USING AGENT-BASED MODELING TO SEARCH FOR ELUSIVE HIDING TARGETS

THESIS

Jeffrey E. Rucker, Captain, USAF

AFIT/GOR/ENS/06-16

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/06-16

USING AGENT-BASED MODELING TO SEARCH FOR ELUSIVE HIDING TARGETS

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Jeffrey E. Rucker, BA

Captain, USAF

March 2006

AFIT/GOR/ENS/06-16

USING AGENT-BASED MODELING TO SEARCH FOR ELUSIVE HIDING TARGETS

Jeffrey E. Rucker, BA
Captain, USAF

Approved:

////Signed////	6 Mar 06
Dr. J.O. Miller (Chairman)	Date
////Signed////	6 Mar 06
Maj Gary W. Kinney (Member)	Date

Abstract

The SCUD hunt problem that emerged during Operation Desert Storm has become a source of great interest to major commands like Air Combat Command. One of the metrics used to measure the effectiveness of our operations in a SCUD hunt is time to detect and target. We use the agent-based System Effectiveness and Analysis Simulation (SEAS) to provide a simulation environment in which all the elements of a SCUD hunt mission can adequately be modeled. Our Blue Force agents are modeled as multirole fighters, satellites and unmanned aerial vehicles (UAV) with various sensor capabilities. The Red Force agents are modeled as SCUD transporter/erector/launchers (TEL). Particular interest is paid to the effectiveness of various sensors modeled in a set of scenarios following an experimental design. Four measures of performance (MOP) were fashioned to provide insight into the contribution of sensors at work in a SCUD hunt. These MOPs were evaluated to show any statistically significant differences between various mixes of sensors.

Acknowledgments

To my Lord and Savior, Jesus Christ: I lift Your name above all names. You are my hope and joy. Because of Your strength, I have defeated my Goliath.

With pride, I acknowledge Dr. J.O. Miller. His knowledge of combat modeling and the numerous models existing in that field is inspiring. His patience with me as I sought understanding and the applicability of agent-based modeling to this thesis was admirable. Finally, his persistence in pushing me towards excellence will never be forgotten.

The modeling required in this research could not have been accomplished without Mr. Eric Frisco (SPARTA, Inc.). His understanding of the inner-workings of SEAS, not to mention object-oriented programming in general, made the completion of this effort possible. He had an amazing ability to turn my wordy scenario descriptions into viable code. I could never have accomplished this research without him.

To the friends I've developed here at the schoolhouse, thank you for the countless hours of help you selflessly gave up for me. You have demonstrated the true meaning of teamwork and I am proud to serve beside you in our Air Force. In particular, are the Least Squares – friends, you will always be near and dear to me. I could always count on you for prayer support and affirmation that I would make it through.

Finally, I want to personally thank my wife and our three beautiful daughters. Without your support, I would not have made it through this program. You endured the hardships just as much as I did. You stayed up late with me to ensure I finished my work and you got up early to encourage me to keep going. You have been counting down the days to when I will be a part of the family again – the time is almost here.

Table of Contents

List of Figures

List of Tables

USING AGENT-BASED MODELING TO SEARCH FOR ELUSIVE HIDING TARGETS

I. Introduction

Background

During Operation Desert Storm, Saddam Hussein made attempts to weaken coalition forces by launching SCUD missiles against Israel. Though weak in military utility, this act demonstrated Iraq's willingness to use the SCUD as a weapon of terror (if armed with a biochemical warhead) and as a weapon of provocation against smaller regional countries. The U.S. countered by supplying Israel with Patriot anti-missile batteries. Additionally, the U.S. initiated "SCUD Hunt" missions to find the elusive, mobile transporter-erector-launchers (TEL) and defeat the threat before it left the ground.

The Iraqi's had become very skillful at minimizing the time required to move the launchers into location and prepare the SCUD for launch. Contrary to the Soviets, who used the SCUD's predecessor, the R-17, and usually spent 90 minutes to set up, the Iraqi's streamlined their efforts to yield a turn-around time of approximately 30 minutes (Ripley, 1996).

Thirteen years later, during Operation Iraqi Freedom (OIF), planners faced the same threat as before – evasive launchers capable of delivering biochemical payloads. Because the problem wasn't new, the difficulty came in deciding how many and what kind of resources to dedicate to solving the problem on the battlefield (i.e., hunting, finding and destroying a mobile launcher.) This gave rise to the nagging question of how effective the tactics employed to hunt, find and destroy the SCUDS were.

Compiling years of data, several hundred potential launch sites were identified throughout Iraq and resources of various kinds were used to monitor those sites for activity. The resources used include special operations forces, unmanned aerial vehicles (UAV), as well as reconnaissance, surveillance and strike aircraft. Each of these had the mission of detecting and identifying a mobile TEL and, once identified, the kill chain was initiated in order to destroy the TEL before a SCUD could be launched.

The above vignette is the classic Blue versus Red scenario – except that much of it is a real-world happening. The difficulty in modeling such events has long been the obstacle many commands have had to deal with. Recent changes in the modeling and simulation world have produced very powerful, yet simple, models capable of better handling situations as described above.

To begin orienting ourselves toward this research, we must first discuss what a model is. The Air Force Modeling and Simulation Resource Repository defines a model as "[a representation of] the processes performed by a system; for example, a model that represents the software development process as a sequence of phases" (DoD Definition, DoD Directive 5000.59, DoD Publication 5000.59-P). A better definition is offered by Law (2000) who says that models are physical, mathematical or logical representations of systems, possibly simplified in some way, to gain insight into how the system behaves, and even predicts future behavior (Law, 2000: 2). Law also notes that models can fall into several different categories: 1) physical; 2) analytical; or 3) simulations. Physical models deal with real-world objects whereas analytical models produce exact results because they are populated with exact relationships and quantities. Simulations,

however, allow for various input and output parameters, thereby giving the modeler greater flexibility in analysis of the results.

For this research, we will be dealing with combat models – those models that reflect aspects of military operations. As can be seen in Figure 1, combat models exist, in one form or another, at all levels of the modeling pyramid. Each level will provide a different amount of insight into the problem being modeled. Starting with the lowest level, the engineering level, models have a narrow scope; thus, they are very detailed and specific in the questions they address. As you move up the pyramid, the models tend to get larger in size typically due to aggregation from the lower levels. Consequently, these models have wider scopes and contributions but less detail for individual processes. Our focus will primarily be at the Mission Level of the pyramid. Our model is an agent-based model called the System Effectiveness and Analysis Simulation, or SEAS.

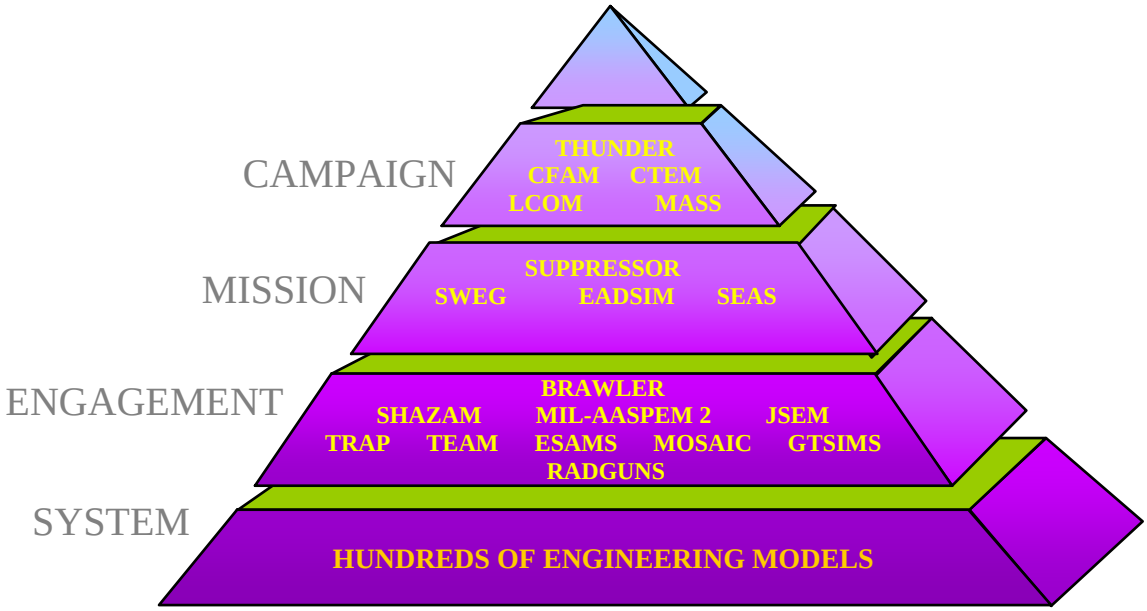

Figure 1. Hierarchy of Models (Miller, 2005)

Problem Statement

Air Combat Command (ACC) is sponsoring this research and supporting our efforts to quantify the effectiveness of "sensors" as modeled in a scenario mimicking the SCUD hunt missions of Operation Desert Storm. Due to their interest in the sensor aspect of the mission, we will model appropriate mission-type sensors on appropriate mission-type airborne platforms, including satellites, and determine the effectiveness of the sensors in shortening the Find-Fix-Track-Target (F2T2) timeline.

Research Objective

The objective of this research is to provide ACC with a flexible analysis tool allowing the capability to analyze trade-offs between various generic sensors and mixes of sensors.

Research Focus

The focus of our research is two-fold. First we will be assessing the value added by incorporating sensor data into the kill-chain timeline in order to shorten the time it takes to detect, engage and remove a potential target. Second, we will be modeling the scenario using SEAS, an agent-based model. SEAS will be used because it has shown itself to be user-friendly and contains the modeling capability required for this effort. Our research will draw primarily on two bodies of work completed to support military analysis. The first resource was completed by the Air Force Research Laboratory Air Vehicles Directorate (AFRL/VACD) and used SEAS to model unmanned aerial vehicles (UAV) involved in hunter/killer missions. The second resource is academic work done by Major Robert Hassler (2005) at the Air Force Institute of Technology (AFIT). His

work used Excel to model the SCUD hunt mission and was supported by ACC. It is believed that the addition of sensor data will bolster the results previously obtained in other research efforts supporting similar research objectives.

Overview of Thesis

Following the introduction, Chapter 2 provides a look at the available literature on agent-based modeling (ABM). The literature review will first take a broad look at the concept of ABM starting with defining what ABM is. The amount of literature in the realm of ABM is voluminous; thus, an overview and some history are provided. Our next step is to familiarize the reader with how ABM has been, and currently is being, used in both the military and industry contexts. It is important to understand that our research supports a military sponsor so much of the literature review will have a military application in its context. As mentioned above, this research will be using SEAS as its modeling platform, so the next section of the literature review will delve into what SEAS is and how it relates to ABM and this research. The literature review next covers the research that has been done supporting similar objectives as our research. Chapter 2 concludes with a section discussing the use of sensors in ABM as it relates to our research objective.

Chapter 3 begins the real work behind this research – the model. Details on what our model does and how our model was built will be provided. The methodology of our research will have direct impact on the results of our model. Thus, care will be taken to describe the methods we will use to accomplish our objective. After completing the work on our model, Chapter 4 will discuss the analysis performed on the results. The post-processing of the data is the pinnacle of any research effort and ours will be no different.

The various measures of effectiveness will be analyzed and the results discussed. Chapter 5 will conclude our research and offer any ideas for future research. Some of these ideas are extensions to our methodology and give insight into new aspects of our problem, or new problems entirely. It is our hope that this research will support others in their pursuit of modeling excellence in operations research.

II. Literature Review

Introduction

Modeling and simulation (M&S) has its roots dating back to the 1940s and its current usage does not lie entirely in military applications. Alongside the military applications, M&S is widely used in the social sciences, biological sciences and environmental sciences. This is important to understand because it is a constant reminder that the mathematics which drive M&S are not restricted to the military, but rather are shared by the world and all its undertakings. The second section of this chapter begins with a definition of agent-based modeling and then discusses its history and some of its current uses with regards to military applications. Following that, the discussion goes into some detail on the specific model being used for this research. It concludes with some limitations and benefits of using agent-based models. The third section of the chapter focuses on some of the specific work that has been accomplished on real-world scenarios. Much of this work is precursory research to our model and is an invaluable resource to our effort. Before concluding Chapter 2 we will touch on sensors – the main thrust of this research. Although the detailed discussion on sensors is reserved for Chapter 3, we will look at some of the general types and uses of sensors in the military environment as well as how sensors are used in the model specific to this research.

Agent-based Modeling

Definitions

Agent-based modeling (ABM) has just about as many definitions as it does authors and researchers, so pinning down a definition can be a difficult task. However,

the designers of one successful ABM define it as an environment in which

> "…complex, real-world systems are modeled as a collection of autonomous decision making entities, called agents. Each agent individually assesses its situation and makes decisions based upon its own set of rules. Agents may execute various behaviors appropriate for the system they represent - for example, sensing, maneuvering, or engaging" (SPARTA, Inc.: Home Page, 2005)

Many characteristics are present in agent-based models, which allow the surfacing of other definitions to take place – at least for the various echelons contained within the realm of agent-based models. The most foundational echelon is the agent itself. In his thesis, Capt Joseph Price states that agent-based modeling "…utilizes small building blocks, known as agents, to represent objects within the system" (Price, 2003:14). Russell and Norvig (1995) describe an agent as being an entity, real or virtual, that perceives and acts in its environment. Rocha (1999) defines agents as "…distinguishable from [their] environment…possess some kind of identity…have some autonomy of action, that they can engage in tasks in an environment without direct external control." Other defining characteristics are an agent's interactions with and amongst other agents (if the simulation is built around multiple agents), the set of rules by which the agent operates, the agent's awareness of self, or even the fact that an agent can have its own resources with which to use in the simulation environment – even against other agents. These constitute just a small portion of the defining characteristics of agent-based models and, in particular, the agent. A formal definition by Ferber (1999) discusses both single agent and multiple-agent systems. An agent is a physical or virtual entity

- that is capable of acting in an environment;
- that can communicate directly with other agents;
- that is driven by a set of tendencies (has autonomy);
- which possesses resources of its own;

- that is capable of perceiving its environment;
- that has only a partial representation of this environment;
- which possesses skills and can offer services;
- that may be able to reproduce itself; and
- whose behavior tends towards satisfying its objectives, taking account of the resources and skills available to it and, depending on its perception, its representations and the communications it receives.

Multi-agent systems must include:

- an environment;
- a set of objects that can be perceived, created, destroyed and modified by the agents;
- an assembly of agents;
- an assembly of relations linking the agents;
- an assembly of operations enabling agents to perceive, produce, consume, transform, and manipulate objects; and
- operators whose task is to represent the application and reaction to these operations.

Another element, which distinguishes ABM from other types of models (i.e. equation-based models like steady-state simulators, discrete event simulators, etc.), and has become an item of great interest in many fields of study, is that of emergent behavior. Emergent behavior is that behavior which is not "pre-programmed" into the agents, but rather surfaces as a result of agent interactions, resource utilization and the changing of an agent's goals. Any one of these can immediately differentiate one agent from another. As agents accomplish their programmed goals and reprioritize other goals due to agent-agent interaction, or agent-environment interaction, they continually morph themselves into something that was not programmed and, thus, exhibit behavior that could not have been postulated. "This ability to possess alternate behaviors enables the agents to adapt over time beyond their initial state and may result in unexpected system behavior" (Price, 2003:16).

As indicated previously, the environment of an ABM is crucial to the operating of all agents therein. The environment is initially constructed by the code written for the

model. Initial model constructs such as terrain, flight paths, deployment locations, and resources such as guns, planes, vehicles, buildings and facilities, are the items an agent has to work with (or own). This type of "ownership" implies that the agent is capable of moving toward and utilizing that resource. However, depending on the type of system you are trying to represent, the use of mobility may be a function you wish not to employ. For example, a model representing forest growth would keep the trees as spatially, immobile agents. The lack of mobility in this case would preclude most emergent behavior due to the fact that the agents don't move and have limited, if any, interaction with each other. If mobility were a desirable characteristic, however, it would give an agent the ability to alter its behavior within the model over time. Thus, a dynamic, flexible operating environment also defines agent-based models and stands it apart from other types of models that are more static in nature.

Overview and History

ABM has a long lineage upon which it stands. Its roots go back to the late 1940s when mathematician John von Neumann came up with the idea of machines that would be able to reproduce. The idea was beyond its time and, at that time, completely impractical. His idea revealed a device that would operate on very simple, yet precisely detailed, instructions in order to make a copy of itself. Having made its clone, the original, or parent, device would then give a copy of the instructions to the duplicate. Now, the duplicate has the capability to reproduce itself, and so on. Von Neumann machines, as they were called, were nearly impossible to construct physically, but a friend urged Von Neumann to draft one on paper. After much contemplation, the result

was a grid on which each cell had the capability to "live" and reproduce itself according to the given instructions. This device became known as cellular automata.

Following cellular automata, ideas and theories began to push the current state of thinking. If cellular automata could simulate a complex process, such as life, using simple, well-defined instructions, what implications might that have in other fields of study? Craig Reynolds (1987) used the emerging theories to develop a model that simulated the flocking of birds, called bird-oids or "boids". This model was the first contemporary use of individual agents. Additionally, it was the first model that revealed the emergent behavior phenomenon. It was expected that the boids would group together as a flock does, but it was not expected that the flock of boids would then reveal related group motion.

Applications of Agent-based Modeling

A quick survey of the literature on ABM will show that its use extends far beyond that of military operations. In fact, it is a driving tool in the simulation and analysis of the social sciences, the biological sciences, oceanography, and traffic management. Though it is not our intent to expound in these areas of study, we wanted to make brief mention of some efforts. Military applications are of primary interest to this research – and there is a plethora of case studies that discuss the applications of ABM to various military issues. It should not come as a surprise that so much information exists since military models, either physical, mathematical or logical, have been used to support force structure (manpower) and whole acquisition processes (or phases contained therein), not to mention their use in education and training – especially training in combat, space operations and pilot training.

Within the past several years, the Air Force Institute of Technology (AFIT) has produced several thesis papers in which ABM was used to simulate, and analyze, events such as German U-boats in the Bay of Biscay (Carl, 2003; Price, 2003). Carl's approach to the historical U-boat event focused on the application of several classic search theory algorithms to a multi-agent model so as to compare the effectiveness of finding high-valued, mobile targets. The foundational search patterns used in his model were borrowed from those in use by the Coast Guard during search and rescue missions, and were employed to develop a "…methodology for empirically quantifying the effects of different search patterns on search efficiency" (Carl, 2003:3). The Defense Modeling and Simulation Office sponsored his work because of their interest in studying the virtue of emergent behavior in agent-based models. Price's thesis work applied game theory to the same historical scenario – U-boats in the Bay of Biscay. An ABM was used because it "…provides the means to model complex systems with non-linearities, by allowing for interactions among independent agents" (Price 2003:ix). The game theory element dovetails nicely within the modeling construct because it focuses on maximizing an objective by providing insight into strategies and outcomes. Is that not the same objective of a military decision maker?

Another thesis of interest was completed by Capt Gregory DeStefano (2004) in which he used ABM to evaluate the transition of information and requirements from the Department of Defense Architectural Framework (DoDAF) into an ABM. The purpose was to show "…how to create an executable model of an actual DoD weapon system described by the DoDAF" (DeStefano, 2004:iv). Additionally, his research aimed at verifying that the translation of data from the DoDAF into the model would accurately

capture measures of effectiveness. All three theses writers found that ABM was the appropriate avenue to accomplish their research goals.

Bullock, McIntyre and Hill (2000) worked, in a combined effort between AFIT and the Air Force Studies and Analysis Agency, to apply ABM to military issues by taking a complex adaptive system approach to ABM in order to capture airpower strategic effects. The authors built the Hierarchal Interactive Theater Model (HITM) to show that Air Force doctrine and airpower theory, when modeled with agents, can provide outcomes consistent with "the expected results of strikes against centers of gravity defined in Air Force doctrine" (Bullock, 2000:1739). Their realistic results demonstrate ABM as a well-suited means of simulating strategic effects at the operational level. They suggest the following as some reasons for using agent-based models:

- To capture force interaction and unpredictability.
- To find out not who won, but why they won. This forces a new level of analysis.
- To determine differing levels of initial conditions and which have the most profound effect on a system. This can lead to identifying centers of gravity.
- It allows for the modeling of an unpredictable system.
- It allows you to focus on the process of conducting military operations and the individual agents that act in conducting those processes.

Contrasting their approach to linear models, they contend that linear models attempt to obtain an analytical solution, and will break a large problem up into smaller pieces. The smaller pieces are then linearized and the individual results then aggregated to obtain a result. The problem with this process is that not all problems have inherent (or natural) points of decomposition. One such problem is war. War is a complex environment, and those engaging in war compose a complex system of actions and interactions. Hence, it can be surmised that an agent-based model would best be able to capture the interactions at various levels because the agents act independently of each

other and are governed by their own set of rules. These rules allow for autonomous agent action and reaction to the environment and other agents' behavior. In other words, each agent has its own Observe, Orient, Decide, Act (OODA) loop that enabled the authors to gain insight to the nature of the entities' interactions and unpredictability. Their bottom line: "…an agent-based modeling approach using OODA loops is an effective way to simulate strategic effects at the operational level of war" (Bullock, 2000:1745).

Other researchers have written volumes on ABM – some in attempts to simply understand what it is; while others use ABM to generate response surfaces from their data in order to describe correlations; and yet others who use agents to gain new insight into existing, and well-worked, problems. Such would be the case for Altenburg, Schlecht and Nygard (2002) writing from North Dakota State University. They worked to maximize mission effectiveness with intelligent munitions deployed against a known number of targets. This problem is not new to the military. Every combatant commander, in every conflict decision, wants to maximize his troops' effectiveness against the enemy. More specifically, when dealing with intelligent munitions, mission effectiveness is crucial due to the cost of the munitions alone; thus, mission effectiveness takes on a whole new importance factor. Their research discussed several characteristics of the agents that can be attributed to many other previous ABMs in the biological and other sciences – namely, the use of avoidance, attraction, following, dispersion, aggregation, homing and flocking. The use of these properties as the basis for the behavior of the agents' sensors is what is interesting. These sensor characteristics are state-dependent and are characteristic of a reactive, behavior-based agent control philosophy. This has direct application to our research in that, our agents will be given

certain rules by which they will operate; however, they will have the ability (under the agent-based modeling construct) to develop emergent behavior. This essentially allows the agents to reprioritize their goals and their approach to obtaining those goals. More will be discussed on this aspect of our research in Chapter three.

Utility has also been found in applying ABM to military issues. For instance, Dr. Michael Young (2000) sees a great future for the use of ABM to help with Command Post Exercises. He states that "Exercises may be conducted at strategic, operational, and tactical levels and are used both to evaluate existing operational plans and develop and test new operational concepts and tactics." Envision, if you will, the same statement but replace the word "exercises" with "models." That is what Dr. Young is getting at – not only does modeling save the Commander money from having to send out "all" his troops to participate in the exercise, but it also saves time – allowing for more rigorous investigation and analysis of the exercise procedures. His opinion on the state of modeling in the Air Force is that the Air Force desires to see more (i.e. improved) modeling capability in the use of more realistic training equipment and the simulation of rank structures (i.e. echelons.) Thus, if the Command Post is modeled as the highest represented unit and all subordinate personnel represented by individual agents, an ABM is a prime candidate to execute this type of model because each agent can now be owned by the higher echelons – in this case, the Command Post.

ABM has also been used in military/industry crossover projects such as the United States Marine Corps Warfighting Laboratory's Project Albert. Project Albert, probably the largest multi-disciplinary modeling effort today, is a Congress-supported initiative aiming to address high-level decision-maker questions. High performance

computers (HPC), and their networking capabilities, are utilized in order to generate the volumes of data the laboratory wants. Project Albert is not actually a model – it is a suite of ABMs. These models include Map-Aware Non-Uniform Automata (MANA), Socrates and Pythagoras. Each of the models contributes a unique perspective to the overall effort while every one of the models is used in the process of data farming and data mining – techniques used, in conjunction with the HPC environment, to produce massive amounts of data. This data is then studied to give insight into non-linearity, intangibles and co-evolution. Non-linearity means that small changes to initial conditions may lead to large changes in expected results. Intangibles are concepts such as leadership, morale and trust. Co-evolution is the internal interaction between agents that results in "thoughts" such as "I think he thinks…." (Widdowson, 2003)

Alongside Project Albert is the Irreducible Semi-Autonomous Adaptive Combat, or ISAAC, computer simulation. ISAAC is not a model, and, according to Illachinski (2000), is a "conceptual playground" of agent-based combat modeling. Because ISAAC is agent-based, a bottom-up approach to combat can be used and emergent behavior can be captured due to interactions at the agent level. This concept ties in nicely with later discussions on the model to be used for our research.

System Effectiveness and Analysis Simulation

The model used for this research is the System Effectiveness and Analysis Simulation (SEAS) developed by SPARTA, Inc. Officially, SEAS is developed and maintained by SPARTA, Inc. for the Space and Missile Systems Center Directorate of Transformation and Development (SMC/TD) and is primarily used to help evaluate the military utility of proposed space systems by quantifying the contributions of space

services - ISR, communications, weather, navigation, strike - to the warfighter in theater-level operations (SPARTA, Inc.: Home Page, 2005). Due to the flexibility in its Tactical Programming Language, SEAS addresses quite well the problem of mobile targets that can evade and surface in order to engage. Additionally, SEAS has a robust communication capability allowing satellites to broadcast targeting information to unmanned aerial vehicles (UAV) which, in turn, transmit the targeting information to the fighter base in order to launch aircraft. These specific qualities are valuable to this research and will be covered in more detail in Chapter 3. For now, consider Figure 2, a graphical representation of SEAS agents in their simulated operational environment, from the TeamSEAS website.

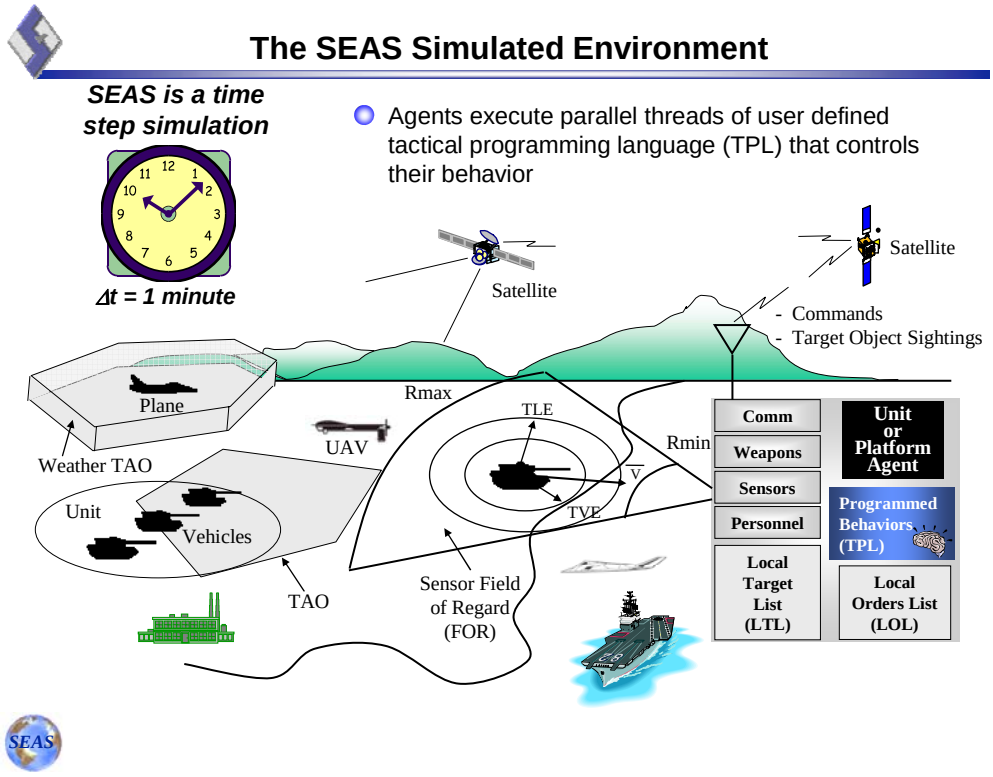

Figure 2. The SEAS Simulated Environment (SPARTA, Inc.: Introduction, 2005)

SEAS will be an integral part of this research because it is an ABM and because of the desired programming capabilities it contains. No model is useful, however, unless you are able to do something with the results it provides; thus, SEAS has an integrated post-processor that will be used to analyze the results. Additionally, SEAS is a verified and validated model accepted as part of the Air Force's Standard Analysis Toolkit. R. Glenn Stockton, Chief, RQD, sees this as an important aspect because, as he notes, "It is possible to do a good, competent analysis using a model that has significant, known, deficiencies. It would be very easy, on the other hand, to do bad analysis using a model that everybody agrees is a good one (if we ever were to get such a model)" (Stockton, 1987).

Limitations and Benefits of Agent-based Models

Thus far, we have seen how ABMs have been used and in what ways they have been applied to some very challenging problems; however, the grass in not always greener on the other side. ABMs do have their drawbacks. The types of models that have been discussed are mathematical models that are played out in simulations. That being the case, one can easily imagine how difficult it is to program agents to have intangibles such as emotions and feelings. Though ABM is also being greatly pursued in the artificial intelligence community, it has not made it to the point where agents make decisions based on how they "feel." Thus, the biggest downfall to agent-based combat models and most models in general, is that they cannot capture random and complex behaviors such as heroism, leadership and morale.

On the positive side, since simulations only require man-hours and computing power as resources, their biggest benefit to the field of knowledge is that they are cost

effective, compared to actually trying to "play" out the simulation in the form of an exercise. It goes without saying that no one is willing to start a war just to find out how the pre-planned strategies are going to pan out.

Also, if you don't like the way the results look you can change the inputs. Knowing that a model exists to give insight and not solutions means that if you change a model's input parameters, then you should expect to get different results. The beauty of a model is that you can make all of these changes without affecting "reality" until you're ready, and have the authority, to do so. Thus, it can be said that models have great impact and utility in giving analysts and decision-makers insight into behaviors encountered in complex scenarios.

The real benefits in ABM are represented by its dynamic operating environment and its ability to model individual agents and allow those agents to develop emergent behavior. Emergent behavior can differ from one scenario to the next, which is what makes ABM so powerful – the more replications of a scenario you run, the more opportunities you have to capture emergent behavior and the more data points you have for great analysis.

ABMs are getting better every day and a growing variety of problems are utilizing its flexibility. ABMs are becoming increasingly more robust as a result of the improvement of faults in other models. They have valuable tools and provide decision makers with more complete information to help make sense out of complex situations. The next section reviews some of the previous research that has been done and is of special interest to our research. These works have added to the contemporary knowledge of military systems and have applied ABM to gain invaluable insight to those systems.

Previous Research

Two certain bodies of work have been done that contain specific aspects of modeling which this research will draw from. Both of these are military studies and, with the addition of our desired measures of effectiveness, provide a solid foundation for our work to begin.

The Air Force Research Laboratory Air Vehicles directorate (AFRL/VACD), has used ABM to assess mission effectiveness on morphing-wing technology enabled UAVs. Additionally, they used ABM to perform a cost-benefit analysis on those UAVs (Brown and Martin, 2005:4). Morphing-wing technology gives an aircraft the ability to "significantly alter its wing shape to accommodate the very different performance demands of varying missions" (Withrow and Sanders, 2005). This kind of technology has been available for years and is, in fact, already in use. Consider the F-14 Tomcat and B-1B Lancer – both have variable sweep wings, which is an early form of morphing technology concepts. What AFRL/VACD has done is recognized ABM's ability to capture the "non-linear behaviors in real-world operations and explicitly model the sensor to shooter chain" (Brown and Martin, 2005:11). This insight, accompanied by the context of their hunter/killer operations scenario, has made it a resource for our research. The morphing-wing UAVs were launched on missions to search for targets and change the shape of their body structure to loiter over the target, engage the target, or dart from one location to another. The results of their work showed that by using an ABM, they were able to reflect sensitivity in many of the key air vehicle input parameters such as fuel burn rates and speeds, as well as varying the flight profiles of the different air vehicles (Brown and Martin, 2005:22). Two characteristics of their work will be of

importance to our work – manipulation of the input parameters for sensitivity analysis and the use of an ABM for a search mission.

Maj Robert Hassler (2005) developed an Excel model to support Air Combat Command's (ACC) desire to know more about the SCUD hunt missions the United States was engaged in during Operation Desert Storm. Specifically, ACC was interested in finding ways to shorten the mobile SCUD launcher kill chain (i.e. the time to search for and identify mobile SCUD launchers, then launch friendly strike aircraft to destroy the launcher before it fires a SCUD). In order to assess the kill chain timeline, Maj Hassler needed to develop a scenario in which the friendly forces performed surveillance missions looking for mobile hidden targets.

The kill chain timeline for a SCUD transporter/erector/launcher (TEL) includes the time for the TEL to be detected, identified, engaged and destroyed – either by the sensor platform or by the strike aircraft queued by the sensor. Maj Hassler roughly estimated the kill chain timeline to equal 10 minutes – three of which are used to adequately detect and identify the TEL and the remaining seven for cueing, engagement and destruction of the TEL. The competing timeline is the 30 minutes required for an Iraqi crew to mobilize, erect and launch a SCUD. These two timelines were compared to compute an effectiveness score. It was determined that as long as each target site is surveyed every twenty minutes, the coalition forces will be able to destroy a target, if active, before it can launch a SCUD.

The focus of Maj Hassler's research was to develop an Excel-based model which automates the many tasks required to map sensors to targets. One way to handle this type of problem is by considering a knapsack problem. In the typical knapsack problem, a

person has a limited volume knapsack but also has a large quantity of items to place inside the knapsack. Since the volume of items is greater than the volume of the knapsack, the person must decide what to take and what to leave. A variation of this problem is the fractional knapsack problem whereby the person can take portions (or fractions) of items and thus maximize what is taken in the knapsack. The fractional knapsack problem can be solved utilizing a greedy algorithm (Cormen, 2003). Though this approach finds an optimal solution to a strict fractional knapsack problem (Cormen, 2003), it is not optimal when employed against variations of the strict fractional knapsack problem. However, applied to these variations solid results are obtained which give good insights and can be easily executed (Jensen and Bard, 2003).

Translated into knapsack terminology, each sensor is a knapsack with its period of activity, or VUL, as the capacity of the knapsack. The targets are the items to be placed in the knapsack. Remaining knapsack capacity is represented by the cost of traveling to a target and surveying it. Surveying a target is the value-added benefit of adding that item to the knapsack. So if a target is to be surveyed, it must have a value-added benefit which is greater than the cost of getting to and surveying it.

Several assumptions are applied to the model. First, each sensor can only survey one target at a time. Multiple targets can be surveyed at once but only if each target is within the sensor's field of regard. Second, in order for a target to be surveyed, the sensor must be at the target's exact location. Third, the flight path between targets is unobstructed by terrain, enemy threat systems, etc. Fourth, the time required to survey a target and the time required to destroy a target are constant. Fifth, each sensor travels at a

user-defined constant speed. Sixth, any target may be revisited at any time by any active sensor.

When all was said and done, Maj Hassler had an Excel model which accomplished two main goals: first, it established flight paths which would, in part, optimize (or maximize) the sensor's contribution to the search effort; second, it calculated an effectiveness score which helped determine how well the sensor set covered the target set.

We now have the necessary tools to begin taking on the next iteration of modeling this scenario. Like AFRL/VACD, we will be using SEAS to model a "search" mission. Like Maj Hassler, we will be "searching" for hiding targets. However, our work will now incorporate various sensor platforms into the model to provide another level of detail and aid in better analysis. Sensors onboard aircraft can be used to detect and surveil elusive, hiding targets. Complementing those would be sensors on satellites and airborne reconnaissance platforms. The next section will detail the types of sensors our work will employ.

Description and Modeling of Sensors

Introduction

Mankind has always employed the use of "sensors" to gain an advantage over an opponent. Early history shows us that scouts were sent out to find and report enemy locations. More recently, balloons were fitted with cameras and sent aloft to take pictures of the enemy's whereabouts. The future of modern-day sensors has been secured due to the technology upon which they are based. As technology advances and allows us to utilize sensors in a new fashion, or to a better level of detail, we will be able to collect the

data required to make timely decisions. As Mark Hewish (2002) states in *The All-seeing Eye Above*, "The challenge is to distil information from the plethora of data, and provide it to warfighters in time for them to act appropriately." Thus, it is an understatement to say that sensors play an important part in military applications – most notably to detect targets, though, sensors are also capable of notifying that they have been detected. Detection of targets is particularly important for this research. At this point the term "sensor" has only a generic meaning. For instance, an aircraft, UAV or a satellite can each be regarded as a sensor – as long as it contains the capability to detect (either actively or passively) an object. Or, a sensor can be just that – a device, contained on another platform, used to detect the target which the platform is looking for. To gain an initial understanding of the use of sensors in SEAS refer to Figure 3 which gives an overview of sensors as devices used by agents.

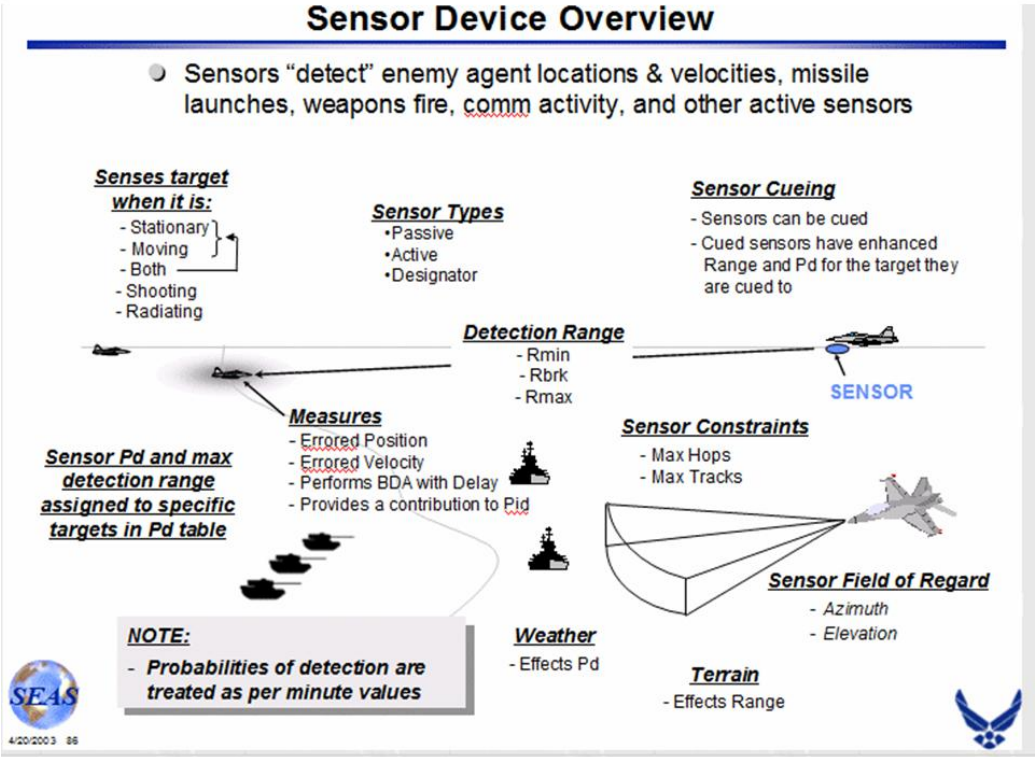

Figure 3. Sensor Device Overview (SPARTA, Inc.: Sensor, 2005)

Onboard Sensors

An onboard sensor is the second type of sensor discussed above. This can include, but is not limited to, synthetic aperture radars (SAR) and electro-optical/infrared (EO/IO) devices, to name just two. As discussed in email correspondence with Air Combat Command (Berg, 2005), there are certain aspects of sensor capability that are desired for this research: Azimuth and elevation limits (i.e. the total field of view, not the field of regard), as well as range, probability of identification (Pid) and probability of detection (Pd). SEAS will allow sensor parameters to be set for each specific entity to which it is attached. Thus, the airplanes can have different onboard sensors compared to the UAVs and the satellites, and the UAVs can have different onboard sensors compared to the airplanes and satellites, etc. Figure 4 is a screenshot from SEAS v3.5 showing the sensor input box. It is in this location that each sensor to be used, by the various entities, is named and its specific parameters are input.

Syntax	Description
Sensor "sensor_name"	
Min_Range [x]	Default = 0.0(km) The minimum detection range
Break_Range [x]	Default = 0.0(km) The range where the Pd starts to drop off linearly (see below)
Max_Range [x]	Default = 4000(km) Range at which Pd goes to zero. Can be set per target in Pd table.
Az_Width [x]	Default = 360(deg) Azimuth field of regard for sensor
El_Min [n]	Default = -90(deg) Minimum elevation angle for sensor field of regard
El_Max [x]	Default = 90(deg) Maximum elevation angle for sensor field of regard
TLE [x]	Default = 20(m) Gaussian one sigma target location error
TVE [x]	Default = 0(m/min) Gaussian one sigma target velocity error. When 0, no velocity reported.
TLE_Mode [n]	Default = 1 A flag to make TLE and TVE range dependent (= 1) or not (= 0).
TLE_Aspect [n]	Default = 1 Ratio of cross range TLE to down range TLE.
TVE_Aspect [n]	Default = 1 Ratio of cross range TVE to down range TVE.
MTI [mode]	Default = 0 Moving target mode. 0=detects moving and stationary, 1=moving only, 2=stationary only
MDV [x]	Default = 0 Minimum detectable target velocity (km/hr). Ignored if MTI not = 1
PId [x]	Default = 1 Contribution this sensor makes to probability of target identification (0-1).
BDA_Prob [x]	Default = 1.0 Probability of correct BDA (type 1 error only)
BDA_Time [x]	Default = 0.0(min) Time to perform BDA
Cued [flag]	Default = 0(false) Can this sensor use a cue to improve detection range & Pd
Cue_Range [x]	Default = 40000(km) Maximum range for cued detection
Cue_Quality [x]	Default = 0.0 Pd improvement for qued detection. Pd = 1-(1-Pd_old)*(1-Cue_Quality)
Daylight_Only [flag]	Default = 0(false)
Active [flag]	Default = 0(false) Does this sensor emit radiation that can be detected by a "Detects_Active=1" sensor
Detects_Active [flag]	Default = 0(false) Does this sensor detect emitters. Pd table must specify the active sensor(s) or comm devices it detects
Detects_Launches [flag]	Default = 0(false) Does this sensor detect missile launches. Pd table must specify the missile(s) it detects
Detects_Weapons [flag]	Default = 0(false) Does this sensor detect weapons fire. Pd table must specify the weapon(s) it detects
Designator [time]	Default = 0(min) How many min should a detected target be illuminated (kept in track regardless of Pd)
Max_Hops [x]	Default = 8 How many comm hops this sensor's sightings can make. Zero will keep it on the local platform.
Max_Tracks [x]	Default = 0 How many targets can be kept in track. Zero = infinite.
Value [x]	Default = 100000
IFF_Prob [x]	Default = 0.0 Probability of misidentifying a friendly as an enemy
Draw [flag]	Default = 1(true) Off(0) On(1) display of sensor's field of regard. (2) adds a dotted line detection display.
Terrain_DB "dbname"	Terrain affects the sensor's range
Weather_DB "dbname"	Weather affects the sensor's Pd
End	

Figure 4. Sensor Input Screen (SPARTA, Inc.: Sensor, 2005)

Types of Sensors

Sensors are not all the same, nor do they all have the same mission. Though they all serve to exploit the distinct parts of the electromagnetic spectrum, various sensors will perform their mission in different ways, on different platforms, and in different

environments (i.e. space, ground, clear weather, inclimate weather, etc.) Though it goes without saying, a sensor will perform optimally if employed in the environment for which it was intended. This proves interesting when you consider that the same sensor, say, a SAR, can be employed on both an airplane and a satellite. So we can capitalize on a sensor's strengths to support/complement other "players" in the overall battle-space. Hewish (2002) remarks on this in *The All-seeing Eye Above*:

> Sensors with a wide field of view can cue those of higher resolution, but with a narrower gaze, to areas of interest. For example, a SAR aboard a high-altitude loitering platform may pick out objects for further examination by an EO/IR package. A radar operating in GMTI [ground moving-target-indication] mode can follow a **mobile** missile launcher until it disappears beneath the tree canopy, then hand it over to a FOPEN [foliage-penetration] sensor. Radio-frequency receivers on a fighter aircraft may **detect** and identify targets for prosecution using the on-board fire-control radar or EO/IR targeting systems. Complementary sensors may be installed aboard different platforms, linked by high-capacity data networks, or grouped together on the same vehicle.

Referring to satellites, he writes:

> Space-based radars offer several advantages, including virtual invulnerability to attack and the ability (assuming a large enough constellation) to conduct continuous surveillance. They can look down into valleys, and observe the deep battle that is beyond the radar horizon of platforms such as Joint STARS or MC2A [multi-Sensor Command and Control Aircraft.]

Jane's International Defence Review was referenced at http://www.janes.com for the types of sensors onboard a Predator UAV and an F-16 Fighting Falcon to be used in the SEAS model so as to best mimic reality.

For the Predator, the Jane's database entitled "*GA-ASI MQ-1 and RQ-1 Predator*" lists the following sensor information:

- fixed, nose-mounted daylight color TV camera to aid remote piloting
- L-3 Wescam 14TS Skyball sensor in the undernose ball turret
- Northrup Grumman AN/ZPQ-1 Tactical Endurance Synthetic Aperture Radar (TESAR) with face-plate antenna in undernose bay
- Raytheon AN/AAS-52 Multispectral Targeting System (MTS) which combines a third-generation infrared sensor, wide-field-of-view TV, and laser range-finding designation and targeting features

For the F-16 Fighting Falcon, the Jane's database entitled "*Lockheed Martin (General Dynamics) F-16 Fighting Falcon*" lists the following critical sensor information:

- AN/APG-68(V)9 high-resolution Synthetic Aperture Radar – offers better air-to-air range and all-weather precision targeting while improving resolution, track accuracy and reliability

For the satellite sensors, I will be modeling a generic satellite that will have the search and detect capabilities that a Synthetic Aperture Radar offers as well as the ability to detect moving targets with a GMTI sensor. Thus, no specific satellite constellation will be modeled and, in order to provide sufficient coverage, a constellation of nine satellites will be modeled.

Conclusion

This chapter has covered a lot of material, beginning with the definition of agent-based modeling. As was seen, there are numerous definitions based on the purpose of the model. We then moved on to view some of the rich history of agent-based modeling in many areas of study (biology, medicine, social and environmental sciences, etc.) to include our main focus on military applications. The first section concluded with some of the limitations and benefits of modeling and simulation.

The next section of material we covered provided a basis for the requirements needed for this research. Namely, what sensors are and how they will be used in our

model. Sensors are, generically, any devices that can detect, either actively or passively, a mobile or stationary target. For this effort, a sensor will be a device owned by an independent agent within the model and used for the same purposes – to detect a target. These sensors will range from aircraft to UAVs to an on-orbit satellite. In order to capture results similar to real-world data, every attempt will be made to model the sensors with real-world parameters from sound unclassified military data sources like Jane's International Defence Review and the Federation of American Scientists (FAS).

Finally, we reviewed previous work in this area of research. The Morphing-wing Technology work done by AFRL/VACD applied the agent model SEAS to UAVs performing a search and kill mission. Maj Robert Hassler did his graduate research project on modeling SCUD hunt missions as they occurred in Operation Desert Storm. Some time was spent discussing his Excel model so as to gain insight into his problem statement and methodology. The section concluded with the results of his model which were presented, with great satisfaction, to Air Combat Command.

The next chapter, Chapter 3, will discuss the methodology behind the building of our model. We will combine the SEAS work done by AFRL/VACD and the fractional knapsack search problem worked on by Maj Hassler, to create a SEAS model that explores the value added by including sensor capabilities in the search for elusive targets.

III. Methodology

Introduction

We now begin to explore the process of selecting and building a combat scenario to be used in the System Effectiveness Analysis Simulation (SEAS) agent-based model. Following the scenario background underpinning this research, an explanation of the code composition will pave the way for our discussion on the Warfile itself. Before that, however, the Measures of Effectiveness will be laid out. Prior to concluding, a description will be given, in detail, of both the Red and Blue forces that will be simulated.

The term "scenario" describes the who, what, when, where, why and how of what will be modeled; however, in the SEAS lexicon, what will be built is a Warfile. SEAS is a modeling environment where agent entities process the commands programmed in the Warfile. Thus, the Warfile contains the "scenario" written in tactical programming language (TPL), or "code", and then simulated by the agents. Once complete, the Warfile will be used for researching how agent-based models can provide valuable insight and information on the role sensors play in searching for elusive, hiding targets. This information will be passed on to Air Combat Command (ACC) which is sponsoring our research.

Scenario Background

Before getting into the actual process of selecting and building a model, we must first understand the who, what, when, where, why and how behind our Warfile. During Operation Desert Storm, Saddam Hussein made attempts to weaken coalition forces by

launching SCUD missiles against Israel. Though weak in military utility, this act demonstrated Iraq's willingness to use the SCUD as a weapon of terror (if armed with a biochemical warhead) and as a weapon of provocation against smaller regional countries. The U.S. countered by supplying Israel with Patriot anti-missile batteries. Additionally, the U.S. initiated "SCUD Hunt" missions to find the elusive, mobile transporter-erector-launchers (TEL) and defeat the threat before it left the ground.

The Iraqis had become very skillful at minimizing the time required to move the launchers into location and prepare the SCUD for launch. Contrary to the Soviets, who used the SCUD's predecessor, the R-17, and usually spent 90 minutes to set up, the Iraqi's streamlined their efforts to yield a turn-around time of approximately 30 minutes.

Thirteen years later, during Operation Iraqi Freedom (OIF), planners faced the same threat as before – evasive launchers capable of delivering biochemical payloads. Because the problem wasn't new, the difficulty came in deciding how many and what kind of resources to dedicate to solving the problem on the battlefield (i.e., hunting, finding and destroying a mobile launcher) – a decision we are still trying to make today.

ACC is sponsoring this research and wishes to gain insight into how sensors affect the SCUD hunt targeting process and the kill chain. Without sensors onboard aircraft, satellites, etc., the acquisition and identification of SCUD targets would take longer thereby lengthening the time required to complete the kill chain. Hence, our research will focus on the “detecting” role sensors play in the SCUD hunt mission. In terms of Blue and Red, the Red force agents will be mobile SCUD TELs that hide, so as not to be detected, and then resurface to fire a SCUD. The Blue force agents will be

modeled as multi-role fighters, UAVs and satellites all trying to detect the SCUD TELs and relay that detection to the appropriate agent for engagement.

SEAS "UseCases"

Our SEAS Warfile will be composed using two popular methods: 1) Writing the file from scratch and 2) Utilizing (with proper citation) well-established code. The use of both methods will provide the benefit of tailoring the Warfile code to contain exactly what is needed and desired, while utilizing other code allows the flexibility of adding key agent components, from trusted sources, and saves the overall time required to develop a mission-level combat model.

Referring to the former method above, the "UseCases" which are included in any version of SEAS software will be an excellent point of reference for code. Some of the various "UseCases" which I will reference are:

- *AC_Avoid_SAMS*
- *AC_FomationFlying*
- *Bugs*
- *GMTI_Qs_SAR*
- *MovingTAO*
- *SimpleUAV*
- *Terrain&Weather*

Many of these include segments of code that directly apply to this research. For instance, *GMTI_Qs_SAR* contains pertinent satellite information that is difficult to construct, or obtain – orbits, sensor parameters, etc.

Air Force Research Laboratory

The Air Force Research Laboratory Air Vehicles Directorate (AFRL/VACD) has also done work that is useful to my research. Their work on modeling morphing technology, as applied to unmanned aerial vehicles (UAV), focuses on concepts that are

also part of my model. Mainly, the UAVs that AFRL/VACD (Brown and Martin, 2005:15) modeled contained segments of their missions where they loitered waiting for a target; segments where they tracked a target; and segments where they left one target in search of another. Our model will contain each of these but for both UAVs and multi-role fighters. For example, in our model, UAVs will be out patrolling a tactical area of operations (TAO), when they detect a target (either by their own sensor or queued by the Blue Air Operations Center (AOC)) they will precede to the target area for further investigation. Once the target is identified, the commanding base will be notified in-turn notifying the patrolling multi-role fighters. The multi-role fighters will then proceed to the target location. The concepts that AFRL/VACD worked on will contribute to the knowledge base used in our model.

Although both methods of code composition will contribute their share to this research, the majority of the underlying scenario code will be original effort. This is mainly due to the time it takes to strip useful code out of another body of work and then apply it to our code in a seamless way. It became apparent, early on, that building the scenario from the bottom up would be faster and easier than relying too heavily on previously developed code. After getting a substantial start on our scenario, it was necessary to call the SEAS developers for guidance in the code. SPARTA, Inc.'s Eric Frisco aided in the development of this model and has become a valuable resource for this research. His knowledge and expertise in SEAS, let alone modeling and simulation in general, have helped streamline the code and have clarified the methodologies we need in place to effectively capture our Measures of Performance.

Measures of Performance

In order to establish a common understanding, some definitions must be given of the terminology that will play an important role in the following text. First, we are attempting to capture data on specific performance parameters of our modeled system. Thus, we use the term Measure of Performance (MOP) to identify said parameters. These MOPs are very detailed and come in the form of a statement because they can aggregate up to support higher-level measures of effectiveness and answer critical operational issues. An example of one of our MOPs is "Find the average number of targets destroyed [across all runs] before being able to fire."

After conversations with ACC, several MOPs were established. As with all models, obtaining the data required to develop the MOPs is a matter of output. Some of our MOP data will come from normal SEAS output files and will be analyzed using the SEAS Postprocessor – which is an Excel-based macro – while other data may require unique experimental design and special code to capture it.

ACC's main question is: Can the probability of identification (P_{id}) and probability of detection (P_d) be developed for each type of sensor modeled? Following on that, ACC would like to know 1) if P_{id} and P_d make a difference to aircraft routing, and 2) if degradation can be captured due to P_{id} and P_d? That is, if the blue forces cover the operational area 100% of the time, 24/7, does the application of P_{id} and P_d values degrade the 100% coverage down to, say, 40% mission success of finding the targets of interest? The complexity of these questions will lead us to develop our model to the best extent possible in order to provide insight. Some of this insight begins with understanding the inputs required by SEAS. For instance, probability of detection, P_d , is a direct input

required by SEAS in order to reference a P_d table; however, probability of identification, P_{id}, is not a direct input and its derivation is beyond the scope of this research. Thus, unless we consider a system of systems approach, the P_d's of the sensors modeled will remain as required inputs and not derived. As for aircraft routing, our Blue Force CONOPS is the driving force behind the engagement of the Red Forces. In other words, even though we have given each *F-16C* a P_d of 100% and P_k of 100% against the *SCUD_TEL*s, the *F-16C*s are not looking for those targets against which they have a better P_d or P_k, they are simply flying CAP until rerouted to target locations broadcast by the AOC. Thus, their flight paths are completely stochastic and will change dynamically as their local target lists are updated. Moreover, it should be noted that giving our sensors a P_d and P_k both equal to 100% is, in essence, defining our sensors as definite range law devices. A definite range law device is one where a target cannot be detected outside the device's maximum detection range because no probability exists for that instance. However, any target within the device's detection range – to include distance, azimuth and elevation – will be detected with certainty. Recall from Figure 2.3, the sensors we apply in SEAS have as inputs a minimum detection range, a break range, a maximum detection range, azimuth width, elevation minimum and elevation maximum. Any target that falls within bounds defined by those six sensor parameters will be detected with probability = 1.

The process of identifying a target can vary by situation. Taylor (2000) describes the target identification process as occurring in 4 different phases.

1. **detection** (target detected at such a level of resolution/discrimination that observer can distinguish an object of military interest that is foreign to the background in its field of view, e.g. distinguish a vehicle from a bush)

2. **aimpoint** (target detected at such a level of resolution/discrimination that observer can distinguish an object by its class, e.g. a tracked vehicle versus a helicopter or a wheeled vehicle; observer can thus establish an aimpoint on the object)
3. **recognition** (observer can categorize targets discriminated at aimpoint within a given class, e.g. recognize a tank versus an armored personnel carrier in the tracked vehicle class)
4. **identification** (observer can distinguish between specific recognized target models, e.g. a T-72 tank versus a T-80) (Taylor, 2000: 929)

Sensors in SEAS process targets at the "identification" level – that is, the sensors detect every target against which they have a P_d value (in our case $P_d = 1$) and P_{id} is assumed perfect. This is due to each agent having a long name associated with it and which uniquely identifies it. The long name includes the agent's parentage all the way back to the force. For instance, a *SCUD_TEL* is a Red force agent, but when observing the Sensor Detection file to see which *SCUD_TEL* a satellite detected at a particular time you would see the *SCUD_TEL*'s long name appearing in this format: Side.Force.Unit#.Agent#. For example, *Red.RedC3I.Red_HQ#1.SCUD_TEL#14*.

In addition to P_{id} and P_d, ACC would like to know: What is the time interval it takes to make an observation? This should include the time it takes the sensor to complete its mission (i.e. Find-Fix-Track-Target (F2T2) and take a picture) and the time for the operator to read and interpret that mission data to make a decision as to whether an airborne asset should be tasked or not. Although we are not modeling the human-in-the-loop aspect behind this question, our model gives insight into the average detection times.

Narrowing this wide range of MOPs down to a reasonable number that will provide excellent insight and a solid foundation for future work, we will consider following MOPs:

1. The number of initial target detections, by any sensor, within two hours of the first target detection.
2. The proportion of targets destroyed before being able to fire.
3. The proportion of targets destroyed by the end of the 24-hour period of operations.
4. Which Blue Force agent detected which target and at what time did the detection occur?

Table 1 summarizes the MOPs we are collecting data for. The column headings provide a very brief description of the design of experiments – please refer to the "Model Design" section for more discussion. Chapter Four contains the results of our analysis on each of the MOPs; as well, the standard descriptive statistics (means, confidence intervals, etc.) are presented to support the insight gained from our analysis.

Table 1 Summary of the MOPs

	Baseline Case (All Agents)	Test Case 1 (No *F-16C*s)	Test Case 2 (No UAVs)	Test Case 3 (No Satellites)
MOP 1	X	X	X	X
MOP 2	X		X	X
MOP 3	X		X	X
MOP 4	X		X	

Verification of the Model

Verification of our model will be accomplished as the model development progresses. Verification of a model involves ensuring that the model coding and structure are correct; i.e. did we build the model correctly (Banks, 2005: 354)? The driving force behind our model verification is the input from subject matter experts (SME). The SEAS model developer – SPARTA, Inc. – has been crucial in filling that role. When new code is added to the model, they have made themselves available to view the addition and make recommendations. In addition to helping with the model

code, they have contributed their knowledge of sensor capabilities and how SEAS processes information.

Validation of the Model

Validation of our model is the process of making sure the model represents the simulated system as accurately as possible and that all the correct pieces to answer the questions being asked are present in the model. According to Banks et al. (2005: 354), the goal of the validation process is twofold:

1. To produce a model that represents true system behavior closely enough for the model to be used as a substitute for the actual system for the purpose of experimenting with the system, analyzing system behavior, and predicting system performance;
2. To increase to an acceptable level the credibility of the model, so that the model will be used by managers and other decision makers.

In other words, did we build the correct model? The CONOPS will play a significant role in answering that question. Thus, we employed guidance from an experienced F-16 pilot whose operational experience includes being an instructor pilot qualified in the Block 42/52/50 and flying combat time in the SEAD/Air Superiority missions. Passing a pilot's "gut check" is a major step toward the seal of approval for our model. Once all agents have been programmed and the CONOPS are in place, we will run the model for a number of operators, whose Air Force pilot experience qualifies them as experts, and see if they agree to the tactics and CONOPS of the agents – as witnessed while the simulation plays out.

Model Design

The Warfile for our model is designed in two major stages. The first stage involves a small test case to verify all communication and sensor-to-shooter links are working. Additionally, all logic is scrutinized making sure that the planes fly at the appropriate times, the satellites are passing overhead, the UAVs are following the correct TAOs, etc. Once the test case is working properly, the second stage contains numerous cases where the number of targets remains constant (see Red Forces discussion below), but the combinations of multi-role fighters, UAVs and satellites are changed to perform sensitivity analysis on the force structure. Since the Warfile uses UAVs, multi-role fighters and satellites as its agents, the study cases are designed selecting the agents in pairs. The number of combinations beyond the baseline case is then three. The following list is the breakout of those three combinations representing the various force structures, starting with the baseline case:

- Baseline case = UAVs, Multi-role fighters, Satellites
- 1st Case = Satellites, UAVs, No Multi-role fighters
- 2nd Case = Satellites, Multi-role fighters, No UAVs
- 3rd Case = UAVs, Multi-role fighters, No Satellites

Another option we initially considered for our model design is to leave all three blue platforms (satellites, UAVs and planes) in each case, but to then vary the number of each type. This approach allows us to get a better understanding of the contribution of each agent type and how that contribution affects the MOPs. After some preliminary design analysis on this option, however, we concluded our current outlined approach provides a sufficient level of fidelity and insight given our time constraints.

Additionally, two different sensor types are applied as devices to the agent platforms. The following design matrix, Table 2, identifies what sensor is assigned to

which agent. Note that the sensor called "PilotEyes" is our representation of giving the *F-16C* agent the ability to get a visual on a target. This was added after conversations with Mr. Dave Berg (2006) in which he indicated that an attack aircraft pilot would be able to discriminate a real target from a decoy when the target is within his visual range.

Table 2. Platform/Sensor Capability Matrix

	Satellites	UAV	*F-16C*
GMTI	X	X	
SAR	X	X	X
PilotEyes			X

Table 2 needs just a little more clarification. The capability of the sensors to detect a moving target is handled explicitly as an input parameter on each sensor. We chose this approach to the sensors to allow for flexibility in our design; thus, we did not design such detailed sensors that the data output would be overwhelming. Figure 5 illustrates the sensor onboard each of the *F-16C*s. The arrows indicate the important input parameters on this sensor making it unique from the other sensors modeled. Although it is built from the generic sensor template, the design points identified make is suitable for the mission that we have applied it to.

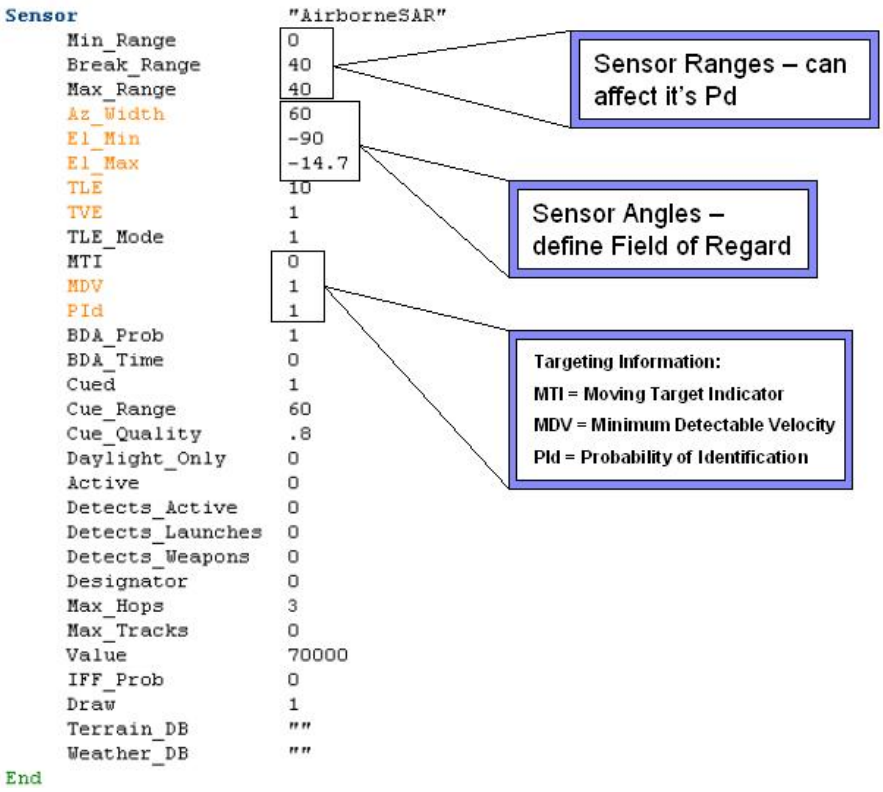

Figure 5. *F-16C* onboard SAR sensor

Force Structures: The Red Forces

We have now arrived at the point where we need to layout what each of the forces (Blue, Red and Sim_Manager) looks like and how they are structured. Since the Sim_Manager is the smallest force, let us start with it. The Sim_Manager has no subordinate agents and is not a combat force like the Red and Blue forces. It is used as a programming construct for ease and traceability by placing all the code for generating the Red force's random starting locations within its domain. The Red force is a very simple structure: it contains one parent agent and subordinate agents – known as children – see Figure 6. The parent agent is the *Red_HQ* and is the Command, Control, Communications and Information (C3I) hub for the Red forces. Due to the nature of this scenario, the *Red_HQ* is a very benign agent. It contains no communication links to its

children and contains no weapons in and of itself. The *Red_HQ* deploys to Baghdad at the start of the simulation.

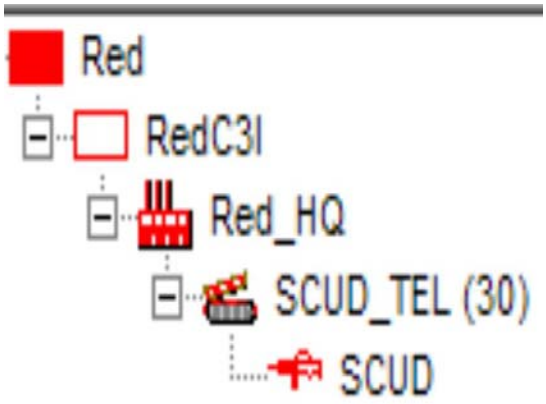

Figure 6. Red Force's Command Structure

The children of the *Red_HQ* are the *SCUD_TEL* agents, and are by far the most complex agents in this Warfile. The complexity of the *SCUD_TEL*s is due to the numerous targets with all with their mobile capability to evade detection. According to ACC, the real-world problem has two kinds of targets: real and decoy (Berg, 2006). A typical scenario would include a few real targets and perhaps ten to twenty times more decoys. For this initial research, we only model real targets (30 for our scenarios), but have included the logic in our code to model decoys in future efforts. Hence, the real *SCUD_TEL*s are those that move and fire at the Blue Air Operations Center (named *B_AOC*). Figure 7 shows the *Location_Generator* TPL from our code that governs the initial placement of the *SCUD_TEL*s. The random time of appearance and placement of the *SCUD_TEL* agents within the area of operations (AOO) both occur according to a Uniform distribution; additionally, the first line of Figure 7 states that this random generation is only to occur on the first Monte Carlo run. That is, over all the runs performed on any test case, the starting locations of the *SCUD_TEL*s are the same every

time. This is due to the location generator writing the initial locations to an external text file named *RealTargetLocs.txt* and then reading those locations at the beginning of every subsequent run.

```
!! only do this on first Monte Carlo trial
    If war->Iteration == 1 & $REAL_TARGET_FILE == 0
        Open Write, "RealTargetLocs.txt"                !! open external text file to write to
        While i < $NUM_REAL_TARGETS
            While Inside("Iraq", rnloc) != 1
                !!rnlong = 43.5  !! central longitude
                !!rnlat = 33     !! central latitude
                !!rnloc = location(rnlong, rnlat, 0) + 500*uniform() * vector(360*uniform())
                rnlong = 35.6 + 16.6*uniform()       !! theater lonmin = 35.6, lonmax = 52.2,
                                                                delta = 16.6
                rnlat = 28.8 + 8.9*uniform()         !! theater latmin = 28.8, latmax = 37.7,
                                                                delta = 8.9
                rnloc = location(rnlong, rnlat, 0) !!+ 500*uniform() * vector(360*uniform())
            EndWhile 1
            SCUDArray[i] = rnloc
            rnloc = location(0,0,0)                  !! reset rnloc to 0
            Write "RealTargetLocs.txt", SCUDArray[i]        !! write location to external file
            i = i + 1
        EndWhile 1
        Close "RealTargetLocs.txt"
    Else
        Open Read, "RealTargetLocs.txt"     !! open target file
        j = 0
        While j < $NUM_REAL_TARGETS
            Read "RealTargetLocs.txt", SCUDArray[j]      !! read target locations from file
            j = j + 1
        EndWhile 1
        Close "RealTargetLocs.txt"
    EndIf
```

Figure 7. Location Generator Tactical Programming Language

Once the *SCUD_TEL* locations are generated and the agents appear in the AOO, the active *SCUD_TEL*s then begin moving a random direction and distance, according to another Uniform distribution, along *SCUDArray* – an array created to hold the directions and distances for each *SCUD_TEL* – as can be seen in Figure 8. Their movement is coded at the agent level not at the parent, or *Red_HQ*, level. Often times, the behavior of subordinate agents is captured, or controlled, at the parent-agent level and it filters down

to the children; however, for programming ease, it was decided that each *SCUD_TEL* should control its own random movement. The mobile *SCUD_TEL*s are equipped with SCUD missiles, are programmed to stop for at least 30 minutes (long enough to setup and launch), and they fire at the Blue Base only once during each run. The SCUDs have no effect on the Blue Base and are there only for completeness and to ensure that we don't miss an opportunity to catch data, for example, on a TEL that was not targeted and fired a lethal weapon.

```
While me->Status == 2
!!      CONOPS for this agent:
!!      Hide for a random time from 4 - 8 hours, then come out of hiding
!!      Pick a random direction and distance and move there
!!      When at new location, setup launcher (30-60 min delay), shoot at blue target,
!!      tear down launcher (15-30 min delay)
!!      Pick a random direction and distance and move there
!!      Repeat cycle
        me->Hide = 1                        !! start out hidden (Hide 1)
        hideDelay = 240 + 240*uniform()     !! choose a random hide delay time
                                            !! from 4 - 8 hours (240 - 480 mins)
        Delay hideDelay                     !! wait the hide delay time before doing anything
        me->Hide = 0                        !! come out of hiding after delay time
        moveLocation = 0                    !! Initializing the moveLocation command
        While Inside("Iraq", moveLocation) != 1
            moveDirection = 360*uniform()   !! picks a random direction 0 - 360 degress
            moveDistance = 100*uniform()    !! picks a random distance 0 - 100 km to move
            !! creates Location to move to
            moveLocation = me->Location + moveDistance * vector(moveDirection)
        EndWhile 1
        Move moveLocation
        While me->Location != me->Goal
        EndWhile
        Delay 30 + 30*uniform()             !! delays 30 - 60 min to set up launcher
        Fire "SCUD", "B_AOC"
        Delay 15 + 15*uniform()             !! delays 15 - 30 min to tear down launcher
        moveLocation = 0                    !!Initializing the moveLocation command
        While Inside("Iraq", moveLocation) != 1
            moveDirection = 360*uniform()   !! picks a random direction 0 - 360 degress
            moveDistance = 100*uniform()    !! picks a random distance 0 - 100 km to move
            !! creates Location to move to
            moveLocation = me->Location + moveDistance * vector(moveDirection)
        EndWhile 1
        Move moveLocation
        While me->Location != me->Goal
        EndWhile
        !! Repeat
EndWhile
```

Figure 8. Random Movement Tactical Programming Language

Force Structures: The Blue Forces

The Blue forces are the main players in our scenario and it is the Blue side for which our MOPs have been developed. The Blue side has more components than the Red side and is also quite complex – but in its entirety, not so much for any single type of agent. For our Blue Force discussion, please refer to Figure 9 below.

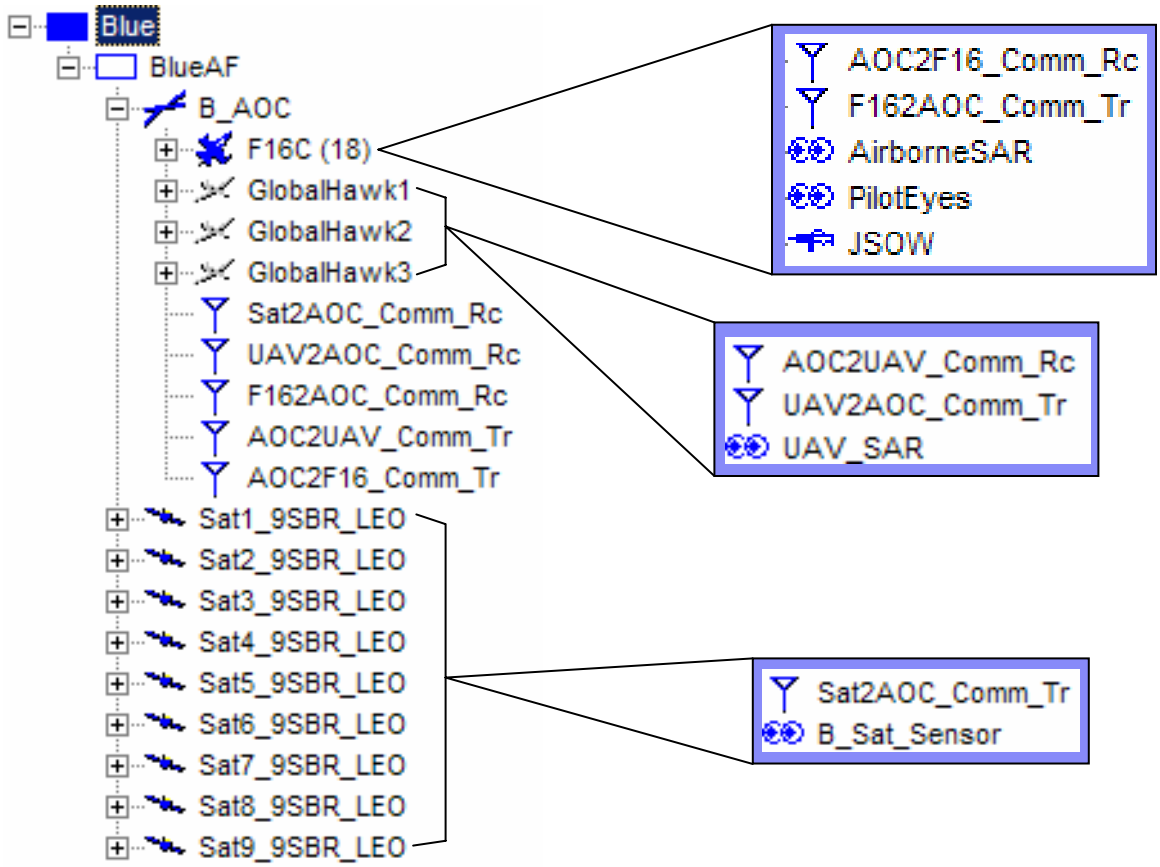

Figure 9. Blue Force Command Structure

Similar to the Red force, the Blue force is governed by a C3I headquarters named *B_AOC*. Unlike the Red force, however, the *B_AOC* does have communication links in place between itself and all the subordinate Blue agents. It is important to note that the satellite constellation feeding data to the *B_AOC* is "owned" by the Blue Air Force not by

the *B_AOC*. Thus, the satellites are not subordinate to the *B_AOC* and cannot take orders from the *B_AOC*. The *B_AOC* owns, as its subordinate agents, the *F-16C*s and the UAVs. As stated, the *B_AOC* has communication links to these agents and relays *SCUD_TEL* target information it receives from the satellites to the UAVs and F-16s. Similarly, when the UAVs detect a *SCUD_TEL* they relay that information to the *B_AOC* who in-turn notifies the F-16s. Figure 9 also shows the break-out of the devices (communication, weapons and sensors) owned by each type of agent.

The driving force behind the Blue force agents' behaviors is their CONOPS. There are several CONOPS assumptions at play in our scenario. First, is that the constellation of nine Blue satellites will relay any *SCUD_TEL* target information to the *B_AOC*. The satellites are equipped with both Ground Moving Target Indicator (GMTI) and Synthetic Aperture Radar (SAR) sensors. The GMTI sensors are flagged to detect only moving targets and the SAR sensors will only detect static targets. Upon receiving target information, the *B_AOC* passes the target information to the UAVs that are out flying surveillance missions covering specified regions of the AOO. The UAV responsible for the TAO within which a target appears will depart its TAO and fly to surveil the target. When the UAV detects the target it will notify the *B_AOC*. The UAV will then loiter around the target and, if it begins moving, follow it until it goes into defilade or until the F-16s arrive and take over operations. The F-16s, out flying Combat Air Patrol (CAP) missions in the AOO, will receive the target data from the *B_AOC*, break off from their current mission profile and proceed to the target location. Because the F-16 pilot can visually determine if the target is an active threat, when the pilot gets a visual on the *SCUD_TEL* he/she will begin the process of determining whether or not to

fire at the target (Berg, 2006). In order to capture attrition of the Red forces, we have added a probability of kill for the F-16 against the *SCUD_TEL* so that when the *SCUD_TEL* is detected and is within range of the F-16, a JSOW will be fired to remove the *SCUD_TEL* agent from the Local Target List (LTL) and, consequently, the scenario run. This data will play an invaluable part in the post-processing of the descriptive statistics.

Figure 10 below shows the AOO with its three UAVs orbiting their TAOs and the three F-16 CAPs. Notice the screen shot was taken before any targets have randomly appeared in the AOO. Figure 11 reveals an increased ops tempo due to the active targets now out of defilade and the UAVs departing their TAOs to discriminate/surveil the targets. Also note, in Figure 3.7, the communication links from the *B_AOC* to the UAVs and the F-16s. The communications links from the *B_AOC* are updating all Blue agents' LTL; however, the UAVs and F-16s only respond to those targets within their TAO and CAP area of responsibility respectively. Thus, neither a UAV nor an F-16C in the top region will respond to a detected target in the middle or lower region.

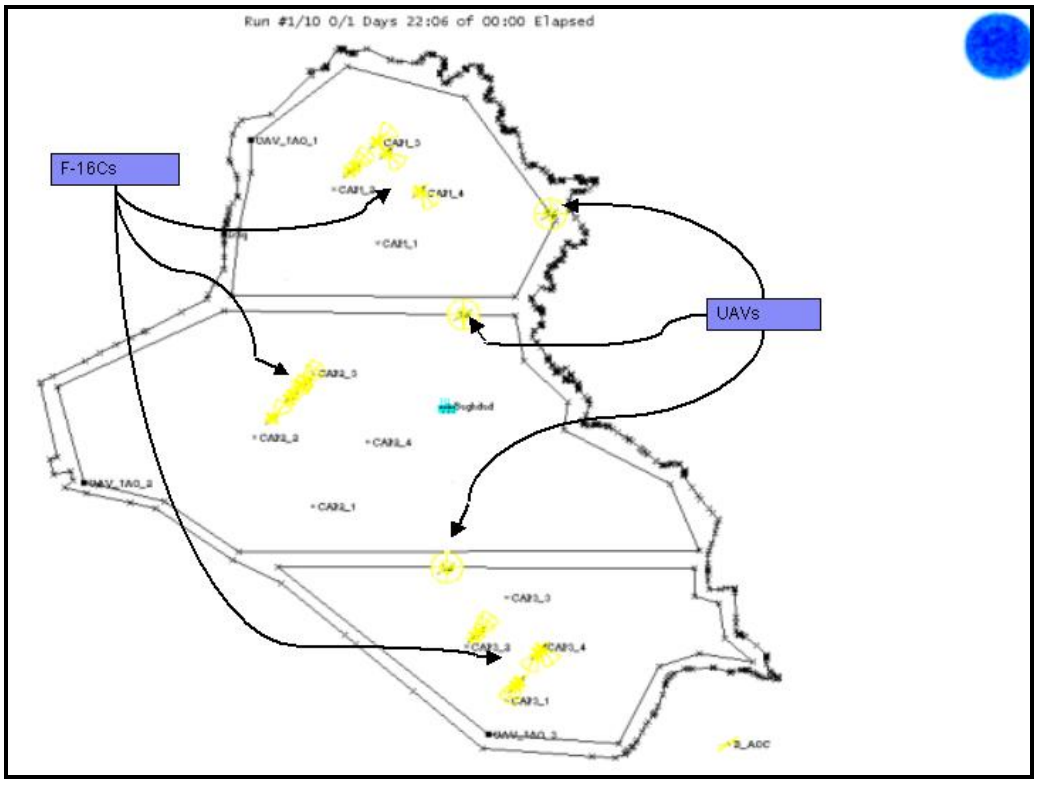

Figure 10. UAV TAOs and F-16 CAPs

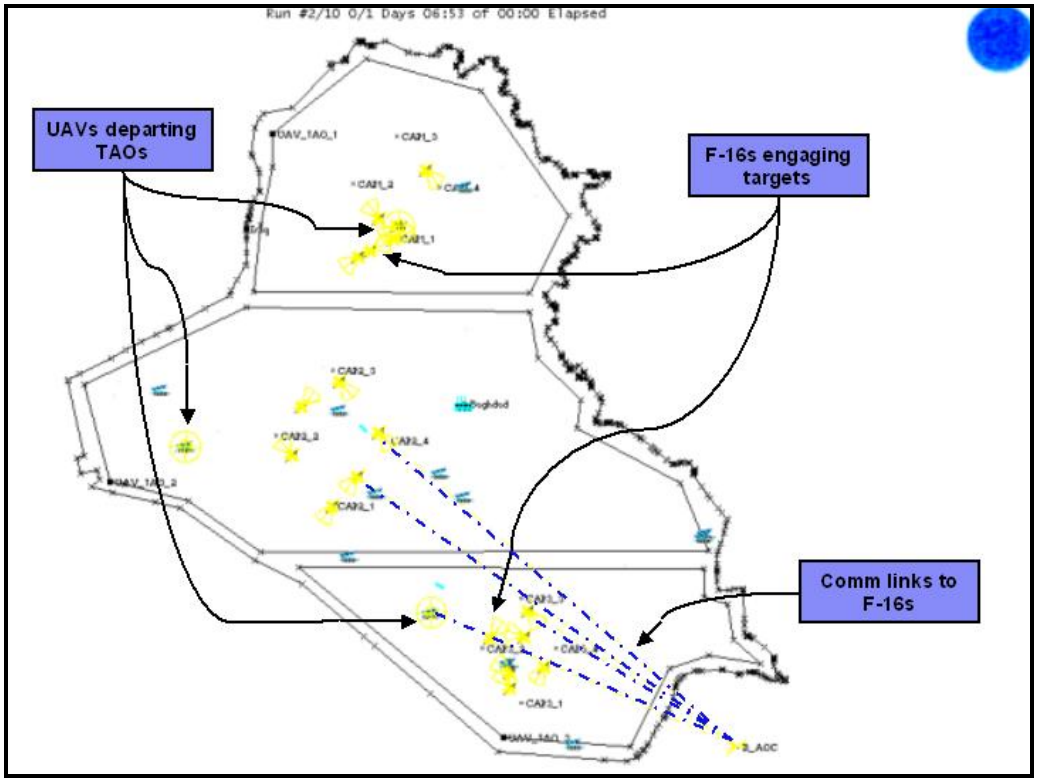

Figure 11. Communication links from *B_AOC* to UAVs and F-16s

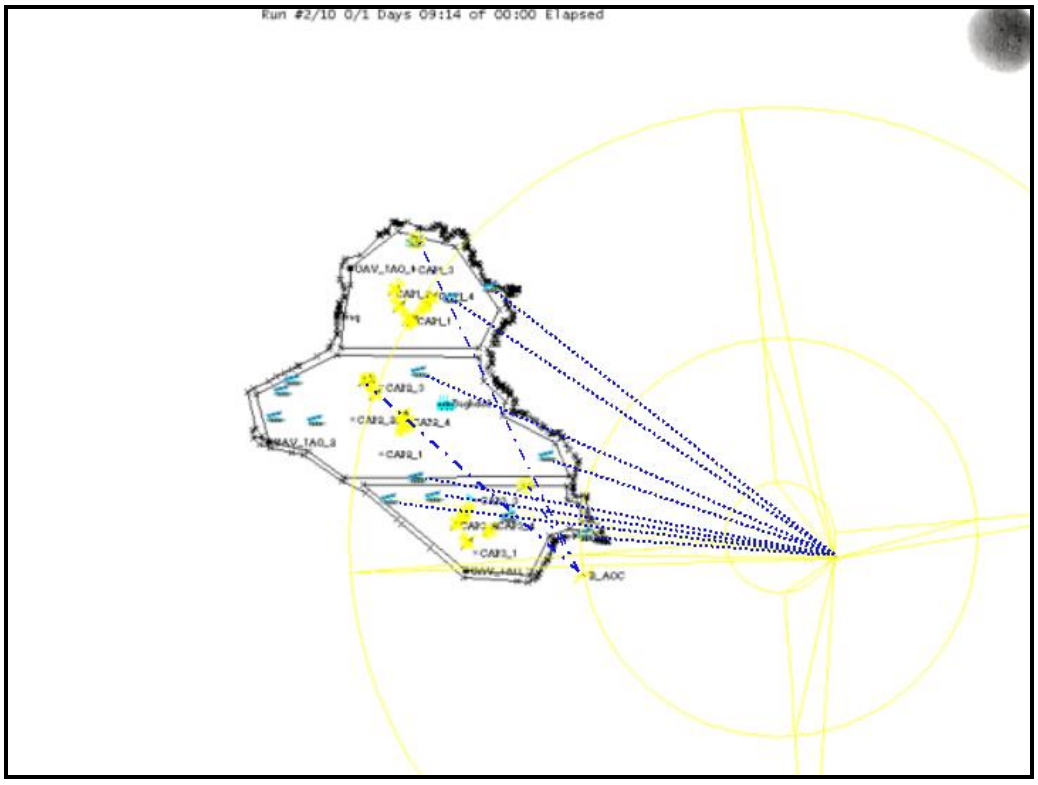

Figure 12. Satellite detecting targets in AOO

From the satellite's perspective, Figure 12 demonstrates several more aspects of the model. First, observe the satellite detecting eight of the targets in the AOO – it happens that three of them were on the move so the GMTI scored those detections. Also, note the moon in the upper right corner of the screen shot. This indicates the shot was taken during nighttime operations. The dashed lines emanating from the center of the satellite are the target detection lines. The resulting communication lines from the *B_AOC* to the UAVs and F-16s are indications that all sensor-to-shooter links are working.

Figure 13 also shows target detections by a satellite, but it also demonstrates one of SEAS's unique and powerful capabilities. In order to capture the screen shot, the simulated Earth was rotated and zoomed out.

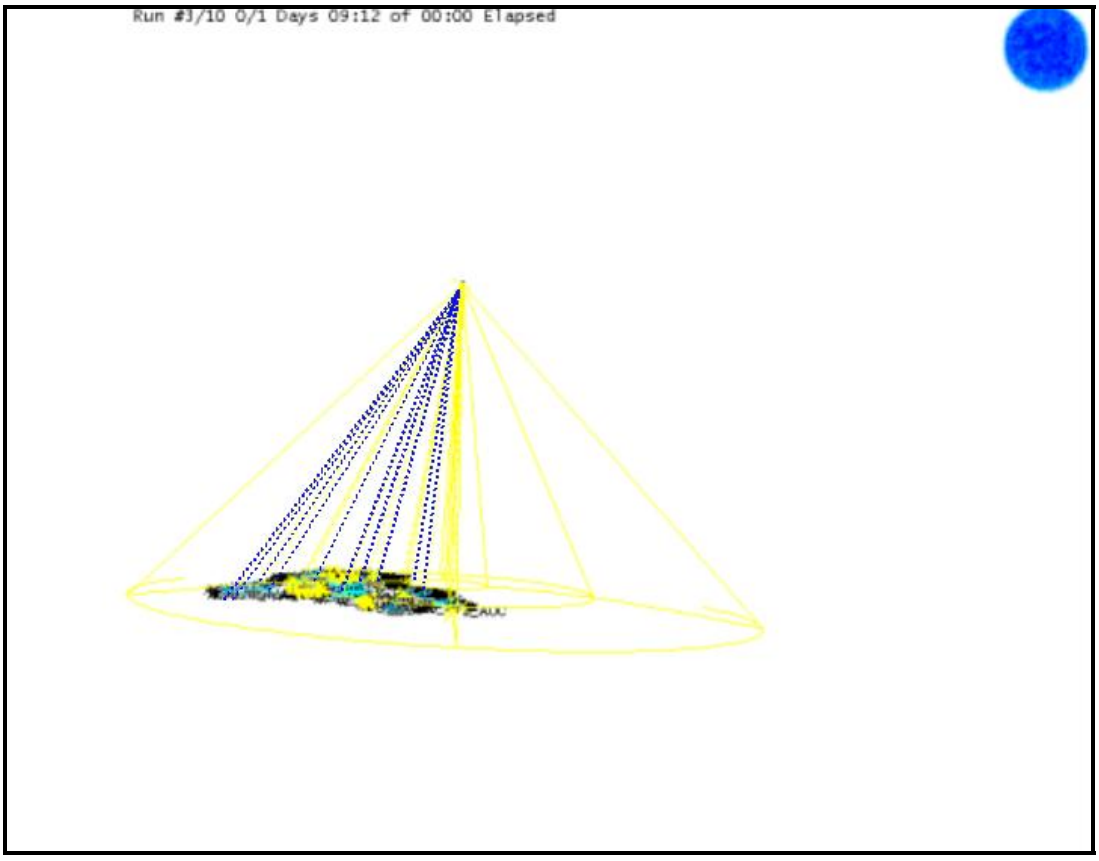

Figure 13. SEAS zoom feature showing target detections by a satellite

Conclusion

Though the major analysis occurs after the model is developed, it is crucial to have a game plan in place while building the model; and since all agents in our scenario play vital roles, every effort has been made to ensure that appropriate levels of detail are reflected in their characteristics and in their CONOPS. This chapter provides the necessary foundation for understanding the construction of the scenario to be run in SEAS. Particular attention should be paid to the MOPs as we are looking to answer specific questions from the sponsor, while at the same time, exploring some creative avenues that might provide new and interesting insight to the system and its components. In the next chapter, we will implement our design and run our simulation. The resulting

data for each test case will then be post-processed as we take steps toward the MOP data collection.

IV. Analysis and Results

Introduction

This chapter describes the post-processing analysis of the data and the results we obtained from that analysis. We start by describing each test case in more detail and then address the measures of performance (MOP) that we collected data on. With the MOP discussion, we will present our analysis and conclusions.

Flexibility in coding became a key driving force behind the development of our warfile. A generally accepted practice in most computer programming languages is the ability to "comment out" lines of code so they are not processed as part of the actual program. An example is to comment out personal notes detailing what the program is doing at that place in the code. This was an important facet of our coding because our design of experiments called for removing certain agents at different times. One approach to this was to physically comment out the agents that were not going to participate in the next scenario. However, that would have been extremely time consuming and labor intensive. Instead, we defined our design of experiments variables early in the coding so that when we desired to change test cases we only needed to change one number and not have to comment out sections of code. Removing an agent from the Baseline Test Case required enabling or disabling both its sensors and communications devices. The sensors were enabled/disabled at the agent level; however, Figure 14 shows how we partially defined the four test cases based on the communication devices that were transmitting or receiving. Figure 15 demonstrates how we were then able to change one number to switch between dramatically different test cases.

```
set Comm based on DOE Case number
If $DOE_CASE == 1
     me->"Sat2AOC_Comm_Rc"->Enabled = 1
     me->"UAV2AOC_Comm_Rc"->Enabled = 1
     me->"F162AOC_Comm_Rc"->Enabled = 1
     me->"AOC2UAV_Comm_Tr"->Enabled = 1
     me->"AOC2F16_Comm_Tr"->Enabled = 0     !! disable Comm to F16s
ElseIf $DOE_CASE == 2
     me->"Sat2AOC_Comm_Rc"->Enabled = 1
     me->"UAV2AOC_Comm_Rc"->Enabled = 0     !! disable Comm from UAVs
     me->"F162AOC_Comm_Rc"->Enabled = 1
     me->"AOC2UAV_Comm_Tr"->Enabled = 0     !! disable Comm to UAVs
     me->"AOC2F16_Comm_Tr"->Enabled = 1
ElseIf $DOE_CASE == 3
     me->"Sat2AOC_Comm_Rc"->Enabled = 0     !! disable Comm from Sats
     me->"UAV2AOC_Comm_Rc"->Enabled = 1
     me->"F162AOC_Comm_Rc"->Enabled = 1
     me->"AOC2UAV_Comm_Tr"->Enabled = 1
     me->"AOC2F16_Comm_Tr"->Enabled = 1
Else
     me->"Sat2AOC_Comm_Rc"->Enabled = 1
     me->"UAV2AOC_Comm_Rc"->Enabled = 1
     me->"F162AOC_Comm_Rc"->Enabled = 1
     me->"AOC2UAV_Comm_Tr"->Enabled = 1
     me->"AOC2F16_Comm_Tr"->Enabled = 1
EndIf
```

Figure 14. Sensor and Comm Device Instantiation for Design of Experiments

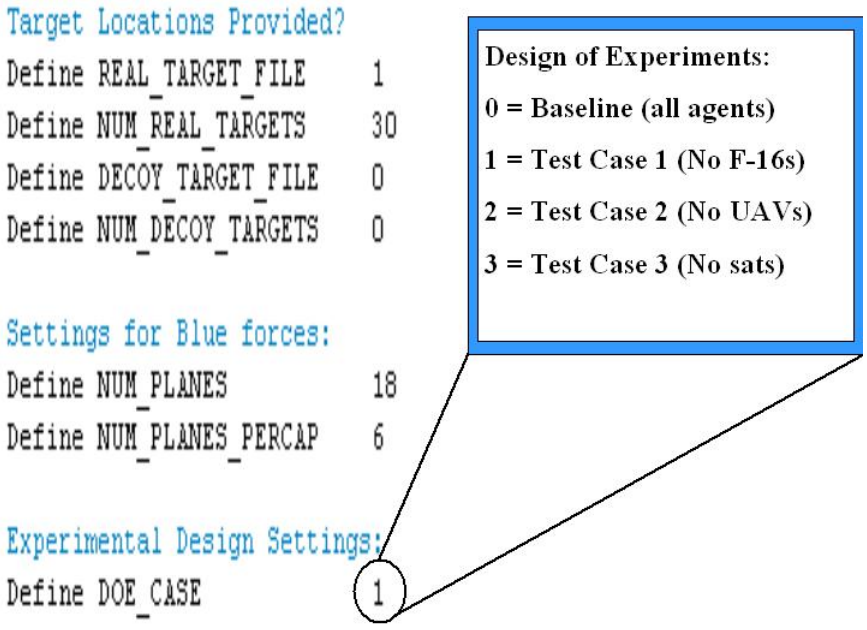

Figure 15. How to change the Design of Experiments

One of the capabilities desired of model, as discussed with Air Combat Command (ACC), was the ability to either generate a set of initial target locations or read in a set of

known target locations. To accommodate this, our model was designed to operate by either means. In other words, the set of initial target locations implemented in the model can either be randomly generated or it can be read in from a text file. This raises the issue of variance in the resulting data. For instance, if, at the start of each run, the initial target starting locations were randomly generated, then there would be a large amount of variance across all the runs for the starting location of each individual agent. In order to control the variance as discussed, we randomly generated a text file containing thirty target locations. We then locked the code to only read said file for every run of every test case. The trade-off here was that we reduced the variance, but we introduced correlation between all the runs. This correlation is in no way a threat to our model's performance and was dealt with by means of a paired-t test in our analysis. The next issue we needed to answer was how many runs to perform for each test case. As an initial cut, and assuming our data was approximately normally distributed, we started with n = 30 runs – where each run simulates 24 hours of operations. Using SEAS version 3.5 on a Dell Dimension 8300 desktop computer running a 2.60 GHz Intel Pentium 4 CPU with 512 MB of RAM, the range of time it took to run any test case was 58 seconds to 96 seconds. Our warfile is small enough at 1884 lines of code that the computing cost of performing 30 runs was negligible. However, after running the early phases of the model for 30 runs, the resulting amount of data overwhelmed the SEAS Excel-based post-processor, as it was needed for our efforts. Figure 16 is a plot produced by the post-processor for the baseline test case over 30 runs. This was produced to capture data for MOP 1: the number of initial target detections, by any sensor, within two hours of the first target detection. In order to sort and filter the data, we needed to graph "time" and "sensor" on

the x-axis with "targets" and "run" on the y-axis. Keep in mind this filtering method returns *all* of the detections by all sensors available, for each minute of each run, during that two-hour time period. As can be seen, the data is extremely dense; in fact, Excel doesn't contain enough columns (it stops at 26*26 = 676) to import all the filtering parameters and consequently removed the "run" parameter from the y-axis. Keeping in mind the plot was rotated for clarity, what we see below is read as follows: for a given time step on the x-axis, a bar indicates the number of times a sensor detected a given target in that same time step across all 30 runs. See the text box in Figure 16 for an example.

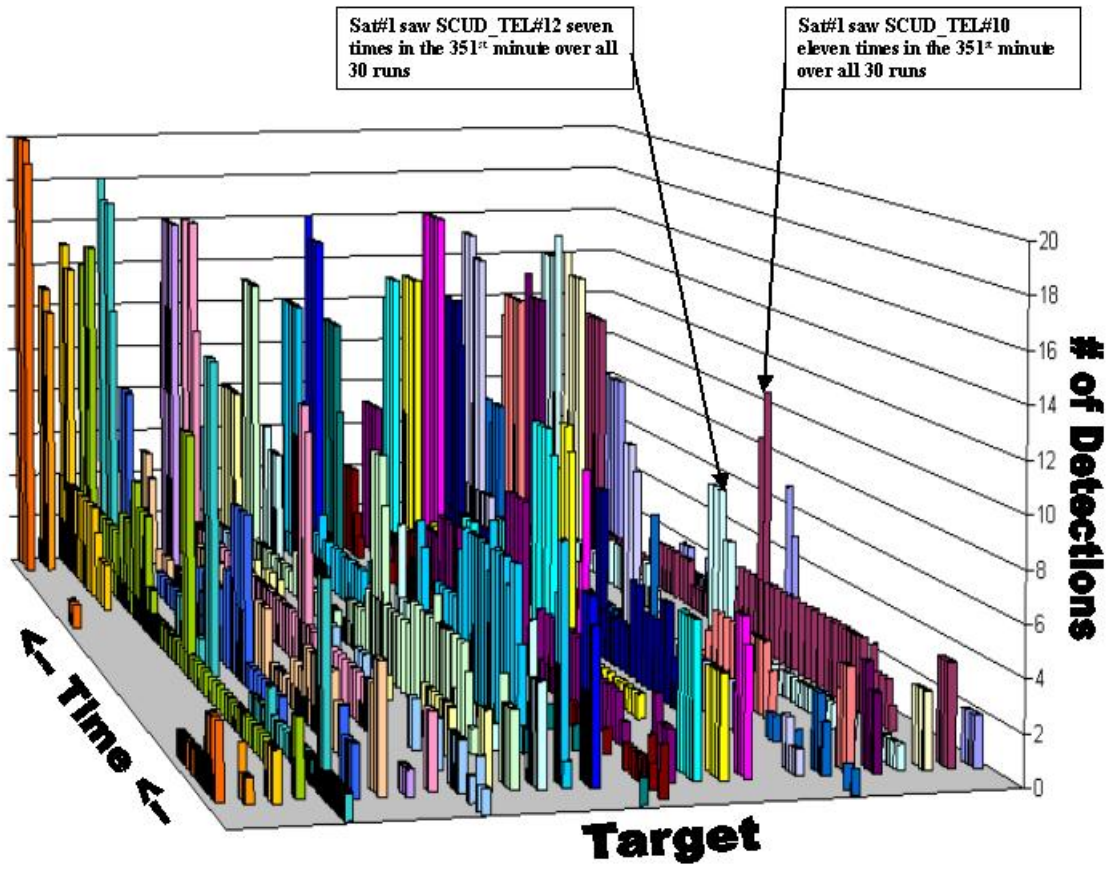

Figure 16. 30 Run data capture for MOP 1

We performed a normality check on the baseline data for ten runs and found that it was approximately normally distributed. The remaining test cases showed some departure from normality, but not enough to warrant further investigation. Thus, we reduced the number of runs down to ten so as to generate output files that were more manageable for our computing resources. Once able to import the data into the post-processor, we could manipulate and sort the data to obtain the values needed to address our MOPs. The final portion of the post-processing was taking the resulting data for each test case and calculating descriptive statistics such as expected value and paired-t test confidence intervals.

Baseline Test Case

The baseline test case is the scenario against which all the other test cases were compared. Thus, it is this test case that contains all agents represented in the warfile. For the Blue Force, those agents are the satellites, UAVs and F-16s. For the Red Force, those agents are the SCUD TELs. All P_d's were set to 1 for optimal detection within the sensor range parameters as discussed earlier, and the P_k for the F-16s against the *SCUD_TEL* agents was set to 1 – similarly, for all weapon range parameters.

One of the most important issues we focused on when comparing other test cases against the baseline was the CONOPS and the effectiveness of the force structure. In the baseline test case, the Air Operations Center (AOC) was the command, control, computer and information (C3I) hub for the Blue Force, and all targeting information was sent from the source agent (the detecting sensor) to the sink agent (the acting sensor) through the AOC. For instance, if a satellite was the first to detect a *SCUD_TEL*, it relayed that information to the blue AOC (*B_AOC*). The *B_AOC* then sent that information out to the

UAV that was responsible for the region in which the target was present. Thus far, the satellite was the source agent and the UAV was the sink agent, and all sensor detections were transferred over communications links through the AOC. Once the UAV arrived at the target location and detected the target, it effectively became the source agent and relayed the confirmed target location back to the AOC. The AOC then tasked the appropriate flight of F-16s (the sink agents) to break off their combat air patrol (CAP) and fly to the target location to destroy it. The process of detection information flowing from source agent to sink agent through the AOC was the same regardless of which agent was acting as the source agent. The one exception was if the source agent was an F-16. When that was the case, the F-16 acted upon its own goal set and destroyed the *SCUD_TEL* without being explicitly tasked by the AOC. As we work through the various test cases, note the changes in the Blue Force CONOPS. These changes, due to which agents were present in the scenario, provided interesting insight to the force structure.

Test Case 1 – No F-16s

Our first change to the baseline test case was to remove the F-16s from the scenario. Physically, they were still visible as the simulation played out; however, their sensors and all communication devices were turned off. Thus, they were effectively rendered missing from the scenario. To verify this, we checked the Sensor Detection Report generated by the processing of the warfile. We needed to make sure there were no detections made by any of the F-16s at any time during a run execution. Though it seemed redundant to turn off both the sensors and the communication devices, it was quite possible that, if only the communication devices were turned off, leaving the

sensors on, and a *SCUD_TEL* drove under one of the F-16 CAPS, the *SCUD_TEL* would be detected and destroyed by a JSOW launched from an F-16 flying overhead. As it turned out, we did indeed catch one of these "kills" and so to ensure this did not happen again, we turned off both the F-16 sensors and communication devices.

The source to sink concept still applies in Test Case 1 but only to the satellites and UAVs as the detecting agents. For instance, when a satellite detected a *SCUD_TEL* it relayed that information back to the *B_AOC* which, in turn, tasked the appropriate UAV to verify the target – the sensor chain stopped here. Because there were no F-16s, this test case could only be used for certain MOPs as discussed later.

Test Case 2 – No UAVs

Test Case 2 only involves the target detection interactions between the satellites and the F-16s. This test case is very similar to Test Case 1 in that the sensor and communication devices on the UAVs were turned off and the UAVs still flew their tactical areas of operations (TAO), but they could not detect any targets. Nor could they receive any targeting information from the AOC.

The sensor-to-shooter link was as follows: Either a satellite or an F-16 could detect a target. If an F-16 detected the target, it would destroy it without an explicit tasking from the *B_AOC*. If a satellite detected a target, the *B_AOC* would pass that target information to the appropriate flight of F-16s. The F-16 closest to the target would take a wingman and go destroy the target. The results from this test case were very interesting and generate insightful discussion later in the text.

Test Case 3 – No Satellites

Our last test case rounds out the experimental design we used in setting up our study. This test case involved only the UAVs and the F-16s. Again, the satellite sensors and communication devices were turned off, yet they were still in orbit. The sensor-to-shooter link could start with either the UAVs or the F-16s. If the F-16s detected a target, again, they fired without being tasked. If the UAVs had the first detection, they transmitted that information to the F-16s via the AOC. The F-16s would then fly to the target to destroy it.

In the next few sections, we will look at our analysis approach and then address each MOP and see how each test case applied to that MOP. Table 3 gives a summary of which test cases support the different MOPs. Refer to that table for the following discussion. We will also present the results of our analysis for each of the MOPs and discuss our conclusions.

Table 3**.** MOP to Test Case Mapping

	Baseline (All Agents)	Test Case 1 (No F-16s)	Test Case 2 (No UAVs)	Test Case 3 (No Satellites)
MOP 1	X	X	X	X
MOP 2	X		X	X
MOP 3	X		X	X
MOP 4	X		X	

Overview of Analysis

The overarching premise throughout this research has been to determine whether changes to the Blue force structure alter its effectiveness. The most valuable way to determine if statistically significant differences exist between our various test cases is through the use of paired-t confidence intervals. More specifically, we used a paired-t test because of correlation introduced by using the same starting random number seed for each test case and same starting locations for the targets in each run (of each test case). The paired-t method utilizes the differences between the means of two populations assumed (hypothesized) to be similar. Therefore, we start our analysis by defining Z_j as the difference between the random variable outputs (X_j and Y_j) from the jth run of each of our four populations (e.g. Baseline Test Case, Test Case 1, Test Case 2 and Test Case 3) taken pair-wise.

$$Z_j = X_j - Y_j, \quad \text{j} = 1,2,\ldots,10 \tag{1}$$

We fix X_j as the random variable output from the Baseline Test Case. Y_j is then the random variable output from any of the other test cases at the time of comparison. For instance, according to Table 3, MOP 3 involved comparing the Baseline Test Case against Test Case 2 and the Baseline Test Case against Test Case 3. More detail is provided when the individual MOPs are addressed. The next variable of interest is the expected value of the $Z_j's$, which is

$$\overline{Z}(n) = \frac{\sum_{j=1}^{n} Z_j}{n}. \tag{2}$$

The approximate 100(1-α) percent confidence interval is defined by

$$\overline{Z}(n) \pm t_{n-1,1-\alpha/2} \sqrt{\frac{VAR[\overline{Z}(n)]}{n}}, \tag{3}$$

where $t_{n-1,1-\alpha/2}$ is the test statistic and $VAR[\overline{Z}(n)]$ is defined as

$$VAR[\overline{Z}(n)] = \frac{\sum_{j=1}^{n} [Z_j - \overline{Z}(n)]^2}{(n-1)}. \tag{4}$$

For MOP 2 and MOP 3, we needed to consider the attrition of the target set. The continual change in the number of targets implied that we needed to look at the proportion of targets for each of those MOPs. The results of these Bernoulli trials have a variance of

$$\hat{p}(1-\hat{p}) \tag{5}$$

Thus, we defined the $100(1-\alpha)$ confidence interval around the sample proportion, $\hat{p}$, as

$$\hat{p} \pm t_{\alpha/2,n-1} \sqrt{\frac{\hat{p}(1-\hat{p})}{n-1}}. \tag{6}$$

The resulting confidence intervals are provided in the discussion for each MOP.

MOP 1

MOP 1 was defined as "The number of initial target detections, by any sensor, within two hours of the first target detection."

Analysis

Reference Table 3 for the populations we compared. Table 4 identifies the lower and upper bounds for the 95% paired-t confidence intervals we developed for MOP 1.

Table 4. MOP 1 Confidence Intervals

Paired-t 95% Confidence Intervals with Statistical Significance				
Baseline Versus	$\overline{Z}(n)$	***C.I. Lower Bound***	***C.I. Upper Bound***	***Statistically Significant?***
Test Case 1	1.00	-2.03	4.03	**No**
Test Case 2	-0.20	-2.68	2.28	**No**
Test Case 3	9.80	7.70	11.90	**Yes**

For the Baseline versus Test Case 1, we see that the confidence interval includes zero meaning there is no statistical difference (at 95%) between the two modeled scenarios. Appendix B contains the actual per-run values for all comparisons. Figure 17 illustrates the initial detection count for the first two hours in the Baseline and Test Case 1 respectively. Recall that in Test Case 1, the F-16s were removed from the scenario resulting in no initial detections from the F-16s and fewer initial detections overall. As can be seen, there are times when the Baseline obtained better results, and there are times when Test Case 1 performed better. The fact that there was often similar behavior for similar runs leaves us with no clear distinction between the two cases and thus confirms the paired-t test results above.

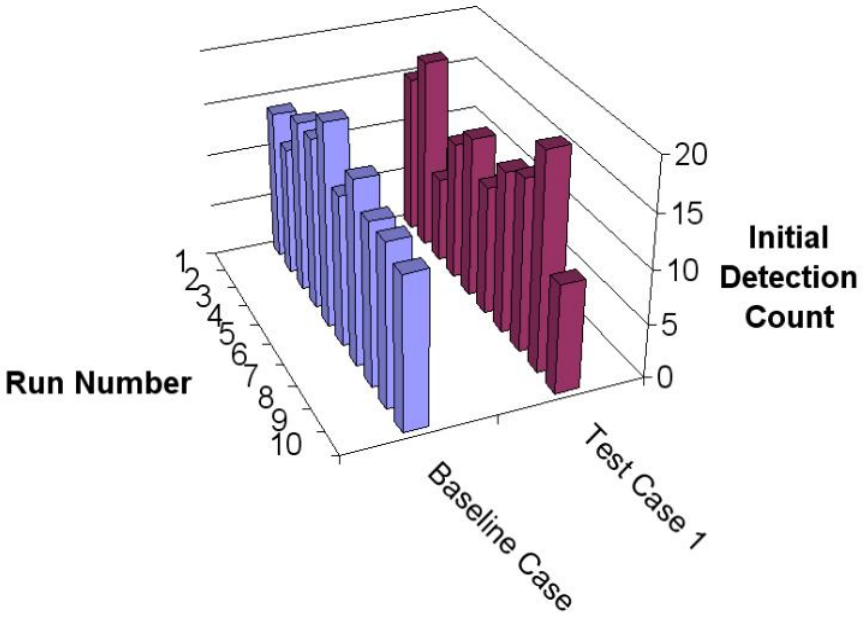

Figure 17. No F-16 Detections (2hrs)

For the Baseline case versus Test Case 2, the confidence interval again includes zero, so there is no statistical difference between the two force structures. However, we see that the average number of detections slightly increased in Test Case 2. Consider the scenario: the UAVs were removed leaving the F-16s getting updated target information from the satellites (via the AOC). Now consider that there are 18 F-16s in the scenario and when they received independent target information, they broke off their CAP in pairs to pursue the first target on their LTL. So there were effectively nine sensors at work in the air space in this scenario versus only three when the UAVs were present, as in Test Case 1. Figure 18 shows the initial sensor detections without the UAVs compared to the Baseline.

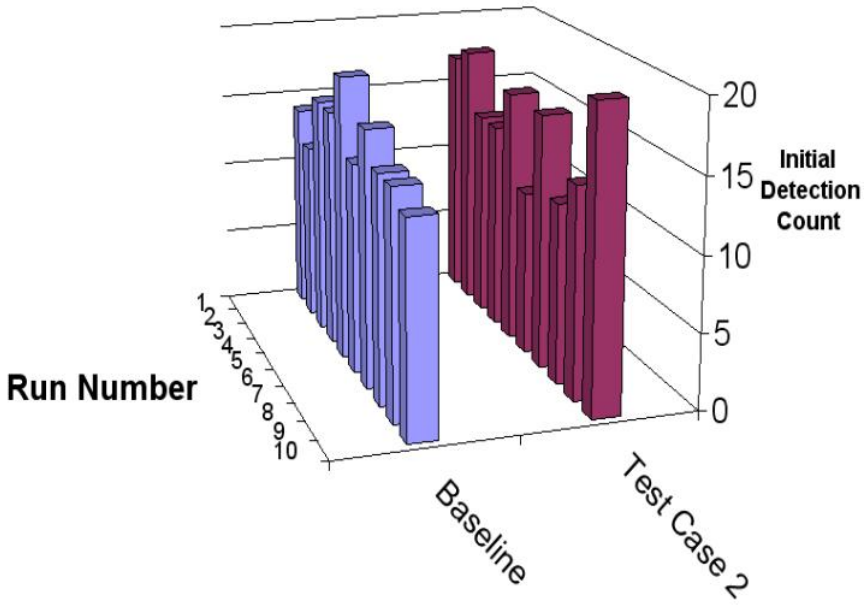

Figure 18. No UAV Detections (2 hrs)

Additionally, Test Case 2 revealed how the UAVs could be a bottleneck in the elimination of time critical targets (TCT). This was because the F-16s flew CAP until targets were verified by the UAVs. At that time, and only at that time, would the F-16s depart their CAP to destroy a target. Although the F-16s detected a target if they flew over it while on CAP or ingressing/egressing from a target on their LTL, the majority of their detection efforts originated from the information they received from the UAVs.

In the Baseline case versus Test Case 3 the confidence interval does not contain zero. Here we can definitively state there is a difference between the force structure, as it exists in the Baseline, and the force structure without any satellites. This is a trivial case because it goes without saying that the satellites have the highest number of initial detections over the 24-hours of operations. Figure 19 illustrates the number of initial detections without satellites compared against the Baseline case.

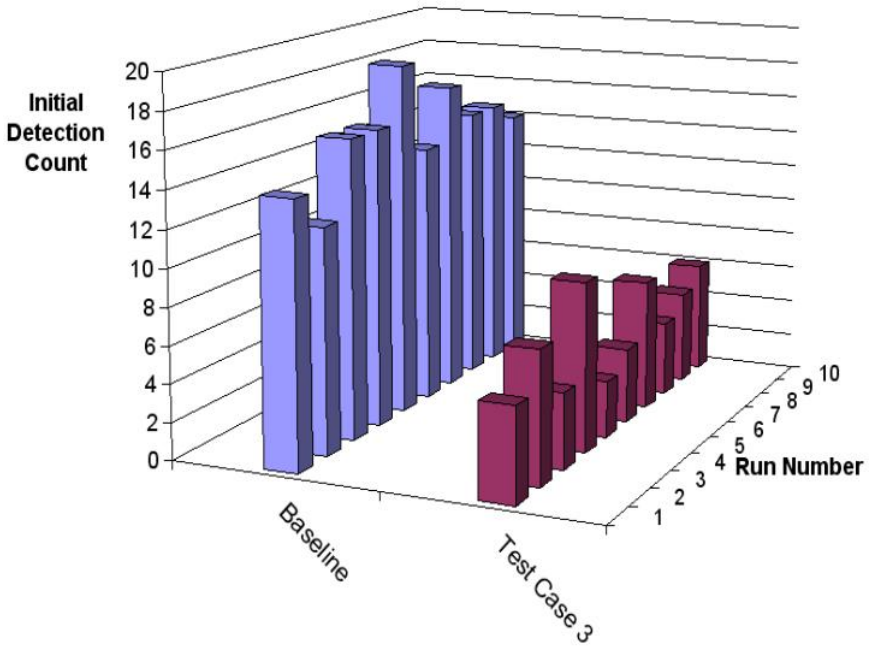

Figure 19. No Satellite Detections (2 hrs)

MOP 2

MOP 2 was defined as "The proportion of targets destroyed before being able to fire."

Analysis

Reference Table 3 for the populations we compared. Table 5 identifies the lower and upper bounds for the 95% proportion confidence intervals we developed. Due to the nature of this MOP, we needed to recognize two things: first, attrition is involved in this MOP causing the probability of detection to change over the course of the 24-hour operations; second, the number of targets destroyed before firing would never exceed the total number of targets in the scenario. Therefore we derive point and interval estimates of the proportion of targets (out of 30) destroyed before being able to fire.

Table 5. MOP 2 Confidence Intervals

	C.I. Lower Bound	***C.I. Upper Bound***	***Average Proportion***	***Statistically Significant?***
Baseline	0.0113	0.7420	0.3767	**N/A**
Test Case 2	0.1502	0.9032	0.5267	**No**
Test Case 3	0.0	0.6500	0.3033	**No**

As is seen in Table 5 the average proportion of "kills" increased from the Baseline to Test Case 2, however, since the confidence intervals overlap we are unable to detect a statistically significant difference at 95% confidence. Recall, Test Case 2 operated without UAVs. We hypothesize that we removed the bottleneck, as discussed for MOP1, and allowed the F-16s to respond more rapidly to the TCTs. At this point, the F-16s are coded to break off their CAP, in pairs (a flight leader and a wingman), and pursue the latest target added to the LTL (as long as the target was within their area of responsibility). Without the UAVs, the nine pairs of F-16s were able to pursue nine different targets simultaneously. If there were less than nine new targets on the LTL, then pairs formed for the current targets and the remaining F-16s went with the last pair. For instance, say there were six targets on the LTL and each CAP region contained two of them. In all three CAP regions, one pair of F-16s would pursue one target and the remaining four F-16s would pursue the second target. Figure 20 graphs the kill count for the number of targets that were destroyed in each run before being able to fire.

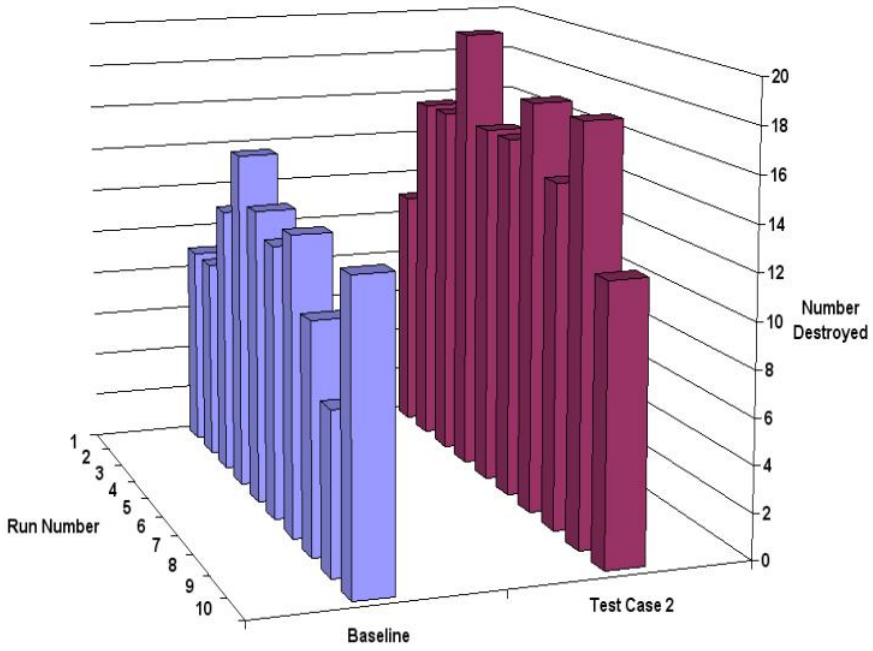

Figure 20. Kills Before Firing With No UAVs

When we compared the Baseline against Test Case 3, we didn't see a statistically significant difference between the two populations. Table 5 depicts this numerically by the confidence intervals and Figure 21 graphically shows the number of targets destroyed in the Baseline and Test Case 3.

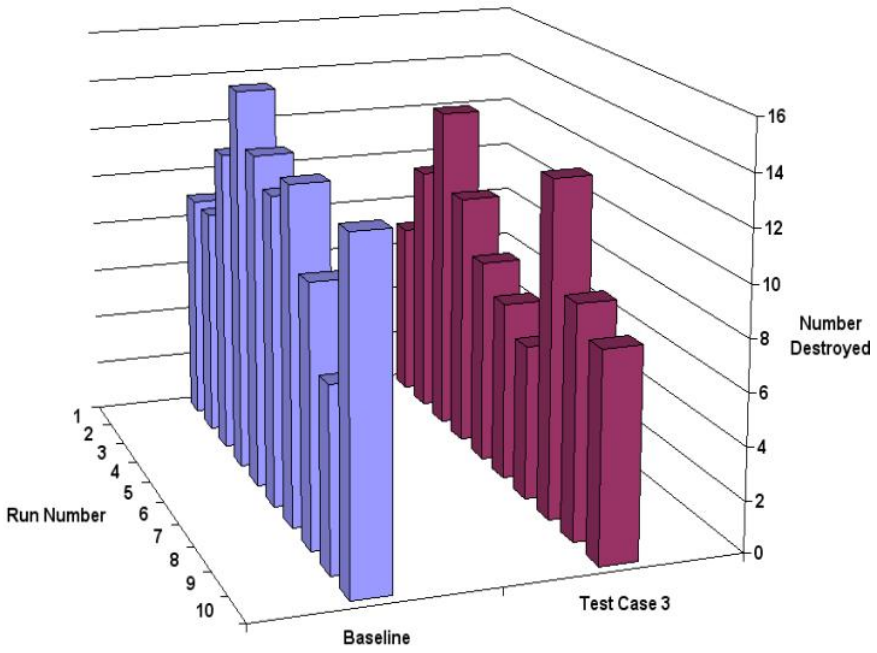

Figure 21. Kills Before Firing With No Satellites

We thought Test Case 3 would be a trivial case. Our initial hypothesis was that when the satellites were removed from the scenario – eliminating a majority of the initial detections – we expected to see an overwhelming decrease in bombs-on-target. However, comparing the proportional confidence intervals indicates there are no statistical significance between and the baseline and Test Case 3. The average number of targets destroyed decreases by only two over all ten runs. The number of detections did not appear to influence the mission objective of targets killed before firing. This doesn't downplay the importance of satellite assets in-theater, but for our model it does reveal an interesting aspect of time critical targeting.

MOP 3

MOP 3 was defined as "The proportion of targets destroyed by the end of the 24-hour period of operations."

Analysis

Reference Table 3 for the populations we compared. Table 6 identifies the average proportions as well as the lower and upper bounds for the 95% proportional confidence intervals we developed for MOP 3. The proportional technique was used for this MOP because, like MOP 2, there was an absence of normality due to the attrition of the target set over time. This attrition caused the probabilities of detection and kill to change over the course of the 24-hour operations. Additionally, the number of targets destroyed would never exceed the number of targets in the scenario forcing us to view the proportion of targets destroyed.

Table 6. MOP 3 Confidence Intervals

	C.I. Lower Bound	***C.I. Upper Bound***	***Average Proportion***	***Statistically Significant?***
Baseline	0.5357	1.1109	0.8233	**N/A**
Test Case 2	0.7376	1.1224	0.9300	**No**
Test Case 3	0.2116	0.9551	0.5833	**No**

Figure 22 shows the number of targets destroyed in both the Baseline and Test Case 2. Even though the average number of targets destroyed before the end of operations increased when the UAVs were removed, the difference in proportions is not statistically significant.

Figure 23 depicts the drop in number of targets destroyed, from the Baseline to Test Case 3, when the satellites were removed from the scenario. Again, drawing from Table 6, we are not able to make any definite conclusions about the effectiveness of the F-16s or the UAVs. The fact that removing the satellites meant the UAVs were, generally speaking, the source of initial target detection still has no definitive meaning here even though the average proportion of targets destroyed dropped in Test Case 3.

Again, the proportionality of the populations, given the attrition and small number of replications, showed no statistically significant difference.

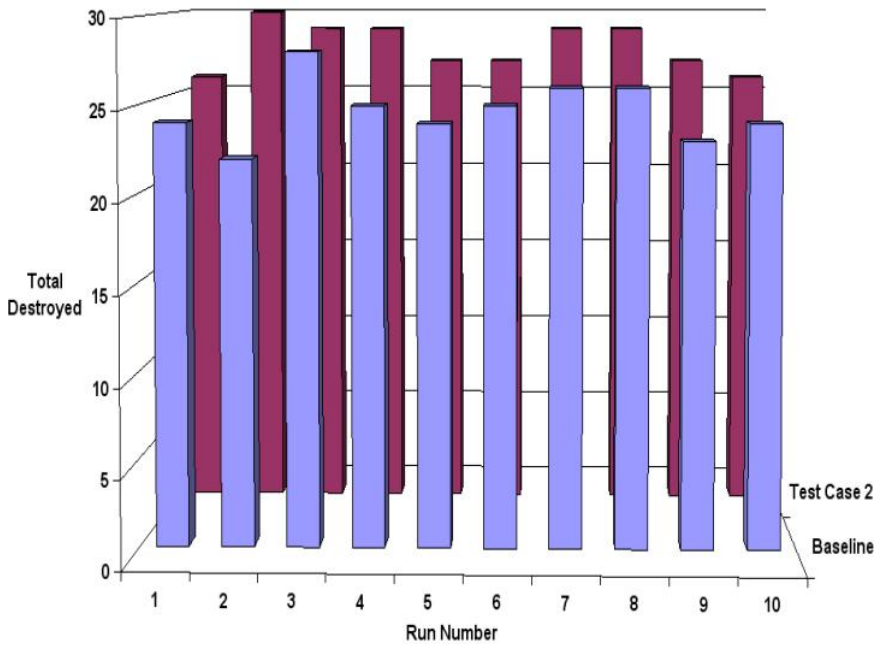

Figure 22. Total Kills With No UAVs

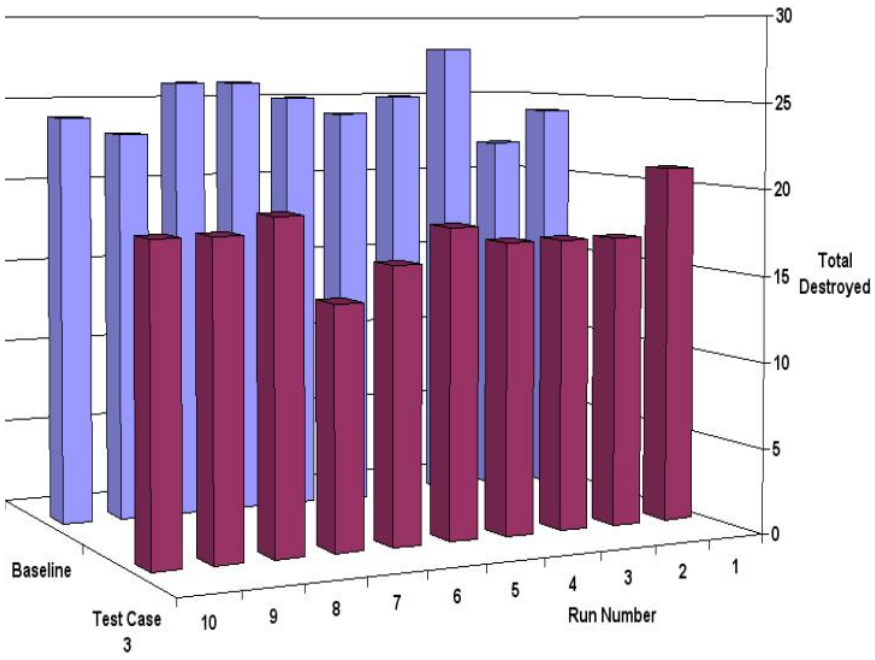

Figure 23. Total Kills With No Satellites

MOP 4

MOP 4 was defined as, "Which Blue Force agent detected which target and at what time did the detection occur?"

Analysis

MOP 4 is a measure we looked at for a single run without any statistical comparisons between test cases. It was selected for inclusion because of its operational nature – TCTs. The data was collected over 10 runs and filtered to display the Blue Force sensor detections, by agent, over one run (24 hours). The variables graphed were "Time" and "Run" on the x-axis, while "Sensor" and "Target" were on the y-axis. We then filtered the data again to display only one target, *SCUD_TEL#10*, over one run. Following Figure 24, we step through the series of events in the graph.

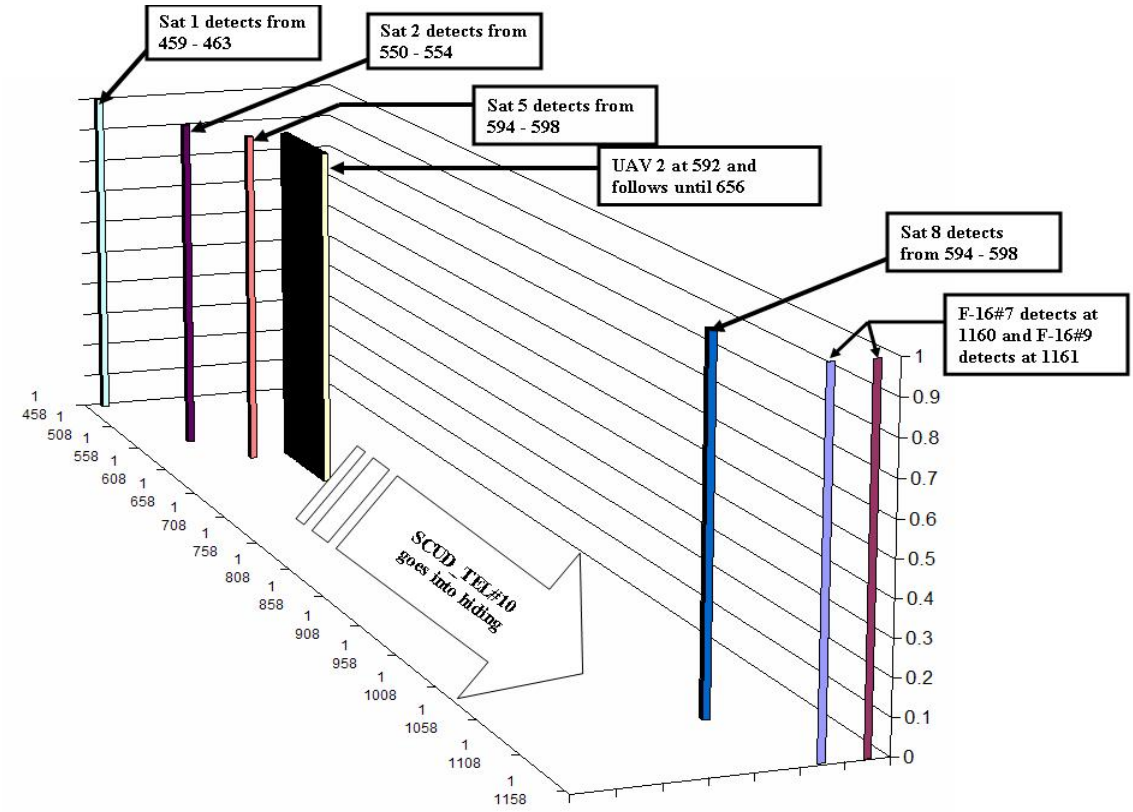

Figure 24. Who saw what and when

At 459 minutes from the start of the scenario, *Sat1* detected *SCUD_TEL#10* and continued the detection through its orbit until 463 minutes. At that time the target detection was transmitted to *B_AOC* where it updated the LTL. The LTL was then transmitted from the *B_AOC* to *GlobalHawk2* which was flying on *UAV_TAO_2*. *GlobalHawk2* then departed its TAO and proceeded to the target location. While it was in-route to the location, *Sat2* passed over the region and detected the target from 550 – 554 minutes. That information was relayed to the AOC and passed on to the UAV. With the LTL now updated, *GlobalHawk2* altered its course towards the new location of that target. We know the target has moved since the first detection because of the Red Force CONOPS, and was now somewhere in its setup, fire and tear down delay and preparing to move for defilade again. *GlobalHawk2* arrived at the location and initially detected *SCUD_TEL#10* at 592 minutes from start. Shortly thereafter, *Sat5_9* crossed over the region and detected the target from 594 – 598 minutes. The target began moving some time after *GlobalHawk2* got its initial fix at 592 minutes because the UAV tracked the target until 656 minutes – that's just over one hour. After the initial UAV detection, *GlobalHawk2* transmitted the target location to *B_AOC* which updated the LTL and re-transmitted to the F-16s flying CAP in that region. The F-16s broke off their CAP and flew towards the target location on the LTL; however, by the time they arrived *SCUD_TEL#10* had already gone into defilade and *GlobalHawk2* had left the area – an opportunity lost for the Blue Forces. The next time *SCUD_TEL#10* appeared on the radar was at 1090 minutes when *Sat8_9* detected it. Once again, the LTL was updated with the target location and transmitted to the AOC and then on to the appropriate UAV – it is possible that the target crossed from one region to another before being detected by

Sat8_9. Thus, it may not have been *GlobalHawk2*'s responsibility. Regardless of where the target was located, the notified UAV never made it to the target location. What appears to have happened this time, is that *F-16#7* and *F-16#9* were flying off CAP (possibly engaging another target or returning to their CAP) and detected *SCUD_TEL#10* with their SAR sensors. F-16#7 was the lead plane and detected *SCUD_TEL#10* at 1160 minutes; F-16#9 was the wingman and detected the target at 1161 minutes. Bombs were on target at 1162 minutes.

By comparison, Figure 25 was graphed to show the difference in the same event when the UAVs were removed. This is possible because we used the same random number seed, the same starting target locations and the same replication number. Notice that the differences in times are due to the hiding and movement "choices" our *SCUD_TEL* agents can make.

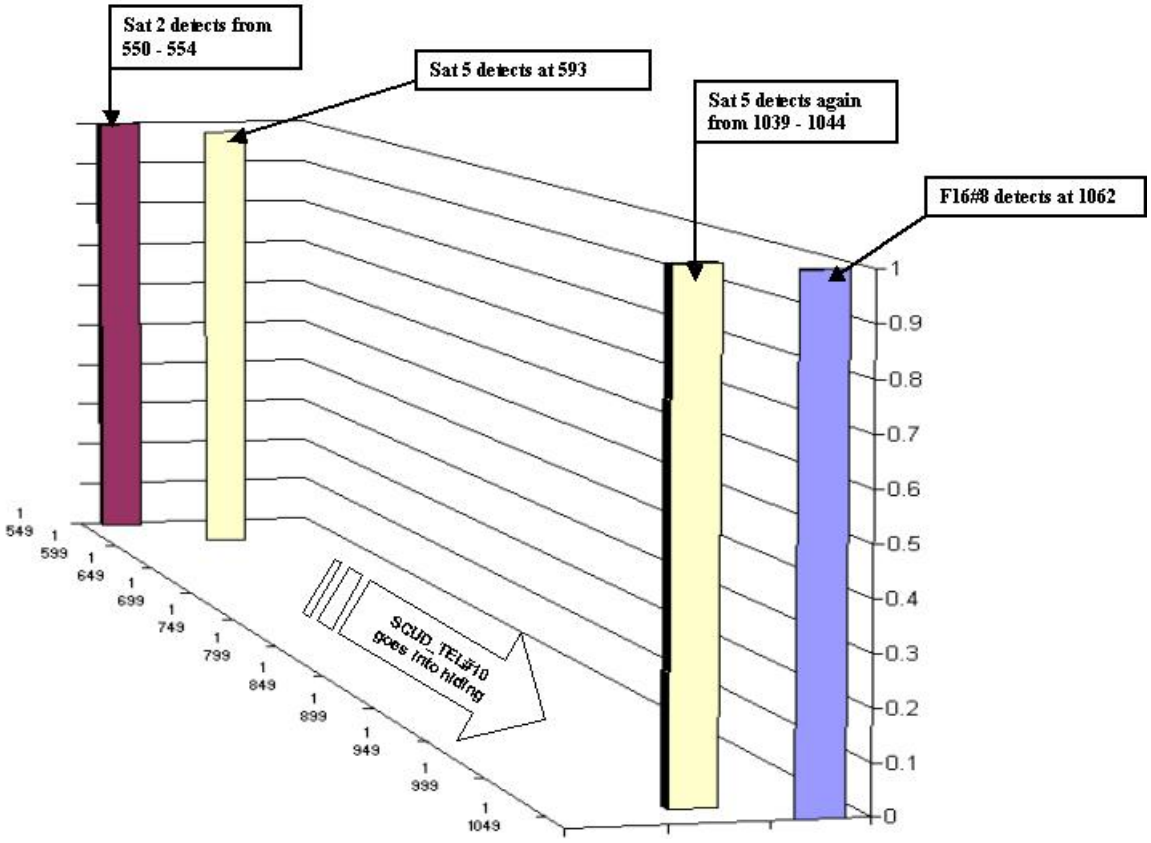

Figure 25 Who saw what and when, with no UAVs

Overall Implications of Results

The analysis of each MOP revealed interesting information about the Blue Force effectiveness. Namely, the satellites are key contributors to early detection of TCTs. However, the early detections did not always lead to faster response times by the F-16s; especially when the UAVs were in the kill chain. Although it didn't always happen that the satellites were the agents scoring the initial detection, we noticed that when they were removed from the scenario completely the overall average number of detections dropped. Additionally, the UAVs seemed to contribute counter-intuitively in our model. That is, we expected the UAVs to improve the kill chain prosecution by being able to cover larger

areas of the AOO, but our results showed otherwise. The slow F-16 response in getting to the targets can largely be explained by the CONOPS in place.

The next chapter concludes our research efforts with some additional insight to the results of our MOPs. Moreover, suggested changes to our model will be made and future areas of research will be recommended.

V. Conclusion

Overview

Our research used SEAS, an agent-based simulation tool, to investigate the SCUD hunt problem that United States and Coalition forces encountered during Operation Desert Storm. Initially we considered using an existing SEAS model; however, we ultimately decided that constructing our own would provide the benefit of containing exactly what we wanted in it. That is to say, other models, though they were perfect for their intended use, may have contained too much agent behavior that was not desired for our purposes. Thus, we were able to build a relatively small and simple model that adequately represented the system we were trying to simulate. Our research was sponsored by Air Combat Command (ACC) and provided insight to measures of performance (MOP) they were interested in. This chapter concludes our research by summarizing the MOPs we collected data for; presenting issues with assumptions and model development; and finally, recommendations for future avenues of research using SEAS or agent-based models in general.

Summary of Measures of Performance

Four unique MOPs were developed for our research. They are listed here:

1. The number of initial target detections, by any sensor, within two hours of the first target appearance.
2. The proportion of targets destroyed before being able to fire.
3. The proportion of targets destroyed by the end of the 24-hour period of operations.
4. Which Blue Force agent detected which target and at what time did the detection occur?

These MOPs are the result of partnering with ACC and understanding the capabilities of an agent-based model such as SEAS. Much of the data could be obtained by using the SEAS post-processor; however, some information had to be manually extracted after post-processing. Table 7 is a summary table of the numerical results from MOPs 1, 2 and 3.

Table 7. Summary of MOP Results

MOP 1 Paired-t 95% Confidence Intervals with Statistical Significance				
Baseline Versus	$\overline{Z}(n)$	***C.I. Lower Bound***	***C.I. Upper Bound***	***Statistically Significant?***
No F-16s	1.00	-2.03	4.03	**No**
No UAVs	-0.20	-2.68	2.28	**No**
No Satellites	9.80	7.70	11.90	**Yes**
MOP 2 95% Proportional Confidence Intervals with Statistical Significance				
	C.I. Lower Bound	***C.I. Upper Bound***	***Average Proportion***	***Statistically Significant?***
Baseline	0.0113	0.7420	0.3767	**N/A**
Test Case 2	0.1502	0.9032	0.5267	**No**
Test Case 3	0.0	0.6500	0.3033	**No**
MOP 3 95% Proportional Confidence Intervals with Statistical Significance				
	C.I. Lower Bound	***C.I. Upper Bound***	***Average Proportion***	***Statistically Significant?***
Baseline	0.5357	1.1109	0.8233	**N/A**
Test Case 2	0.7376	1.1224	0.9300	**No**
Test Case 3	0.2116	0.9551	0.5833	**No**

Before reviewing the results from the above table, we must address the fact that our sample populations were small, n = 10. This sample size was chosen primarily because larger sample sizes resulted in data manipulation issues within Excel. In addition, a quick analysis of the Baseline data revealed our output was approximately normal with only 10 replications. By using the same starting random number seed for each test case and because of correlation introduced in generating the target starting

locations, our analysis plan for MOP1 included a paired-t test. Specifically, we programmed our model to read in the same set of target starting locations for all replications across all test cases. MOP 2 and MOP 3 required the use of a proportional test for their confidence intervals. This was due to a violation of the normality assumption due to attrition of target set.

The results of our MOPs fall into three categories: 1) Effectiveness of the F-16s; 2) Effectiveness of the UAVs; and 3) Effectiveness of the satellites. The three test cases we used were the result of our design of experiments where we removed one type of sensor platform at a time. In other words, we had a Baseline case where all three agent-types contributed to the data, Test Case 1 excluded only the F-16 agents, Test Case 2 excluded only the UAV agents, and Test Case 3 excluded only the satellite agents. Test Case 1 maps to the first category just mentioned, Test Case 2 maps to the second category, and Test Case 3 to the third.

Let us start by summarizing the confidence intervals. When a confidence interval spans zero, we are left only to say that we cannot make any definitive statement about the difference between the two populations. Such is the case for MOP 1 when we compared our results for the Baseline against the same results from Test Case 1 and then against Test Case 2. Comparing against Test Case 3, however, resulted in a statistically significant difference in the populations. We concluded that removing the satellites left a large gap in early detections. The lingering results of that were less overall detections by the UAVs and, consequently, the F-16s.

For MOP 2, when we compared the Baseline against Test Case 2 the confidence intervals overlapped indicating no statistically significant difference between the two

populations. However, we noted that the average proportion of targets destroyed before being able to fire increased. We speculated that was because the UAVs were absent. The F-16 behavior became more evident, though, while observing the simulation play out showing they could cover a larger target set without having to wait for the UAV handoff of one target. The Baseline compared to Test Case 3 brought a decreased average (not statistically significant) because of the loss of early detection from the satellites.

Finally, MOP 3 again no statistically significant differences were detected between the populations. In Test Case 3, the presence of the UAVs and the consequent slow through-put of targets to the F-16s nullified any early satellite detections as seen in the other test cases; however, if anyone asks, "Why space?" – this research gives some insight.

MOP 4 is a stand-alone measure looking at a specific simulated engagement for time critical targeting. In this MOP, we generated results to address the question "Who saw what when?" The results of this question can have far-reaching impact. We derived no quantitative metrics for this MOP, but we graphically displayed the results of all sensor detections against one target over the course of one run. The conclusion we arrived at is "it depends." It depends on what the CONOPS were that we programmed for our agents. It depends on how robust our model was and how we implemented operational tactics. It depends on the assumptions we made about sensor parameters. There were many components of our model that we found could have made a difference in the TCT aspect of our research, but in the end, we concluded that our model does give a starting point for valuable insight.

Assumptions and Model Development

As just mentioned, the assumptions we made while developing our model had an impact on the results we obtained and presented for our research. This section details some of those assumptions and the resulting impact on the model.

First, to keep our model simple we gave all agents unlimited fuel. This seems plain enough, but it implies that our UAVs and F-16s never had to worry about leaving their position (even if they were replenished by another similar mission asset). It also meant we didn't have to concern ourselves with modeling tanker assets or turn-around time for maintenance. Additionally, crews were eliminated altogether by this assumption.

Second, we assumed instantaneous communication transfer with no degradation to the signal or corruption of the data. This means that all sensor-to-shooter links happened with no latency in the signal. For instance, the real-world Global Hawk platform can be flown from in-theater or from halfway around the world. In either case, the bandwidth required to send all of its imaging data and communication data is enormous and can take minutes to accomplish due to the numerous nodes the signal may have to pass through before reaching the command and control element. Even then, there can be considerable delay in the processing of the imagery and making the requisite decisions. Our model assumes no effect on the performance of the agents due to these nodal delays and latencies. Thus, it ultimately provides a "best-case" scenario for all communication and data links. Additionally, we chose not to model weather and terrain for the same reasons.

Our third assumption was to set all probabilities of detection and probabilities of kill to one (1). Considering our military's experience with the SCUD hunt mission and the effectiveness of our modern inventory, we chose to eliminate the target set on a one-strike basis.

One last assumption to mention is that we assumed all sensor capabilities were present in our "cookie-cutter" design. Ultimately, we desired flexibility in our model's ability to represent numerous sensor types. Thus, we used the SEAS sensor template and applied appropriate settings and input parameters that could be tailored in the future to meet more specific sensor objectives. The inclusion of weather and terrain TAOs would greatly increase the complexity of this model and the resulting affects on sensor capabilities.

Recommendations for Future Research

The SCUD hunt problem is not new to the United States military, but it reveals a larger problem that we have yet to master. That is, searching for elusive, hiding targets. Thus, our first recommendation is to modify the F-16 CONOPS allowing them to sweep for a target should they arrive at a target location after the target moved. Additionally, changing MOP 1 to reflect the time of the first target appearance would ultimately reduce the kill chain timeline. Another opportunity for model growth would be the addition of weather and terrain TAOs. Weather TAOs would affect the sensors' P_d whereas terrain TAOs affect the sensors' range. Finally, instituting decoy targets would round out our model's design. ACC currently uses decoys as well as real targets in their Extended Air Defense Simulation (EADSIM) model. The value of decoys is that they can distract the Blue Forces from attaining their goal of finding the real SCUD_TEL targets. Moreover,

decoys would force the use of P_{id} and the IFF probability. For SEAS, the P_{id} is the probability of identifying any target against which the sensing agent has a probability of detection. The IFF probability is the probability of misidentifying a friendly agent as an enemy target.

We believe that agent-based modeling is a powerful and robust concept that can greatly aid in the modeling and simulation of the SCUD hunt problem. Having a model where agents can interact with each other, with the environment and contain the ability to make decisions is applying a "near-reality" capability to a very real problem. Our model has demonstrated itself to be a useful tool – flexible for analysis and capable of analyzing trade-offs between the various mixes of sensors used for the SCUD hunt problem. The model adds value to the understanding of this type of problem and is a dynamic model useful for analyzing additional tradeoffs.

Appendix A. List of Acronyms

ABM	Agent-based Model (Modeling)
ACC	Air Combat Command
AOC	Air Operations Center
AOO	Area of Operations
C3I	Command, Control, Computers and Information
CAP	Combat Air Patrol
CONOPS	Concept of Operations
EO/IR	Electro-Optical/Infrared
F2T2	Find-Fix-Track-Target
GMTI	Ground Moving Target Indicator
HPC	High Performance Computer
JSOW	Joint Standoff Weapon
LTL	Local Target List
MANA	Map Aware Non-Uniform Automata
MOP	Measure of Performance
OIF	Operation Iraqi Freedom
OODA	Observe-Orient-Decide-Act
P_d	Probability of Detection
P_{id}	Probability of Identification
P_k	Probability of Kill
SAR	Synthetic Aperture Radar
SEAS	System Effectiveness and Analysis Simulation

TAO	Tactical Area of Operations
TEL	Transporter/Erector/Launcher
TPL	Tactical Programming Language
UAV	Unmanned Aerial Vehicle

Appendix B MOP Data

MOP 1 Baseline and all Test Cases

Run	Baseline	No F-16s	No UAVs	No Satellites
1	14	15	17	5
2	12	18	18	7
3	16	8	14	4
4	16	13	14	9
5	19	15	17	3
6	14	12	11	4
7	17	15	17	7
8	15	16	12	4
9	15	20	14	5
10	14	10	20	6

MOP 2 Baseline and Test Cases two and three

Run	Baseline	No UAVs	No Satellites
1	9	11	7
2	9	16	10
3	12	16	13
4	15	20	10
5	13	16	8
6	12	16	7
7	13	18	6
8	10	15	13
9	7	18	9
10	13	12	8

MOP 3 Baseline and Test Cases two and three

Run	Baseline	No UAVs	No Satellites
1	24	26	21
2	22	30	17
3	28	29	17
4	25	29	17
5	24	27	18
6	25	27	16
7	26	29	14
8	26	29	19
9	23	27	18
10	24	26	18

Bibliography

Altenburg K., J. Schlecht, and K. Nygard. “An Agent-based Simulation for Modeling Intelligent Munitions.” In *Proceedings of the Second WSEAS International Conference on Simulation, Modeling and Optimization*, Skiathos, Greece (Sep). Available at http://www.cs.ndsu.nodak.edu/nygard/research/munitions. 2002

Banks, Jerry, et al (2005). Discrete-Event System Simulation (pp. 354-355). New Jersey: Pearson Prentice Hall.

Berg, David, "Personal Correspondence." Air Combat Command, Langley AFB, VA, October 2005 – March 2006.

Brown, Terry, and Martin, Eric. "Assessing the Mission Effectiveness of Morphing Aircraft Structures Technologies in Hunter/Killer Operations." Briefing to the 73 Military Operations Research Society Symposium, 2005.

Bullock R., McIntyre G., and Hill R. "Using Agent-Based Modeling To Capture Airpower Strategic Effects." Proceedings of 2000 Winter Simulation Conference. 2000.

Carl, R.G. *Search Theory and U-Boats in the Bay of Biscay*. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH: MS thesis, AFIT/GOR/ENS/03-05, March 2003.

Cormen, Thomas H. et al. *Introduction to Algorithms* (2nd Edition). Boston: McGraw-Hill Book Company, 2003.

DeStefano, G.V. *Agent Based Simulation: SEAS Evaluation of DoDAF Architecture*. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH: MS thesis, AFIT/GOR/ENS/04-05, March 2004.

Ferber, J. *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence.* Harlow, England: Addison-Wesley, 1999.

Hassler, Robert T. *An Excel-based Surveillance Planning and Scoring Tool for the SCUD Hunting Mission*. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH: MS Graduate Research Project, AFIT/GOS/ENS/05-08, June 2005.

Illachinski, Andy. "Irreducible Semi-Autonomous Adaptive Combat (ISAAC): An Artificial-Life Approach to Land Combat," *Military Operations Research*, Vol. 5, No. 3: 29-46 (2000).

Jane's International Defence Review. "Sky-high Surveillance Realigns the Battlefield." Edition 1997, v. 030, issue 009, p 65.

Jane's International Defence Review. "The All-seeing Eye Above." December 01, 2002.

Jensen, Paul A. and Bard, Jonathan F. *Operations Research: Models and Methods*. New York: John Wiley & Sons, 2003.

Law, Averill M. and W. David Kelton. Simulation Modeling and Analysis, 3rd Ed. Boston: McGraw-Hill, 2000.

Miller, J.O. OPER 672: Combat Modeling II. Classroom Lecture Slides. 2005

Price, Joseph C. *Game Theory and U-Boats in the Bay of Biscay*. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH: MS thesis, AFIT/GOR/ENS/03-18, March 2003.

Reynolds, Craig W. "Flocks, Herds, and Schools: A Distributed Behavioral Model." Computer Graphics, Vol. 21, No. 4: 25 – 34, July 1987.

Ripley, Tim. "Scud Hunting: Counter-force Operations against Theatre Ballistic Missiles," *Bailrigg Memorandum 18*. Centre for Defence and International Security Studies (CDISS), Lancaster University, 1996.

Rocha, L.M. *From Artificial Life to Semiotic Agent Models: Review and Research Directions.* Los Alamos National Laboratory report, LA-UR-99-5475. Los Alamos NM, 1999.

Russell S. and Norvig P. *Artificial Intelligence: A Modern Approach.* New York: Prentice-Hall, Inc., 1995.

SPARTA, Inc.: Home Page. Retrieved Nov. 12, 2005, from SEAS Web site: http://www.teamseas.com/ (2005).

SPARTA, Inc.: Introduction. Retrieved Nov. 12, 2005, from SEAS Web site help files: http://www.teamseas.com/resources/downloads.html (2005).

SPARTA, Inc.: Sensor. Retrieved Nov. 12, 2005, from SEAS Web site help files: http://www.teamseas.com/resources/downloads.html (2005).

Stockton, R. G. *Models Versus Analysis*. Discussion Paper distributed within the Center for Army Analysis, May 1987.

Taylor, James G. *Hierarchy-of-Models Approach for Aggregated-Force Attrition.* Proceedings of the 2000 Winter Simulation Conference. 2000: 925 – 932.

Widdowson, Brian. "Project Albert: Developing Capabilities, Transitioning to Applications." Briefing for USMC M&S Review, 4 Nov 03.

Withrow, Melissa and Sanders, Brian, PhD. "Morphing Wing Technology." AFRL Horizons. Retrieved Nov. 15, 2005, from AFRL Horizons Web site, reference document http://www.afrlhorizons.com/Briefs/Jun05/VAH0502.html (2005).

Young, M.J. "Agent-Based Modeling and Behavioral Representation." Air Force Research Laboratory report, HE-00-09. Wright-Patterson AFB OH. 2000.

Vita

Captain Jeffrey E. Rucker graduated from Sumner Academy of the Arts and Sciences in Kansas City, Kansas. He began his undergraduate studies pursuing the biological sciences. After marrying, he turned his academic attention towards his passion: mathematics. He graduated with his bachelor's degree in mathematics from California State University in Northridge, California. Shortly thereafter, he completed his Air Force commissioning via Officer Training School in 2001.

His first assignment was to Detachment 4, Air Force Operational Test and Evaluation Center (AFOTEC) in Colorado Springs, Colorado. At that assignment, he was introduced to the world of space-based navigation as the lead analyst for the Global Positioning System (GPS). He was responsible for analysis as the GPS system was undergoing modernization. He also took on analyst responsibilities with the MILSTAR test team.

He departed AFOTEC in August 2004 and entered the Graduate School of Engineering and Management, Air Force Institute of Technology (AFIT), to pursue his Master's Degree in Operations Research. Upon graduation from AFIT, he will report to Headquarters, Air Combat Command, Langley AFB, Virginia, to join their analysis division.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
24-02-2006	**Master's Thesis**	Jun 2005 – Mar 2006

4. TITLE AND SUBTITLE

USING AGENT-BASED MODELING TO SEARCH FOR ELUSIVE HIDING TARGETS

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Rucker, Jeffrey, E., Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 642
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GOR/ENS/06-16

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

SAF/XCOM
ATTN: Lt Col Robert Boyles
1570 Air Force Pentagon DSN 425-5057
Washington, DC 20330-1570 e-mail:robert.boyles@pentagon.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The SCUD hunt problem that emerged during Operation Desert Storm has become a source of great interest to major commands like Air Combat Command. One of the metrics used to measure the effectiveness of our operations in a SCUD hunt is time to detect and target. We use the agent-based System Effectiveness and Analysis Simulation (SEAS) to provide a simulation environment in which all the elements of a SCUD hunt mission can adequately be modeled. Our Blue Force agents are modeled as multirole fighters, satellites and unmanned aerial vehicles (UAV) with various sensor capabilities. The Red Force agents are modeled as the SCUD transporter/erector/launcher (TEL). Particular interest is paid to the effectiveness of various sensors modeled in a set of scenarios following an experimental design. Four measures of performance (MOP) were fashioned to provide insight into the contribution of sensors at work in a SCUD hunt. These MOPs were evaluated to show any statistically significant differences between various mixes of sensors.

15. SUBJECT TERMS

Agent-based modeling, SEAS, System Effectiveness and Analysis Simulation, SCUD hunt, elusive hiding target, sensor, design of experiments, measure of performance, MOP

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			John O. Miller, PhD, (ENS)
U	U	U	UU	100	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-3636, ext 4326; e-mail: john.miller@afit.edu