equation 4.2 and Equation 4.3, respectively.

Equation 4.2 "Between Groups" Variance Calculation

$$s_{bet}^2 = n \cdot \sigma_{\bar{x}}^2 = 10 \cdot 0.005266531 = 0.183263$$

Equation 4.3 "Within Groups" Variance Calculation

$$s_{wit}^2 = \frac{1}{s}\sum_{i=1}^{s}\sigma_i = 0.00505086$$

We can now readily calculate the F-Statistic according to Equation 3.14 and the value is 36.28367. We now tentatively adopt our null hypothesis H_0 and assume that the differences in means are not significant.

Before testing our hypothesis we must calculate the $F_{critical}$ value which requires us to calculate the degrees of free with according to Equation 3.17. The DOF of the numerator is 9 and the DOF of the denominator is 531. Given DOF values and a confidence level of 95% the $F_{critical}$ value is 1.9. Most reference books do not include tables that are large enough to capture the DOF of the denominator and the value was calculated using an online tool [2] for such large calculations. The $F_{critical}$ value in this case is not greater than the F-Statistic and we must reject the null hypothesis and instead accept the alternative hypothesis—the difference in means is significant.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

The difference in means of the ADR_{denial} metric of the SBM and the ESFM cannot reasonably be attributed to chance. The ADR_{denial} "plateau" of the State-Based Model is equal to the ADR_{denial} "peaks" of the ESF Model. I.e., the average performance of the SBM is as good as the very best performance of the ESFM. ADR_{denial} metric and the state-based and ESF models. Chapter 4 provided a review and analysis of the experiment including a statistical analysis. We attribute the significant difference in UAV cooperation performance to the fact that we imparted significantly more design intent into the ADR_{denial} metric value was higher in the state-based model and we safely concluded that the difference was not due to chance alone via an F-test. Yet, there is much more work to in refining our work.

5.1.1 Models

It is interesting to note that the F-test caused us to reject the null hypothesis and we had to accept the alternative—variances in the SBM model mean ADR_{denial} metric were not due to chance alone. This result means that our models do not validate each other. We imparted design intent into the SBM by defining meaningful operating modes for the UAVs (i.e., initial, diffuse, attack, follow), by defining meaningful types of packets (i.e., ADR, jam, follow, anti-collision), by defining meaningful actions to be performed by the UAVs (i.e., move toward, move away from, jam), by defining meaningful transitions between UAV operating modes, by defining meaningful conditions on those transitions (i.e., a certain type of packet have just been removed from the queue), and by carefully sizing UAV packet transmission power to jam an ADR at a desired distance. The ESFM in contrast, we imparted design intent by only choosing the masses and charges of the UAV and ADR particles.

Foundational differences between the models are that the ESFM is "continuous-time" and the SBM is "discrete-event". obvious that what is needed is a kind of interoperable simulation that can coordinate between OPNET Modeler and Matlab as they run—the OPNET model can generate accurate radio packet data and cooperation commands as long as the positions of the UAVs are known. The Matlab model can generate accurate positions of the UAV as long as the cooperation command being executed by each UAV is known. Future work can investigate the interoperable simulation needed to make this happen. The DoD/IEEE HLA [15] is a promising infrastructure to investigate this further.

5.2 Future Work

If we had to do this research over again, if we wish to publish our research outside of UTA, or if we wish to otherwise develop this research further, we would work harder to place the ESF approach to UAV cooperation and the SB approach to UAV cooperation within the same mathematical framework: As for the ESFM, we would define a set of system state variables and equations to model the SB approach. By definition, the state of a system is completely defined by the assignment of values to its System State Variables. Such a model would be placed within the discipline of discrete-event modeling, queuing theory, Markov processes, etc. Each UAV or ADR would be a state transition machine with an input queue to receive packets from all other UAVs and ADRs. Packets generated by each UAV and ADR would potentially enter the input queue of any other UAV or ADR, thus the queuing representation would be a fully connected graph. listed here.

System state variables modeling the SB approach would include

1. the current coordinates of each UAV,
2. the previous coordinates of each UAV,
3. the operating mode or state of each UAV and ADR,
4. the packet input queue for each UAV and ADR, and
5. the UAV or ADR that is the subject of each UAVs move_away_from or move_toward behavior.

The equations modeling the SB approach would relate

1. the current location of each UAV to the UAV's previous location and the UAV or ADR to which the UAV is moving toward or away from, if any,
2. the probability that a UAV or ADR receives a certain packet from a UAV or ADR as a function of distance and possibly other factors as well,
3. the next operating mode or state for a UAV or ADR depending on the current state and packets present in the object's input queue, and

4. the next contents of the “future event list” as a function of the current contents of the future event list and the current state of the system.

An equation model such as the one outlined above would be easier to understand and modify than the OPNET simulation model we currently possess. We expect that it would be clearer how to improve the equation model in the area of movement fidelity, even though it is a discrete-event model.

An equation model for the SB approach such as the one outline above would be amenable to sizing decisions that are based on established theories of physical phenomena (e.g., electromagnetic theory), rather than being sized by manual tinkering as we did with the OPNET model we currently possess.

We could execute the equation model outlined above using Matlab or a discrete-event simulation package. We could then validate the equation model against the current OPNET model.

The current, equation model of the ESF approach could be placed alongside the proposed equation model of the SB approach. Such a side-by-side comparison, using the same mathematical framework, would reveal more qualitative and potentially quantitative differences between the SB and ESF approaches.

In addition to refining our mathematical framework and improving its rigor, future work should consider the other approaches to agent cooperation in light of the ADR_{denial} metric. We chose to develop a metric and carry out an experiment that allows us to compare two disparate approaches to agent cooperation. The state-based and AP approaches were natural candidates for comparison because they are very disparate. All of the state-based approaches we reviewed were comprised of a state machine controller embedded in each UAV while the AP approach was governed by a set of equations. Other approaches to agent cooperation were not compared using the ADR_{denial} metric and a comparison should be undertaken in the future work—but which is the next approach to evaluate? In [14] we reviewed the concept of a neighborhood and flocking which would be the next candidate approach to evaluate. The neighborhood approach is very different from both the state-based approaches of [5], [11], and [12] because it is a particle based system. Additionally this approach is very different from the AP approach found [16] because the particles have geometric representation and orientation whereas the AP approach does not possess such characteristics because of its point-particle representation. The comparison could be side-by-side comparison of all three approaches or the neighborhood approach could be compared against one of the state-based approach or the AP approach.

APPENDIX A

NOMENCLATURE

Table A.1 NOMENCLATURE

term	**Definition**	**units**	**Formula**
A	Ampere		
ADR	Air Defense Radar		
ADR_{denial}	ADR_{denial} is defined as the total duration during which each ADR is “jammed” and is calculated as the sum of the durations of all jammed states.	s	$ADR_{denial} = \frac{\sum_{i=0}^{i=4} D_{within}(R_{jam}, ADR_i, UAV_0, UAV_1, UAV_2, UAV_3)}{4T_{mission}}$
Ampere	<definition of an Ampere…>	A	
C	Coulomb	C	
Coulomb	<definition of a Coulomb…>		
D_{within}	The total time duration during a mission simulation during which a particle ADR_i is within distance R_{jam} of particles UAV_0, UAV_1, UAV_2, or UAV_3.	s	
e_0	Electrostatic Permittivity Constant		
ESF	Electrostatic force acting on a particle	N	$ESF = \frac{1}{4\pi\varepsilon_0}\frac{q_1 q_2}{r^2}$
F	Total force acting on a particle	N	
FSM	Finite-State Machine		
k_{ESF}	ESF constant	?	<include the formula>
kg	Kilogram		
m	Meter		

Table A.1 - *Continued*

MAS	Multi-Agent System		
N	Newton		
Newton	the force needed to accelerate a mass of 1 kg. at a rate of 1 m/s/s		
other	a subscript denoting a particle that exerts a force upon the particle under consideration (i.e., the "self" particle)		
p	The number of times the radius of a circle can be wrapped around a circle		
q	Charge of a particle in Coulombs	C	
r	Position of a particle in Cartesian coordinates	m	
R_{jam}	For the AP model, if a UAV agent particle comes within R_{jam} of an ADR particle, the ADR particle is assumed to be jammed.	m	$R_{jamFSM} \approx 1050$ m $R_{jamAP} = 1050$ m
s	second		
self	a subscript denoting the particle upon which forces are acting		
Territory	…		
$T_{mission}$	The duration of a simulated mission	s	$T_{mission} = 1800$ s
UAV	Unmanned Aerial Vehicle		
x	Horizontal coordinate of a particle	m	
y	Vertical coordinate of a particle	m	

APPENDIX B

RELATED WORKS COMPARISON

Table B.1 RELATED WORKS COMPARISON

Realted Work Name	Finite-State Machine Controlled?	FSM: State Transition Trigger	Real Variable, Equation-Based Ctrl?	Medium for Agent Cooperation	Emergent MAS Behaviors
Parunak	✓	combination of context, constraints, and preferences	×	Inverse Pheremone	cooperative search
Gaudiano	✓	signals from agents	×	Pheremone	cooperative search
Sears	×	×	✓	artificial forces	formations
McLain	×	×	✓	commmunication manager	cooperative attack
Reynolds	×	×	✓	"neighborhood"	flocks
Espino	✓	signals from agents or targets	×	radio proximity	cooperative search
Gazi & Passino	×	×	✓	function of attraction and repulsion	aggregation and stabilization

Table B.1 - *Continued*

Realted Work Name	Sensor Fidelity, Simulated	Comm Fidelity, Simulated
Parunak	idealized	idealized
Gaudiano	Stochastic Cone	idealized
Sears	idealized	N/A
McLain	idealized	idealized
Reynolds	idealized	N/A
Espino	High	High
Gazi & Passino	N/A	N/A

Table B.1 - *Continued*

Realted Work Name	Motion, 3-D?	Motion, Control Pitch?	Motion, Control Yaw?	Motion, Control Roll?	Motion, Control Thrust?	Motion, Drag?	Motion, Kinematics?	Motion, Dynamics?	Mission, Search?	Mission, Situation Awareness?	Mission, Imaging?	Mission, Suppression?	Geography Bounding, Simulated	Geography, Territories, Simulated?	Std Units/Measurement, Simulated?
Parunak	×	×	×	×	×	×	×	×	✓	✓	✓	×	2-D	✓	×
Gaudiano	✓	✓	✓	×	✓	×	×	×	✓	×	×	✓	2-D	✓	×
Sears	✓	×	×	×	×	×	✓	✓	×	×	×	×	3-D	×	×
McLain	✓	×	×	×	×	×	✓	×	✓	×	×	✓	3-D	✓	✓
Reynolds	✓	✓	✓	✓	×	×	✓	×	×	×	×	×	3-D	✓	✓
Espino	×	×	×	×	×	×	×	×	✓	×	×	✓	2-D	✓	✓
Gazi & Passino	×	×	×	×	×	×	✓	×	×	×	×	×	n-space	✓	✓

APPENDIX C

SIMULATION RUNS DATA

TABLE C.1 SIMULATIONS RUNS AND DATA

x	y	OPNET	ESF 0,0	ESF 100,0	ESF 0,100	ESF '-100,0	ESF 0,-100
1230.00	1430.00	0.259925	0.025473	0.023059	0.018394	0.025753	0.016721
1230.00	1432.50	0.255650	0.022238	0.022000	0.035792	0.042611	0.030616
1230.00	1435.00	0.253450	0.088698	0.062612	0.082674	0.157882	0.120429
1230.00	1437.50	0.251575	0.218323	0.223098	0.194032	0.207143	0.182889
1230.00	1440.00	0.250300	0.250000	0.250000	0.250000	0.250000	0.250000
1232.50	1430.00	0.255675	0.046202	0.024641	0.045960	0.027252	0.032954
1232.50	1432.50	0.257575	0.023859	0.022682	0.027771	0.072623	0.035083
1232.50	1435.00	0.253525	0.048340	0.066337	0.046956	0.093193	0.054960
1232.50	1437.50	0.251075	0.168295	0.108647	0.117180	0.134309	0.150661
1232.50	1440.00	0.250325	0.201952	0.196708	0.160430	0.220991	0.162943
1235.00	1430.00	0.253250	0.068954	0.129982	0.061553	0.078109	0.037051
1235.00	1432.50	0.253375	0.041811	0.120105	0.099956	0.045428	0.044220
1235.00	1435.00	0.251950	0.022303	0.039352	0.021435	0.030387	0.049479
1235.00	1437.50	0.251650	0.054111	0.030179	0.030074	0.031729	0.044655
1235.00	1440.00	0.250975	0.048264	0.047241	0.030642	0.014690	0.016178
1237.50	1430.00	0.251525	0.153765	0.208067	0.202255	0.177590	0.220371
1237.50	1432.50	0.251025	0.232105	0.174323	0.114617	0.152757	0.097253
1237.50	1435.00	0.251625	0.027256	0.044443	0.031977	0.030326	0.042517
1237.50	1437.50	0.252250	0.018753	0.017683	0.020343	0.018224	0.012582
1237.50	1440.00	0.251750	0.014691	0.015005	0.012639	0.014408	0.012014
1240.00	1430.00	0.250300	0.250000	0.250000	0.250000	0.250000	0.250000
1240.00	1432.50	0.250350	0.198582	0.242699	0.191555	0.186549	0.170143
1240.00	1435.00	0.251000	0.026820	0.028081	0.028919	0.024870	0.046443
1240.00	1437.50	0.251750	0.019189	0.018986	0.012480	0.015930	0.014320
1240.00	1440.00	0.252825	0.010225	0.010993	0.010871	0.012696	0.007847
1242.50	1430.00	0.249875	0.137386	0.175931	0.167391	0.129998	0.144127
1242.50	1432.50	0.249825	0.075541	0.057870	0.041953	0.064030	0.049215
1242.50	1435.00	0.249575	0.017055	0.022993	0.010176	0.017188	0.012309
1242.50	1437.50	0.250475	0.016902	0.013538	0.011321	0.007622	0.018053
1242.50	1440.00	0.251825	0.007948	0.007989	0.006507	0.007890	0.005532
1245.00	1430.00	0.249600	0.017278	0.031906	0.020032	0.017175	0.019267
1245.00	1432.50	0.249575	0.015843	0.009478	0.014450	0.018119	0.012969
1245.00	1435.00	0.249425	0.009788	0.010321	0.007236	0.010061	0.009760
1245.00	1437.50	0.249200	0.008663	0.009817	0.007729	0.010483	0.008548
1245.00	1440.00	0.248975	0.005603	0.007888	0.008331	0.006629	0.007589
1247.50	1430.00	0.224050	0.012374	0.005503	0.011165	0.005310	0.008467
1247.50	1432.50	0.224695	0.011321	0.008243	0.008245	0.008630	0.009789
1247.50	1435.00	0.224695	0.005475	0.008376	0.006088	0.011564	0.006595
1247.50	1437.50	0.223675	0.006090	0.006526	0.006343	0.006421	0.006317
1247.50	1440.00	0.223425	0.004027	0.004244	0.003430	0.004234	0.005001
1250.00	1430.00	0.223738	0.004425	0.003189	0.007639	0.003792	0.006658
1250.00	1432.50	0.249050	0.004322	0.004555	0.006321	0.005896	0.007145
1250.00	1435.00	0.248975	0.005309	0.002541	0.003272	0.003552	0.005224

Table C.1 – *Continued*

1250.00	1437.50	0.248850	0.004589	0.004831	0.005599	0.003913	0.004966
1250.00	1440.00	0.003200	0.003187	0.005718	0.003901	0.003857	0.003637
1252.50	1430.00	0.252825	0.004822	0.002785	0.001904	0.001799	0.002641
1252.50	1432.50	0.249050	0.002051	0.003731	0.003004	0.004709	0.003174
1252.50	1435.00	0.248825	0.004465	0.003547	0.002896	0.005136	0.003451
1252.50	1437.50	0.005100	0.003577	0.002258	0.003131	0.004276	0.004990
1252.50	1440.00	0.004150	0.003802	0.003326	0.002479	0.002111	0.002099
1255.00	1430.00	0.004350	0.004486	0.001395	0.003267	0.001051	0.002519
1255.00	1432.50	0.252350	0.002432	0.003668	0.003367	0.002643	0.002910
1255.00	1435.00	0.479250	0.001420	0.002556	0.002525	0.001368	0.003530
1255.00	1437.50	0.238350	0.003473	0.002838	0.004295	0.001629	0.002928
1255.00	1440.00	0.237750	0.003469	0.002176	0.001668	0.001695	0.002661
1257.50	1430.00	0.002125	0.000833	0.001902	0.003094	0.002485	0.003914
1257.50	1432.50	0.002050	0.001837	0.002522	0.001710	0.001303	0.001733
1257.50	1435.00	0.011775	0.002146	0.001238	0.001866	0.003113	0.002002
1257.50	1437.50	0.235663	0.002203	0.000579	0.000716	0.002044	0.002557
1257.50	1440.00	0.005775	0.003175	0.000712	0.001463	0.002064	0.001581

Table C.1 – *Continued*

x	y	OPNET	ESF 71,71	ESF '-71,71	ESF '-71,-71	ESF 71,-71
1230.00	1430.00	0.259925	0.025416	0.019244	0.019141	0.011768
1230.00	1432.50	0.255650	0.035117	0.031065	0.018768	0.026071
1230.00	1435.00	0.253450	0.134826	0.078434	0.064975	0.047563
1230.00	1437.50	0.251575	0.200233	0.205569	0.216107	0.227876
1230.00	1440.00	0.250300	0.250000	0.250000	0.250000	0.250000
1232.50	1430.00	0.255675	0.063857	0.025964	0.033717	0.038958
1232.50	1432.50	0.257575	0.018653	0.047846	0.023628	0.044269
1232.50	1435.00	0.253525	0.053065	0.048239	0.036889	0.044434
1232.50	1437.50	0.251075	0.174414	0.109936	0.120205	0.106139
1232.50	1440.00	0.250325	0.176019	0.192279	0.176461	0.159926
1235.00	1430.00	0.253250	0.076604	0.050291	0.086880	0.069310
1235.00	1432.50	0.253375	0.022512	0.043111	0.043178	0.028541
1235.00	1435.00	0.251950	0.052434	0.090187	0.018779	0.042696
1235.00	1437.50	0.251650	0.051709	0.028544	0.057248	0.022153
1235.00	1440.00	0.250975	0.033044	0.043328	0.040065	0.031058
1237.50	1430.00	0.251525	0.211103	0.203612	0.221836	0.199986
1237.50	1432.50	0.251025	0.156358	0.141294	0.189187	0.171377
1237.50	1435.00	0.251625	0.023322	0.024147	0.043785	0.043806
1237.50	1437.50	0.252250	0.017237	0.026241	0.037704	0.021282
1237.50	1440.00	0.251750	0.012584	0.011798	0.014173	0.013180
1240.00	1430.00	0.250300	0.250000	0.250000	0.250000	0.250000
1240.00	1432.50	0.250350	0.187179	0.173938	0.184138	0.201215
1240.00	1435.00	0.251000	0.076890	0.025362	0.037795	0.012056
1240.00	1437.50	0.251750	0.013871	0.019315	0.015437	0.013390
1240.00	1440.00	0.252825	0.013352	0.010978	0.009160	0.007287
1242.50	1430.00	0.249875	0.153743	0.165967	0.159915	0.184644
1242.50	1432.50	0.249825	0.057365	0.150542	0.105316	0.072197
1242.50	1435.00	0.249575	0.016229	0.017270	0.026734	0.020353
1242.50	1437.50	0.250475	0.008952	0.017608	0.011715	0.008787
1242.50	1440.00	0.251825	0.009048	0.008957	0.008999	0.007620
1245.00	1430.00	0.249600	0.017437	0.016187	0.019901	0.017208
1245.00	1432.50	0.249575	0.015723	0.014185	0.034609	0.011371
1245.00	1435.00	0.249425	0.011860	0.012279	0.030154	0.014757
1245.00	1437.50	0.249200	0.009548	0.019485	0.006961	0.008836
1245.00	1440.00	0.248975	0.008172	0.006286	0.004833	0.008653
1247.50	1430.00	0.224050	0.014254	0.008880	0.009130	0.006837
1247.50	1432.50	0.224695	0.006143	0.006676	0.027439	0.008378
1247.50	1435.00	0.224695	0.005529	0.010231	0.006846	0.006356
1247.50	1437.50	0.223675	0.006332	0.005363	0.007885	0.009805
1247.50	1440.00	0.223425	0.003572	0.005253	0.004618	0.004894
1250.00	1430.00	0.223738	0.006776	0.006655	0.006075	0.004933
1250.00	1432.50	0.249050	0.007114	0.006433	0.004767	0.007455
1250.00	1435.00	0.248975	0.004303	0.004991	0.003771	0.004668

Table C.1 – *Continued*

1250.00	1437.50	0.248850	0.006965	0.005968	0.008170	0.005097
1250.00	1440.00	0.003200	0.002296	0.003619	0.003079	0.003094
1252.50	1430.00	0.252825	0.004838	0.004057	0.003845	0.005053
1252.50	1432.50	0.249050	0.003126	0.003058	0.005311	0.004267
1252.50	1435.00	0.248825	0.005557	0.002359	0.004201	0.004035
1252.50	1437.50	0.005100	0.002701	0.002698	0.003122	0.005109
1252.50	1440.00	0.004150	0.002200	0.002117	0.004393	0.005115
1255.00	1430.00	0.004350	0.001507	0.004947	0.007415	0.001690
1255.00	1432.50	0.252350	0.001414	0.002153	0.003686	0.001152
1255.00	1435.00	0.479250	0.002273	0.001696	0.003803	0.002873
1255.00	1437.50	0.238350	0.004165	0.002040	0.001698	0.002198
1255.00	1440.00	0.237750	0.002137	0.002321	0.001818	0.002431
1257.50	1430.00	0.002125	0.001228	0.001689	0.002394	0.001926
1257.50	1432.50	0.002050	0.001559	0.002018	0.001562	0.001876
1257.50	1435.00	0.011775	0.000792	0.002340	0.001972	0.001553
1257.50	1437.50	0.235663	0.001089	0.000883	0.002690	0.001210
1257.50	1440.00	0.005775	0.001843	0.001241	0.001652	0.001465

REFERENCES

[1] R.W. Beard, T.W. McLain, M. Goodrich, “Coordinated target assignment and intercept for unmanned air vehicles,” IEEE Transactions on Robotics and Automation, vol. 18, no. 6, Dec., pp. 911-922, 2002.

[2] Biokin, Ltd., “Critical Values of F-Statistic: An on-line statistical calculator.” [Online document], Oct. 2004, [cited 2004 October 30], Available at http://www.biokin.com/tools/fcrit.html.

[3] J. Ferber, Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence, New York City: Addison Wesley, 1999.

[4] V. Gazi, K. M. Pasino, “Stability analysis of swarms,” IEEE Transactions on Systems, Man and Cybernetics, vol. 48, no. 4, Apr., pp. 539-557, 2003.

[5] P. Gaudiano, B. Shargel, E. Bonabeau, “Control of AUV swarms: what the bugs can teach us,” in Unmanned Unlimited Systems, Technologies, and Operations, San Diego, 2003.

[6] S. A. Glanz, Primer of Biostatistics, New York City: McGraw-Hill, 1992.

[7] The MathWorks, Inc., “MATLAB 7.0.1: The Language of Technical Computing.” [Online document], May 2004, [cited 2004 October 3], Available at http://www.mathworks.com/products/matlab/.

[8] National Aeronautics and Space Administration, “SP-367 Introduction to the Aerodynamics of Flight,” [Online document], Mar. 2000, [Cited 2004 August 14], Available at http://history.nasa.gov/SP-367/contents.htm, 2004.

[9] P.A. Laplante, Dictionary of Computer Science, Engineering and Technology, Chelsea, MI: Lewis Publishers, 2000.

[10] H.V.D. Parunak, R.S. Vander Bok, “Managing emergent behavior in distributed control systems,” in Proceedings of ISATech ‘97, Anaheim, CA: Instrument Society of America, 1997.

[11] H.V.D. Parunak, S.A. Brueckner, J.J. Odell, “Swarming coordination of multiple UAV’s for collaborative sensing,” in Unmanned Unlimited Systems, Technologies, and Operations, San Diego, CA: Publisher, 2003.

[12] J.J. Odell, H.V.D. Parunak, S. Brueckner, J. Sauter, "Temporal aspects of dynamic role assignment," in Proceedings of Workshop on Agent-Oriented Software Engineering, Melbourne, Australia: Publisher, 2003.

[13] OPNET, Inc., "OPNET Modeler: The World's Leading Network Modeling and Simulation Environment," [Online document], May 2004, [cited 2004 Sep. 5], Available at http://www.opnet.com/products/modeler/home.html.

[14] C. Reynolds, "Flocks, herds, and schools: a distributed behavioral model," in Conference Proceedings of ACMSIGGRAPH, Anaheim, CA: ACM, 1987.

[15] SISC 1516-2000, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Framework and Rules.

[16] W.S. Spears, D. Spears, J. Hamann, R. Heil, "Distributed, physics-based control of swarms of vehicles," Autonomous Robots, vol. 17, no. 2, pp. 137-162, 2004.

BIOGRAPHICAL INFORMATION

Jose R. Espino holds a B.S.C.S.E from UTA and has five years of industry experience in software engineering. Jose plans to pursue a Ph.D. in computer science from UTA.