NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

MATHEMATICAL ANALYSIS OF ALGORITHMS WITHIN MANA

by

James R. Williams

June 2014

Thesis Advisor:	Bard Mansager
Second Reader:	Carlos Borges

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 2014	Master's Thesis

4. TITLE AND SUBTITLE
MATHEMATICAL ANALYSIS OF ALGORITHMS WITHIN MANA

5. FUNDING NUMBERS

6. AUTHOR(S) James R. Williams

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943–5000

8. PERFORMING ORGANIZATION REPORT NUMBER

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

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

MANA (Map Aware, Non-uniform, Automata) is an agent-based, time-stepped, stochastic mission-level modeling environment developed by the New Zealand Defense Technology Agency (DTA).

While the MANA user manual goes into detail about setting up a scenario and navigating the user interface, it does not discuss some of the underlying mathematical procedures and algorithms resulting in many individuals utilizing MANA to analyze military operations without necessarily understanding how the results are achieved.

The purpose of this thesis is to explore the mathematical formulas that MANA utilizes in an effort to aid in creating a more informed understanding of results reached by MANA. This work is intended as a supplement to the current MANA user manual.

We will investigate how manipulating the parameters of the squads' influence behavior on the battlefield. The format will follow a militarily oriented thought process of shoot, move and communicate, investigating mathematically how results are reached within the model. At the conclusion, there will be recommendations as to follow-up work.

14. SUBJECT TERMS:
Agent-Based Simulation, Pseudo Random Number Generator, Algorithm, Map Aware Non-Uniform Automata, Combat Model

15. NUMBER OF PAGES
105

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

NSN 7540–01–280–5500

Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

MATHEMATICAL ANALYSIS OF ALGORITHMS WITHIN MANA

James R. Williams
Major, United States Army
B.S., Greensboro College, 2004

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 2014

Author: James R. Williams

Approved by: Bard Mansager
Thesis Advisor

Carlos Borges
Second Reader

Carlos Borges
Chair, Department of Applied Mathematics

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

MANA (Map Aware, Non-uniform, Automata) is an agent-based, time-stepped, stochastic mission-level modeling environment developed by the New Zealand Defense Technology Agency (DTA).

While the MANA user manual goes into detail about setting up a scenario and navigating the user interface, it does not discuss some of the underlying mathematical procedures and algorithms resulting in many individuals utilizing MANA to analyze military operations without necessarily understanding how the results are achieved.

The purpose of this thesis is to explore the mathematical formulas that MANA utilizes in an effort to aid in creating a more informed understanding of results reached by MANA. This work is intended as a supplement to the current MANA user manual.

We will investigate how manipulating the parameters of the squads' influence behavior on the battlefield. The format will follow a militarily oriented thought process of shoot, move and communicate, investigating mathematically how results are reached within the model. At the conclusion, there will be recommendations as to follow-up work.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to first thank Gregory McIntosh of the Defense Technology Agency, whose correspondence greatly aided in my understanding of what was going on behind the scenes in MANA. I would also like to thank Mary McDonald, whose advice was vital in developing the tutorial within the appendix of this thesis. Lastly, I would like to thank my advisor, Bard Mansager, and the rest of the math department at the Naval Postgraduate School for never departing from their goals of helping students.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

In this chapter, a combat model and its purpose are defined. This introductory chapter also discusses the difference between low- and high-resolution models with the advantages and disadvantages of both, provides a brief introduction to the program MANA and provides the purpose of the thesis.

A. MODELING

According to *Engineering Principles of Combat Modeling and Distributed Simulation* [1], modeling is defined as the purposeful abstraction and simplification of a real or imagined system to provide insight into a problem or question. Combat modeling attempts to perform these actions as they relate to combat scenarios and answer defense related questions [1]. The history of modern combat modeling can be traced back to the early 1900s as Frederick Lanchester first introduced his equations utilizing force size and attrition rate coefficients to describe a battle between two forces. Lanchester's idea was to utilize differential calculus to solve the equations that resulted in identifying the winning side and how many survivors would be left [2].

The original Lanchester equations did not take into account important aspects of the battle to include heterogeneous forces, terrain, battlefield intelligence, spatial variations in forces, and changing attrition rates over time [1]. While one could gain some insight into the overall result of the battle, the lack of detail prevented a robust analysis of the events and revealed no specific details in how the forces fought. Through the years, mathematicians and operations research professionals have attempted to improve on Lanchester's original ideas, incorporating additional variables and structure into the equations. At a certain point, solving the equations by hand becomes infeasible [2]. With the expansion of computing power, the complexity and detail within these models has been allowed to vastly increase over time. This has led to the development of high-resolution models, which can look at many individual entities and their interactions over sub intervals of time within the course of a scenario. The low-resolution models, such as Lanchester equations, utilize relatively simple equations and minimal animations

to look at the results of a battle overall and therefore do not require as much computing power to evaluate. They do not incorporate the detail necessary to answer many of the questions of interest to military research personnel. This necessitates the use of high-resolution models when analyzing the impact of multiple variables within a scenario.

B. HIGH-RESOLUTION MODELS

The purpose of high-resolution combat models is to closely simulate reality and provide a greater understanding and insight into how elements on the battlefield interact [2]. While a low-resolution model usually only takes into account little more than force size and attrition rates; high-resolution models often allow a user to observe the actions of individual entities [1]. A user can manipulate a vast array of agent behavior and capability characteristics as well as different battlefield conditions within the scenario. This increases the complexity of the determining algorithms and the time necessary to evaluate a single scenario but can provide much greater detail about the events within the scenario at specific points in time. High-resolution models can lead to improved decision making by the user.

One type of high-resolution model is an agent-based simulation. Agent-based simulations do not utilize a central controller to manipulate or maneuver individual agents every move on the battlefield [2]. The agents act autonomously and self-organize as they react to their environment in the model based upon initial inputs creating a "personality." Within agent-based simulations the agent's movements are determined by penalty functions that are calculated at each time step. The agent's actions are guided by "personality" inputs determined by the user based upon the intent of the scenario [2]. The movement functions utilize user-specified properties such as; desired terrain type for movement, desire to move towards an enemy, desire to cluster with friendly forces, desire to adhere to a specified path, etc. [2]. The models also incorporate stochastic or probabilistic elements in the algorithms for detection of enemies, communicating with friendlies, and engaging enemies with weapon systems. Each agent reacts to the "perceived world" based upon their established characteristics, leading to, emergent behavior, which has been shown to possess an uncanny ability to mimic human or

"natural" behavior. Agent-based simulations have been used in attempting to understand natural phenomena such as swarming behavior in biological models, vehicles in traffic, and crowd behavior [1, 2].

C. DISADVANTAGES OF AGENT-BASED SIMULATIONS

Despite the level of detail within the agent-based simulations, there are disadvantages caused by the details they include and exclude [2]. If a user could quantify every variable associated with a real-life battle, including weather conditions, terrain effect, and personality differences, the variable matrix would be infinite. Thus, in any model, there are variables that cannot be quantified and are excluded. Deciding which variables to retain is a difficult task and leads to the inevitable question of what affect a left-out variable would have on the simulation. Also, the correct quantification of the variables and the means to verify and validate them becomes an issue. Kirk Yost put it best when he said "modeling difficulty increases exponentially and explainability goes to zero as the number of [variables in a model] increases [2]."

As would be expected, the increase of detail within the model leads to an increase in complexity of the determining algorithms, leading to an increase in the computing power required to evaluate each subsequent time step. This results in an increased time requirement for individual iterations, making it less likely that an analyst can get a full breadth of results from which to make decisions. With the ever-increasing computing capability, the detail of these models also increases bringing them closer to modeling real life situations. The amount of detail in each agent-based simulation limits the scale at which they can be utilized [2]. These simulations are best used at a mission or engagement level rather than a theater or campaign level where the number of entities could be in the thousands. A limiting element of agent-based models (and MANA, specifically) at campaign and theater levels is that the models do not incorporate logistical operations and requirements effectively [2]. At those levels of planning and execution, sustaining operations are the most essential aspect for waging war. However, at the tactical level, the lack of logistical needs are mitigated by the fact that the model only simulates a relatively short period of time when the combatants can operate without

the effects of logistical attrition on their combat effectiveness. In certain models, the user has the ability to manipulate the starting values of ammunition and fuel, but the models do not have an effective ability to "call in" re-supply in the middle of battle. Leadership is an area that is not modeled well. There are no "commander agents" that can direct the behavior and actions of their "subordinates [2]." This prevents the employment of established military tactics as a rule for maneuver. However the emergent behavior from these models has shown to be quite remarkable, resulting in an impressive repertoire of maneuvers that happen as the simulation progresses [2].

D. MANA INTRODUCTION

MANA (Map Aware, Non-uniform, Automata) is an agent-based, time-stepped, stochastic mission level modeling environment developed by the New Zealand Defense Technology Agency (DTA). The model was based upon two key ideas: that the behavior of the entities within a combat model is a critical component of the analysis of the possible outcomes, and that analysts are wasting time with highly detailed physics-based models for determining force mixes and combat effectiveness [2].

The features of MANA are based upon "squads," which are a group of agents that share common physical and behavioral characteristics. Within the program, there are a vast number of "personality" and capability metrics that can be adjusted by a controller to meet the intent of the scenario. These metrics are utilized within the algorithms that determine movement and within the probabilistic calculations regarding detection, hit, and communication. MANA has a relatively simple user interface that facilitates a lower learning curve for first-time users as they begin to build simulations. It is fast running, which allows for many more iterations, compared to other physics based models, to be completed in a relatively short amount of time. This facilitates a controller in conducting a more robust statistical analysis on the scenario [2].

E. PURPOSE

Many student theses from the Naval Postgraduate School and the SEED (Simulations, Experiments and Efficient Designs) Center have used MANA to analyze military operations for a variety of reasons, possibly, without an understanding all the

mathematical processes being utilized within the model. Describing how these calculations are done and presenting them in a format that is familiar to military personnel will allow researchers to conduct more informed analysis.

The purpose of this thesis is to examine the mathematics of the algorithms utilized within the program MANA. This thesis will provide a greater understanding for users into how and why results are achieved. The evaluation is broken into the basic tenets of combat: shoot, move, and communicate. This work is intended to supplement the current MANA user manual and show how manipulating the parameters of the squads influence their behavior on the battlefield. This thesis also develops a tutorial that walks through how a user creates a scenario where a mechanized Iranian company attacks a light infantry platoon set in a hasty defensive position. The tutorial facilitates a rapid familiarization with the use of the model by explaining some of the intricacies involved with establishing the parameters of the squad's and agent's behavior. The next chapter covers how random numbers are generated within the program and what role they play. The subsequent chapters follow a militarily oriented thought process of shoot, move, and communicate, investigating mathematically how results are reached within the model. At the conclusion, there are recommendations as to follow up work and findings.

THIS PAGE INTENTIONALLY LEFT BLANK

II. MANA RANDOM NUMBERS

In this chapter, the algorithm utilized within MANA to generate random numbers is shown, the reason random numbers are important to the model is explained, and a brief review of modular arithmetic and an example of how a sequence of "random" numbers is developed utilizing a linear congruential generator is also shown.

A. RANDOM NUMBERS

In MANA, random numbers play an important role in determining the outcome of interactions between agents during the scenario. A stochastic model relies on randomness that allows for "realism" in the model. The results can be studied and evaluated by a user to gain insight into the effects changes have within the battle. A true random number sequence is a sequence where each subsequent number is not predictable. True randomness would be preferable in combat models; however, it is nearly impossible for a combat model to consistently generate random numbers. Since true random number sequences are not attainable in combat models, there needs to be another way to generate a "random" number sequences. A pseudo-random number sequence is a sequence of numbers generated from a deterministic algorithm which retains many properties of a true random number sequence despite not being truly random. MANA utilizes the Delphi function "Random" to create pseudo-random numbers [3]. The Delphi "Random" function utilizes a linear congruential algorithm [4] to generate the pseudo-random numbers utilized within MANA calculations. The algorithm uses an initial number, or "seed," to initiate the sequence of "random" numbers. MANA allows a user to input the "seed" value. The utilization of a pseudo-random number generator allows a user to input the same initial seed so MANA will replicate the same exact results of the scenario at each time step. This feature may prove useful in evaluating an iteration of a scenario that has outlying results. The initial seed is visible in the Heads up Display (HUD) on the right hand side of the screen, and can be manually entered by a user.

B. LINEAR CONGRUENTIAL GENERATOR

A linear congruential generator is a classic random number generator developed by D. H. Lehmer in 1948 and has become widely used in a number of different areas [5–8]. This is due to the fact that they are relatively simple to understand and are not computationally expensive to execute by a computer. This is important for the model as there could be thousands of calculations per second that need a random numbers. A more computationally expensive algorithm may slow the process to the point where the model would be unable to operate effectively. Linear congruential generators are defined by the following recursive relation [5, 7, 9, 10]:

$$X_{n+1} = (a * X_n + c) \, \boldsymbol{mod} \, M$$

Repeatedly evaluating this relation provides a sequence of numbers which are the "random" numbers that MANA uses. The initial seed is the X_0 value and each subsequent calculation replaces the X_0 with the previous X_1. This algorithm is linear in that each subsequent term of the sequence is defined as a linear function of the previous term in the sequence [10].

$$X_1 = (a * X_0 + c) \, \boldsymbol{mod} \, M$$

To make sense of this equation a quick review of modular arithmetic is shown. First, start with how to express division between two numbers where the quotient does not divide the numerator such as $\frac{12}{5}$. In this case, there is a remainder when 12 is divided by 5. There exists a number that multiplies 5 such that the remainder is between 0 and 4. Using this concept the division algorithm is shown.

$$12 = 2 * 5 + 2$$

$$y = dx + r$$

The above equation defines y as the dividend, d as the divisor, x as the quotient, and r as the remainder.

In modular arithmetic the following expression is used to re-define the variables from the division algorithm where d becomes the modulus:

$$r = y \ \boldsymbol{mod}\ x$$

In the linear congruential generator "M" is called the modulus of the equation. The value "a" is known as the multiplier, "c" is the increment, X_0 is the initial seed and X_1 is the result when reduced by the modulus [5, 6, 8]. Because the algorithm utilizes modular arithmetic, it can take a negative seed value and always produce a positive integer in return. A more in depth review of modular arithmetic can be found in the text *Discrete Mathematics and its Applications* by Kenneth H. Rosen [11].

When the X_n value is computed, it is then reduced modulus 100 to return a value between 0 and 99. If the desired result is between 0 and 1 then the resulting value of X_n is divided by the M value. The following is an example of how the generator works to provide two numbers of the sequence.

To demonstrate how the generator works we utilize 16 as the modulus, which is 2^4, a = 7, and c = 1. With this generator it is important in which values for "a" and "c" are chosen, the reason for why this is discussed in a subsequent paragraph. The value for X_0 = 35 this results in the following equations:

$$X_1 = (7 * 35 + 1)\ \boldsymbol{mod}\ 16$$
$$X_1 = (246)\ \boldsymbol{mod}\ 16$$
$$X_1 = 6$$
$$X_2 = (7 * 6 + 1)\ \boldsymbol{mod}\ 16$$
$$X_2 = (43)\ \boldsymbol{mod}\ 16$$
$$X_2 = 11$$

Thus, the resulting sequence for X_0, X_1, X_2, etc., is {35, 6, 11, ….}

Within MANA there is a potential requirement for large quantities of random numbers. This necessitates that you would need a sufficiently large modulus so as to avoid a repeated sequence of random numbers in a long simulation. In MANA, the modulus utilized is 2^{32} which will return approximately 4.3 billion numbers before repeating. One potential pitfall of this algorithm is that if the values for "a" and "c" are not chosen correctly than the algorithm will fail to reach the full potential size of the sequence [12]. The maximum quantity of values is reached if and only if [6, 13]:

- "c" and "M" are relatively prime
- (a – 1) is divisible by the prime factors of M and
- (a – 1) is a multiple of 4 if M is a multiple of 4

The values for "a" and "c" can differ between programs utilizing the same algorithm to generate pseudo-random numbers. There are $2^{2^{37}}$ unique sequences of numbers that can be produced using 2^{32} as the modulus.

Another reason that MANA utilizes this algorithm is that computers store integers in binary very efficiently. Since division is a relatively slow operation for a computer to conduct compared to addition, subtraction or multiplication, finding the remainder, mod m, must be made more efficiently. A property of binary numbers, when reduced in modular arithmetic, is that the computer will take the k lowest-order bits in the binary representation to return the value mod m where 2^k is the modulus [5]. An example is 22 mod 16 we immediately see that $16 = 2^4$ and that 22 is expressed as 10110 in binary. The 4 lower-order bits of 22 are 0110 which is 6 expressed in binary. Utilizing the division algorithm it is shown that 22 = 1*16 + 6. Thus, 22 mod 16 is 6. One potential issue with this algorithm is that within the sequence of numbers generated the lowest order bits alternate between 0 and 1, return alternating even and odd numbers in the sequence. [6]

Since it has been shown that the sequence of random numbers generated by the linear congruential generator do not share some important statistical properties that true random number sequences possess [5, 8]; and investigation into if this correlation

between numbers in the sequence affects the scenario and skews the distributions of results requires analysis.

Random numbers are utilized in a large number of situations within each time step of MANA. This includes moving and firing order, detection of an agent, where to place an agent at the start of the scenario, whether an agent is hit after being shot at, and as a tie breaker for available moves with the same penalty value.

THIS PAGE INTENTIONALLY LEFT BLANK

III. MOVEMENT

In this section, how the terrain is represented and how it affects the maneuver of the agents in the simulation is shown as well as how the determining algorithm for movement and how MANA calculates the maneuver for an agent at each time step. A more robust example of the calculations that determine movement than the published manual provides is completed. The changes between MANA V and MANA 4 are also shown.

A. INTRODUCTION

MANA is a high-resolution model and seeks to simulate individual weapon systems as they maneuver on the battlefield. Capturing a realistic representation of this movement is essential for the validation of the model as a viable simulation for combat. MANA utilizes the user established personality weightings to calculate penalty values for each available movement location for each agent at each time step [3]. The subsequent movement results in behavior that can be characterized as emergent behavior and has been shown to mimic actual maneuver more closely than might be expected [2].

Terrain has a great impact on a high-resolution model at the engagement level. Terrain is vitally important to an individual tank maneuvering against an enemy in terms of where it can maneuver and what it can sense and shoot. The tank maneuvers through the battlefield avoiding restrictive terrain, and utilizing intervisibility lines to gain an advantage on its enemy. While MANA does not explicitly model tactics, the movement algorithm allows each agent to "decide" on the best course of action based upon its perception its surroundings [3].

MANA 4 is an explicit grid model that utilizes a grid to store the required terrain characteristics for each location on the battlefield [1,3]. The environment is represented in a rectangular pattern which allows the model to closely align with military maps and a Euclidean coordinate system which is familiar within the military [3]. Each rectangle is assigned characteristic values for elevation, cover, concealment, and maneuverability (or how fast an agent can move through the terrain) based upon the type of terrain that is

assigned to it [3]. These characteristic values are utilized within the movement algorithm as part of a penalty calculation that determines where each agent moves and can sense at each time step. MANA begins with five basic types of terrain that are pre-set with characteristic values. The basic terrain types are road (easy going), light bush, heavy bush, wall, and hill top [3]. A user can adjust the characteristic values for the each terrain type or establish new terrain types altogether depending on the needs of the scenario. There are three levels of "backgrounds" that influence what is observed on the screen. The first is imagery of a specific area loaded as a picture for cosmetic purposes into the background. The agents maneuver on top of the background and provide spatial context for the user. The terrain map is behind the background image. The user creates the terrain by utilizing the different terrain types within MANA and aligns the specific type to match the effect that the terrain would have on an agent. This terrain map is the actual information that establishes the terrain characteristic values for each grid location. The third is the elevation map. MANA utilizes 256 levels of grey corresponding to changing elevations where white is the highest elevation and black is no elevation [3]. Building the terrain and elevation maps are tedious tasks. It is important that a user represents the desired terrain accurately [3].

B. THE "STEPHEN" ALGORITHMS

The movement algorithm determines a penalty value for all available movement locations for an agent. The agent utilizes a minimax strategy to select where to maneuver, at each time step. MANA can utilize three algorithms separately to determine agent movement. The default algorithm is referred to as the "Stephen" algorithm. The "Gill" algorithm criticized the original "Stephen" algorithm for not taking into account the number of agents within sensor range. Gill introduced two exponent values that acted as a weighting of proximity of other agents relative to the size of the map [14]. The third is the "Path Following" algorithm, which is utilized to increase the realism of aerial movement. "Path" uses a shortest path algorithm as an agent in flight across the battlefield maneuvers from one point to the next. The reason for this is that the effects of terrain do not play into an aerial vehicle's movement.

The standard "Stephen" equation utilized in MANA 4 is the following:

$$P_i = 1 + \frac{\sum_{m=1}^{M}\left(D_N(m) - D_O(m)\right)D_W(m)}{100\sum_{l=1}^{M} D_W(l)}$$

The number of entities per personality setting whose distance from the agent is to be used in the penalty equation is indexed by, m = 1, 2,…, M where M is the total number of entities. D_N and D_O are the new and old distances, respectively, from the agent to each entity. D_W is the weighting factor:

$$D_W = Round\ (BDL - D_O)$$

BDL is the length of the main diagonal on the battlefield. This is a normalization factor which reduces the effect of different sizes of maps on agent behavior. The value within the parenthesis is rounded to the nearest integer. P_i is the penalty value for each personality settings. The penalty for moving to any grid location is the sum of 32 penalty values corresponding to the 32 personality settings multiplied by the personality weighting established by the controller. Once the P_i value is calculated, it is multiplied by W_i, which is the weight given to each personality weight. The resulting value is summed with up to 32 other personality settings within sensor range of the agent moving. This provides the penalty value for the movement location.

$$penalty = \sum_{i=1}^{N} W_i P_i$$

The agent seeks to move into the location with the minimum penalty assigned. [3]

C. THE "GILL" ALGORITHM

A. Gill noted several issues with the original algorithm utilized in the first two of versions of MANA and proposed a new algorithm which was adopted by MANA as the "Gill" algorithm [14]. The first issue was that the original "Stephen" algorithm treated all agents as if they were 100 units away, regardless of the actual distance. The second issue

noted was that the number of agents detected was not considered. This means that an agent would maneuver against one enemy the same way it would against 50. The "Gill" algorithm suggested a means to correct these issues by incorporating variables α and r, where α is the number of agents detected, and r is the distance between agents. It should be noted that the "Stephen" algorithm attempts to mitigate these criticisms with the utilization of the D_W calculation in the denominator of the equation. When selecting the values for α and r, a controller should refer to the paper "Validation of agent-based distillation movement" [15], which goes into depth as to how different values effect the movement of the agents. Sensitivity analysis should be done by the user utilizing both the "Stephen" and "Gill" algorithms to observe any significant variations in the results of the scenario. Gill notes that the results are more sensitive to adjustments to the r variable than α. The original "Gill" algorithm proposed is the following:

$$Penalty = \frac{W_R}{R^{\alpha}} \sum_{i=1}^{R} \left(\frac{D_{i\,new} - D_{i\,old}}{D_{i\,old}} \right)^r + W_F \left(\frac{D_{F\,new} - D_{F\,old}}{D_{F\,old}} \right)^r$$

The following are the definitions for the variables in the proposed "Gill" algorithm [14]:

- R: Number of red agents within sensor range
- W_R: Weighting towards red agents
- $D_{i,new}$: Distance to the i^{th} red agent from the new location
- $D_{i,old}$: Distance to the i^{th} red agent from the current (old) location
- W_F: Weighting towards the "goal"
- $D_{F,new}$: Distance to the "goal" from the new location
- $D_{F,old}$: Distance to the "goal" from the current (old) location
- r: User defined non-negative variable
- α: User defined non-negative variable between zero and one

Where the "goal" is the destination waypoint established by the user.

MANA adapted the proposed algorithm into the following form [4]:

$$Penalty = M^{-\alpha} \sum_{m=1}^{M} \left(\frac{D_N(m) - D_O(m)}{D_O(m)} \right)^r$$

It should be noted that the value within the summation raised to the exponent r will be a negative number if the distance between the new locations to the influencing factor is closer than the old location. This presents a problem when attempting to evaluate the expression when r is set to the recommended value of (0.5). MANA would be attempting to take the square root of a negative number which results in a complex number. However, if the expression is negative, MANA takes the absolute value of the number under r. The resulting value is then negated [14, 16]. An example calculation is completed later in the chapter that demonstrates how this is done.

There are three movement constraints that a controller can impose on agents within the scenario which are calculated as personality weightings within the movement algorithm [3]. The combat constraint ensures that an agent will not advance on an enemy without an established numerical superiority. The cluster constraint ensures that each agent will not group with more agents than the established amount. This prevents a "hoard" from moving across the battlefield. The advance constraint prevents an agent from advancing without support. The agent moves toward friendly forces until the appropriate numbers of friendly agents are within sensor range before advancing.

D. BEST MOVE CHOICE

In actual movement it cannot be expected that every soldier makes the correct movement every time. MANA 4 replicates this by including a small probability that the agent does not move into the position with the least penalty. This is likely accomplished by adding a small, uniformly distributed, amount to the final totals for each location. This ensures that while the agent usually maneuvers into the best location, it remains unlikely that the agent would move into what would be considered a completely irrational position.

E. "STEPHEN" ALGORITHM EXAMPLE

The following example of the "Stephen" algorithm expands on the one seen within the MANA 4 manual. Assume a 4x4 board where there is only one object influencing the movement of the blue agent represented as "B". The blue agent is maneuvering against the personality weight "enemy threat 1", represented as R_1. The weighting given to this agent is 15, indicating a desire to maneuver towards the threat. The blue agent has three potential positions to move into as well as staying in the original location. Figure 1 is the environment. Figure 1 is the example 1 battlefield set-up.

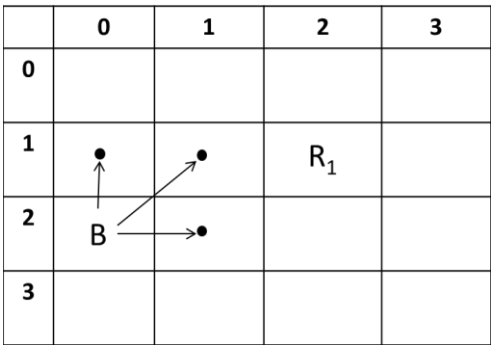

Figure 1. Simple Battle Field Environment for Example 1, after [3].

Table 1 displays the calculation values within the algorithm for example 1.

MVT Location	Dist. To R_1	$D_n(R_1) - D_o(R_1)$	$D_w(R_1)$
(0,1)	2	$2 - \sqrt{5}$	Round $(4\sqrt{2} - \sqrt{5})$
(1,1)	1	$1 - \sqrt{5}$	Round $(4\sqrt{2} - \sqrt{5})$
(1,2)	$\sqrt{2}$	$\sqrt{2} - \sqrt{5}$	Round $(4\sqrt{2} - \sqrt{5})$
(0,2)	$\sqrt{5}$	0	Round $(4\sqrt{2} - \sqrt{5})$

Table 1. Stephen Equation Values for First Example 1.

Since there is only one agent influencing the equation the summation drops off and the resulting equations for the movement locations are displayed in Table 2.

$P_{R_1}(0,1) = 1 + \frac{(2-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100(4\sqrt{2}-\sqrt{5})}$	$P_{R_1}(1,1) = 1 + \frac{(1-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100(4\sqrt{2}-\sqrt{5})}$	$P_{R_1}(1,2) = 1 + \frac{(\sqrt{2}-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100(4\sqrt{2}-\sqrt{5})}$
$P_{R_1}(0,1) = 1 + \frac{(-.236)}{100}$	$P_{R_1}(1,1) = 1 + \frac{(-1.236)}{100}$	$P_{R_1}(1,2) = 1 + \frac{(-.822)}{100}$
$P_{R_1}(0,1) = .997$	$P_{R_1}(1,1) = .988$	$P_{R_1}(1,2) = .992$

Table 2. Values for Movement Locations in Example 1.

Once the P values are calculated the weighting value is applied to determine the penalty value for each movement location.

- $Penalty\ (0,1) = 15 * .997 = 14.95$
- $Penalty\ (1,1) = 15 * .988 = 14.82$
- $Penalty\ (1,2) = 15 * .992 = 14.88$
- $Penalty\ (0,2) = 15 * 1.00 = 15.00$

The resulting move should be into position (1, 1) as that is the least penalized position. Staying in the original location is the most penalized position in this case. This makes sense as the weighting for R_1 indicated that there was a desire for the blue agent to advance towards "enemy threat 1". The location that put the blue agent closest was in position (1, 1).

In this next example, the number of agents of the battlefield is expanded. The new agents are specified as having a negative personality weighting. The new agents are considered "enemy threat 2." The weighting for "enemy threat 2" is set to -5, which indicates a small repulsion for the blue agent. Figure 2 is the environment in the second example.

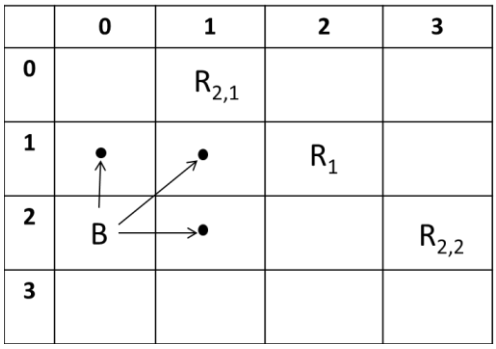

Figure 2. Simple Battlefield Environment for Example 2, after [3].

Table 3 displays the calculated values for the Stephen Equation, and Table 4 displays the resulting values for movement locations in example 2.

MVT Location	Dist. To R_1	Dist. To $R_{2,1}$	Dist. To $R_{2,2}$	$D_n(R_1) - D_o(R_1)$	$\sum_{m=1}^{2} D_n(m) - D_o(m)$	$\sum_{l=1}^{2} D_W(l)$	$D_w(R_{2,1}), D_w(R_{2,2})$
(0,1)	2	$\sqrt{2}$	$\sqrt{10}$	2 - $\sqrt{5}$	$(\sqrt{10} - 3) + (\sqrt{2} - \sqrt{5})$	Round $\left((4\sqrt{2} - 3) + (4\sqrt{2} - \sqrt{5})\right)$	Round (4$\sqrt{2}$ - $\sqrt{5}$), Round (4$\sqrt{2}$ -3)
(1,1)	1	1	$\sqrt{5}$	1 - $\sqrt{5}$	$(\sqrt{5} - 3) + (1 - \sqrt{5})$	Round $\left((4\sqrt{2} - 3) + (4\sqrt{2} - \sqrt{5})\right)$	Round (4$\sqrt{2}$ - $\sqrt{5}$), Round (4$\sqrt{2}$ -3)
(1,2)	$\sqrt{2}$	2	2	$\sqrt{2}$ - $\sqrt{5}$	$(2 - 3) + (2 - \sqrt{5})$	Round $\left((4\sqrt{2} - 3) + (4\sqrt{2} - \sqrt{5})\right)$	Round (4$\sqrt{2}$ - $\sqrt{5}$), Round (4$\sqrt{2}$ -3)
(0,2)	$\sqrt{5}$	$\sqrt{5}$	3	0	0	Round $\left((4\sqrt{2} - 3) + (4\sqrt{2} - \sqrt{5})\right)$	Round (4$\sqrt{2}$ - $\sqrt{5}$), Round (4$\sqrt{2}$ -3)

Table 3. Stephen Equation Values for Example 2.

$P_{R_2}(0,1) = 1 + \frac{(\sqrt{10}-3)(4\sqrt{2}-3) + (\sqrt{2}-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100\left((4\sqrt{2}-3)+(4\sqrt{2}-\sqrt{5})\right)}$ $P_{R_2}(0,1) = 1 + \frac{(.4868) - (2.466)}{600}$ $P_{R_2}(0,1) = .9967$	$P_{R_2}(1,1) = 1 + \frac{(\sqrt{5}-3)(4\sqrt{2}-3) + (1-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100\left((4\sqrt{2}-3)+(4\sqrt{2}-\sqrt{5})\right)}$ $P_{R_2}(1,1) = 1 + \frac{(-2.2971) + (-3.7082)}{600}$ $P_{R_2}(1,1) = .9900$	$P_{R_2}(1,2) = 1 + \frac{(2-3)(4\sqrt{2}-3) + (2-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100\left((4\sqrt{2}-3)+(4\sqrt{2}-\sqrt{5})\right)}$ $P_{R_2}(1,1) = 1 + \frac{(-3) + (-.7082)}{600}$ $P_{R_2}(1,1) = .9938$
$P_{R_2}(0,1) = 1 + \frac{(\sqrt{10}-3)(4\sqrt{2}-3) + (\sqrt{2}-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100\left((4\sqrt{2}-3)(4\sqrt{2}-\sqrt{5})\right)}$ $P_{R_2}(0,1) = 1 + \frac{(.4868) - (2.466)}{600}$ $P_{R_2}(0,1) = .9967$	$P_{R_2}(1,1) = 1 + \frac{(\sqrt{5}-3)(4\sqrt{2}-3) + (1-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100\left((4\sqrt{2}-3)(4\sqrt{2}-\sqrt{5})\right)}$ $P_{R_2}(1,1) = 1 + \frac{(-2.2971) + (-3.7082)}{600}$ $P_{R_2}(1,1) = .9900$	$P_{R_2}(1,2) = 1 + \frac{(2-3)(4\sqrt{2}-3) + (2-\sqrt{5})(4\sqrt{2}-\sqrt{5})}{100\left((4\sqrt{2}-3)(4\sqrt{2}-\sqrt{5})\right)}$ $P_{R_2}(1,1) = 1 + \frac{(-3) + (-.7082)}{600}$ $P_{R_2}(1,1) = .9938$

Table 4. Values for Movement Locations in Example 2.

- $Penalty\ (0,1) = (15 * .997) + (-5 * .9967) = 9.97$
- $Penalty\ (1,1) = (15 * .988) + (-5 * .9900) = 9.87$
- $Penalty\ (1,2) = (15 * .992) + (-5 * .9938) = 9.91$
- $Penalty\ (0,2) = (15 * 1.00) + (-5 * 1) = 10$

With the addition of the two other agents into the scenario, the best location for the blue agent to maneuver remains position (1, 1). This still makes sense because the desire to maneuver towards R_1 is three times the repulsion of R_2 . However, it is seen that the difference between the four positions is much closer with the introduction of the R_2 objects than it was with only the R_1.

F. "GILL" ALGORITHM EXAMPLE

The "Gill" algorithm is used in this example to see the differences between it and the "Stephen" algorithm. The expectation is that both algorithms reach the same conclusion on the best location to move.

In this case, the blue agent is maneuvering against the personality weight "enemy threat 1" represented as R_1 with a weight of 15. Again, three potential places are considered for the blue agent to maneuver into. Figure 3 is the example 3 battlefield set-up.

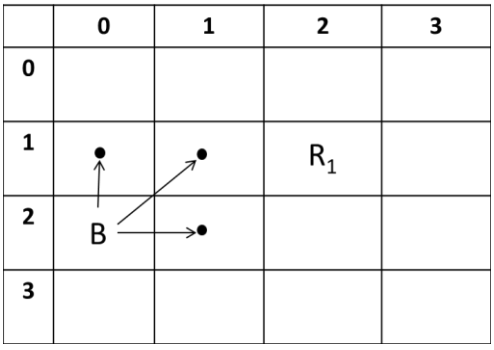

Figure 3. Simple Battlefield for Example 3, after [3].

$$P_{R_1} = M^{-\alpha} \sum_{m=1}^{1} \left(\frac{D_N(m) - D_O(m)}{D_O(m)} \right)^r$$

Table 5 displays the calculated movement values for each location for example 3.

$P_{R_1}(0,1) = 1^{-1}\left(\frac{2-\sqrt{5}}{\sqrt{5}}\right)^{.5}$	$P_{R_1}(1,1) = 1^{-1}\left(\frac{1-\sqrt{5}}{\sqrt{5}}\right)^{.5}$	$P_{R_1}(2,1) = 1^{-1}\left(\frac{\sqrt{2}-\sqrt{5}}{\sqrt{5}}\right)^{.5}$
$P_i(0,1) = -(.1056)^{.5}$	$P_i(1,1) = -(.5527)^{.5}$	$P_i(0,1) = -(.3675)^{.5}$
$P_i(0,1) = -.325$	$P_i(1,1) = -.743$	$P_i(0,1) = -.606$

Table 5. Values for Movement Locations in Example 3.

- $Penalty\ (0,1) = (15 * -.325) = -4.875$
- $Penalty\ (1,1) = (15 * -.743) = -11.145$
- $Penalty\ (1,2) = (15 * -.606) = -9.090$

The least penalized position is still (1, 1), however, there is a much greater disparity between the penalties of all three positions than what was seen in the "Stephen" algorithm.

In the next case, the "Gill" algorithm is utilized with the introduction of two other agents; "enemy threat 2" which are represented as $R_{2,1}$ and $R_{2,2}$. The weighting for "enemy threat 2" will remain -5, indicating the blue agent is repulsed by them. Figure 4 is the example 4 battlefield set-up.

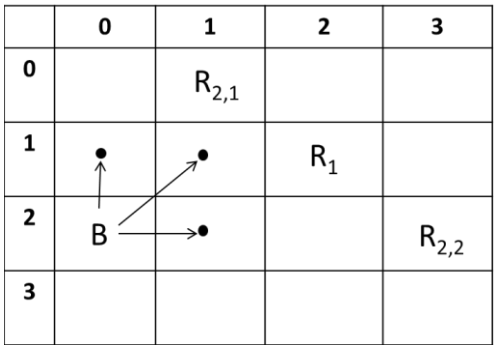

Figure 4. Simple Battlefield for Example 4, after [3].

$$P_{R_2} = 2^{-1} \sum_{m=1}^{2} \left(\frac{D_N(m) - D_O(m)}{D_O(m)} \right)^{.5}$$

$$P_{R_2} = \frac{1}{2} \left(\left(\frac{\sqrt{2} - \sqrt{5}}{\sqrt{5}} \right)^{.5} + \left(\frac{\sqrt{10} - 3}{3} \right)^{.5} \right) = \frac{-6.06 + .0541}{2} = -.330$$

$$P_{R_2} = \frac{1}{2} \left(\left(\frac{1 - \sqrt{5}}{\sqrt{5}} \right)^{.5} + \left(\frac{\sqrt{5} - 3}{3} \right)^{.5} \right) = \frac{-743 - .504}{2} = -.624$$

$$P_{R_2} = \frac{1}{2} \left(\left(\frac{2 - \sqrt{5}}{\sqrt{5}} \right)^{.5} + \left(\frac{2 - 3}{3} \right)^{.5} \right) = \frac{-.325 - .577}{2} = -.451$$

We combine these P_i values with those calculated from the first previous example to reach the penalty values for each movement location.

$$penalty = \sum_{i=1}^{2} W_i P_i$$

- $Penalty\ (0,1) = (15 * -.325) + (-5) * (-.330) = -3.225$
- $Penalty\ (1,1) = (15 * -.743) + (-5) * (-.624) = -8.025$
- $Penalty\ (1,2) = (15 * -.606) + (-5) * (-.451) = -6.835$

Again, it is shown that the best position to move is into (1, 1). When utilizing the "Gill" algorithm, the more desirable moves have a larger negative value. Adding one to the "Gill" algorithm would put it into the same scheme as with the "Stephen" algorithm where a value of less than one indicates a desirable move and a value greater than one would lead to an undesirable move. This results in the following equation:

$$P_i = 1 + \left(M^{-\alpha} \sum_{m=1}^{1} \left(\frac{D_N(m) - D_O(m)}{D_O(m)} \right)^{r} \right)$$

G. MANA V

MANA V makes several changes to the movement algorithm used in MANA 4. The explicit grid based model is replaced with a vector based scheme [17]. The vector scheme aligns more closely with the natural movement of entities across the battlefield since it is continuous rather than discrete [17]. The vector movement also reduces the computing power required to maneuver agents at each time step and allows for larger maps and more agents within each scenario. The personality characteristics are now separated between enemies, waypoints and terrain features [17]. Each of the three are calculated as their own separate vector. The overall movement vector is calculated utilizing vector arithmetic. Each vector is multiplied by the personality weight and then added together to form the overall movement vector, **F** [17]. MANA V also incorporates the idea of inertia into the maneuver [17]. Most land based scenarios it can be assumed that the default setting for mass is correct. A setting of .01 indicates almost instantaneous acceleration [17]. In the case where a user is attempting to model large ships maneuvering in a naval scenario, adjusting this setting may be appropriate to prevent unrealistic behavior.

In this scheme, each agent and entity has an x and y coordinate value. This facilitates the creation of a movement vector between the agent and each entity influencing the maneuver. The subsequent example demonstrates how this calculation is done later in this chapter.

H. VECTOR ARITHMETIC REVIEW

This section utilizes Stewart's calculus work for the review of vector addition and scalar multiplication of vectors. [18]

The term vector indicates a quantity that has both magnitude and direction and is usually represented by an arrow or directed line segment. The vector consists of an initial point and a terminal point. In the case of MANA V, the initial point is the location of the agent and the terminal point is the location of the influencing entity. In two dimensional space each vector will have to components that correspond to the x and y axis. When the

initial point is located at the origin the terminal point expresses the vector. If the vector is located at any other point then the following equation is used to determine the vector.

$$\vec{a} = < \boldsymbol{x_2} - \boldsymbol{x_1}, \boldsymbol{y_2} - \boldsymbol{y_1} >$$

The values for x_2 and y_2 are the coordinates for the terminal point and the values for x_1 and y_1 are the coordinates for the initial point. When adding two vectors the components within the vector are added together.

$$\vec{a} + \vec{b} = \langle a_1 + b_1 , a_2 + b_2 \rangle$$

Graphically the equation is expressed by the following figure and is known as the Triangle Law in Figure 5.

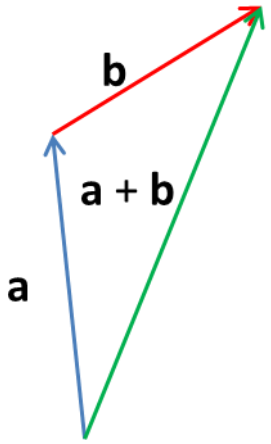

Figure 5. Triangle Law Example, after [18].

The extension of the Triangle Law is displayed in Figure 6 as the Parallelogram Law.

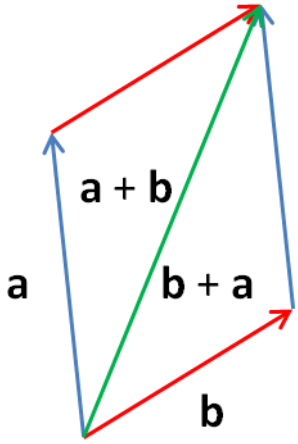

Figure 6. Vector Addition Graphic Example, after [18].

When the vector is multiplied by a scalar value, each component of the vector is multiplied by that value and effectively extends the length of the vector by the amount the vector is multiplied by. This is displayed graphically in Figure 7.

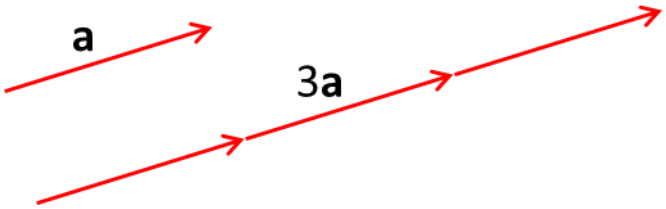

Figure 7. Scalar Multiplication Example, after [18].

I. MANA V EXAMPLE

Utilizing a simple example, how the direction that the agent will move is shown. There are five entities influencing the agent's movement. In this example, real-world distances are not specified and the example only considers the coordinates of the entities on the battlefield. Figure 8 depicts the battlefield set up for the MANA V example.

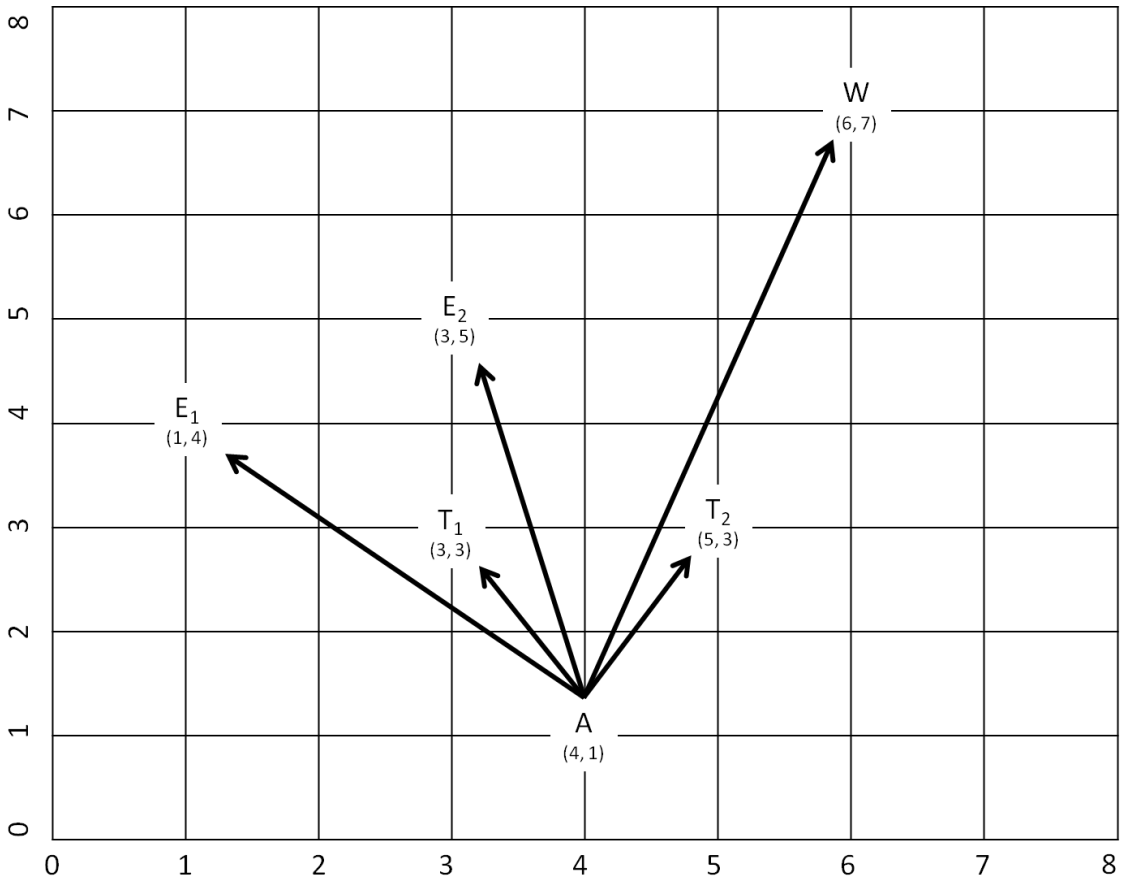

Figure 8. MANA V Example Environment.

The blue agent is annotated as A, and is located at the grid coordinate (4,1). There are two enemies that the agent senses which are annotated as E_1, and E_2. Their coordinate locations are respectively (1,4), (3,5). There are also two terrain features, T_1, T_2 with coordinates (3,3) and (5,3), respectively. The agent also senses a way point, represented as W, which is located at the coordinate (6,7). The personality weightings that are specified for each entity are as depicted in Table 6.

Entity	Personality Weighting
E_1	15
E_2	10
T_1	-20
T_2	-10
W	25

Table 6. MANA V Example Personality Weightings

Table 7 displays the vector values for the MANA V example.

Vector	Vector to Entity 1	Vector to Entity 2
F_E	$F_{E_1} = \langle 1-4,\ 4-1\rangle = \langle -3,\ 3\rangle$	$F_{E_2} = \langle 3-4,\ 5-1\rangle = \langle -1,\ 4\rangle$
F_T	$F_{T_1} = \langle 3-4,\ 3-1\rangle = \langle -1,\ 2\rangle$	$F_{T_2} = \langle 5-4,\ 3-1\rangle = \langle 1,\ 2\rangle$
F_W	$F_W = \langle 6-4,\ 7-1\rangle = \langle 2,\ 6\rangle$	

Table 7. Vector Equations for MANA V Example

$$\boldsymbol{F}_E = \langle -3,\ 3\rangle * 15 + \langle -1,\ 4\rangle * 10 = \langle -55, 85\rangle = \langle -11, 17\rangle * 5$$
$$\boldsymbol{F}_T = \langle -1,\ 2\rangle * -20 + \langle 1,\ 2\rangle * -10 = \langle 10, -60\rangle = \langle 1, -6\rangle * 10$$
$$\boldsymbol{F}_W = \langle 2,\ 6\rangle * 25 = \langle 50, 175\rangle$$
$$\boldsymbol{F} = \langle -55, 85\rangle + \langle 10, -60\rangle + \langle 50, 175\rangle = \langle 15, 190\rangle = \langle 3, 38\rangle * 5$$

Figure 9 displays the resulting movement vector after the calculations are complete.

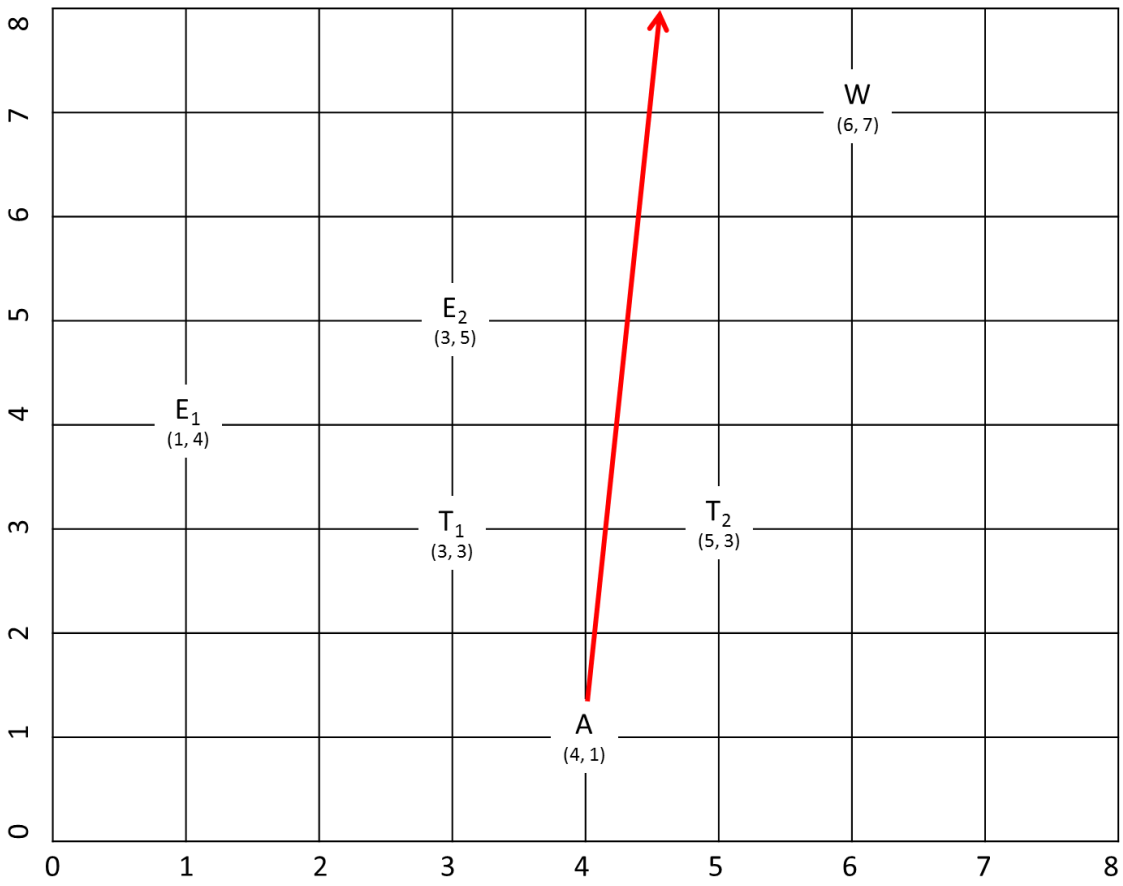

Figure 9. MANA V Example Approximate Movement Vector.

The vector in Figure 9 indicates the direction of movement for the agent at the current time step relative to the influencing entities. Essentially, this movement algorithm acts as a weighted average of the "best" movement directions for the agent at each time step. Careful consideration must be utilized when designating agent characteristics to ensure that the model accurately reflects the intent of the scenario. A controller can manipulate additional characteristics of the agents forcing the agents to a more constrained movement. An example of this is if the user attempts to model a convoy operation where the convoy would not maneuver off the established path.

J. ENGINEERS

Operationally, engineer assets are utilized to influence the maneuver of friendly and enemy forces across the battlefield. While engineer assets are not directly represented

in MANA it is possible to express their effects. This is accomplished by developing new "terrain types" as a part of the terrain map that can represent obstacles employed in a defensive situation. In a scenario where a user attempts to model a deliberate defense, obstacles such as tank ditches and concertina wire can be created with an appropriate reduction to movement of the attacking force. A mine field is tougher to represent, but there are ways to construct such a feature utilizing additional agents that cannot move and have appropriate levels of stealth. There are no "breaching" assets represented in MANA, but the slowed or altered movement of the agents and across the obstacles established by the user can replicate the desired effects with the correct settings.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. SENSE AND SHOOT

This section addresses how agents within MANA engage each other on the battlefield with both direct and indirect weapons, as well as how the agents sense entities and develop a "situational awareness" of the battlefield.

A. SENSE

In combat, all parties attempt to acquire total ground truth. This is a perfect understanding of the location and movement of the opposing forces against whom they can maneuver against most effectively. Rarely in battle does each side achieve a perfect dovetail of perception and ground truth. Thus, accurately modeling perception is vital for a simulation to provide meaningful insight into actions between two forces. Understanding how different aspects within the model affect the perception of the agents on the battlefield is a powerful and important aspect of modeling. MANA possesses a robust capability to model perception by providing each agent up to six sensors that are specified as either "simple" or "advanced [3]." "Advanced" sensors can be offset which simulates tertiary means of sensing such as satellite or other means [3]. A user can adjust detection and classification probabilities to simulate degraded states of sensing within a scenario.

MANA is a continuous sensing model that utilizes a detection rate function to model the time the detection of an agent occurs [19].

$$p(detection) = 1 - e^{\left(\frac{-t}{T}\right)}$$

The t value is the designated time scale with the default being 1.0 second per time step. This can be changed, but the user must be aware of the impacts on the scenario. A time step less than one second per time step would result in very precise movements, however, the run time of the simulation would increase dramatically. If the time step was set to represent more than one second per time step the user runs the risk of the simulation "skipping" events that happen in between time steps. In the advanced sensor

type menu the average time between detections is the T value, which becomes the detection rate. An average time of 5 seconds implies a detection probability of 0.2, where detection probability is:

$$detection\ probability = \frac{1}{T}$$

When an agent is within sensor range a random number between (0, 1) is generated at each time step and detection occurs if:

$$random\ number < 1.0 - e^{\left(\frac{-t}{T}\right)}$$

1. Sensing Derivation

It is an interesting exercise to see where this equation comes from and why it makes sense to use for detection. The following derivation utilized *Mathematical Modeling of Warfare and Combat Phenomenon* by Jeffrey Strickland [20].

MANA models a circular area around an agent from which if another agent enters there is a certain probability that they are detected. Since the opposing agent is relatively small compared to the overall area that is being sensed, the agent can be represented as point within the calculation. The sensing agent is firing sensing "bullets" where if the opposing agent is "hit," then they are detected. The probability an agent is hit is referred to as the single-shot probability of "kill."

$$P_{ssk} = P$$

Then the probability of hitting with one shot out of n shots is:

$$P(n\ shots) = 1 - (1 - P)^n$$

Assuming the joint probability distribution is $p(x, y) = p(x)p(y)$, the covariance $\sigma_{xy} = 0$, a normal probability distribution with mean μ, and standard deviation σ the function is expressed as:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\left[-\frac{(x-\bar{x})^2}{\sigma^2}\right]}$$

Then:

$$p(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{\left[-\frac{(x-x_o)^2}{2\sigma_x^2} - \frac{(y-y_o)^2}{2\sigma_y^2}\right]}$$

In this equation the x and y coordinates are the location of the opposing agent, x_o and y_o are the location of the detecting agent and σ_x and σ_y are the mean standard deviation in the x and y directions respectively. The probability of "hit" within the circle with radius R is:

$$P = \iint_{\sqrt{(x^2+y^2)}<R} p(x,y)dxdy$$

If detection of the opposing agent at (x,y) is denoted by D(x,y), then the unconditional probability of detection is:

$$P_d = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} D(x,y)p(x,y)dxdy$$

Utilizing the "cookie cutter" damage or detection function:

$$D(x,y) = \begin{cases} 1; r \le R \\ 0; r \ge R \end{cases}$$

Since there is no offset and equal variance, we can substitute the probability distribution and "cookie cutter" functions into the detection function resulting in:

$$P_d\left(\frac{R}{\sigma}\right) = \int_0^{2\pi}\int_0^R \frac{1}{2\pi\sigma^2} e^{\left(-\frac{r^2}{2\sigma^2}\right)} r\, dr\, d\theta = \int_0^R \frac{r}{\sigma^2} e^{\left(-\frac{r^2}{2\sigma^2}\right)} dr = 1 - e^{\left(-\frac{r^2}{2\sigma^2}\right)}$$

The limits of the integration are converted into polar coordinates resulting in the probability of "hit" or detection as:

$$P(r) = 1 - e^{\left(-\frac{r^2}{2\sigma^2}\right)}$$

If we substitute t for r^2 and T for $2\sigma^2$ we see the original equation of:

$$p(detection) = 1 - e^{\left(\frac{-t}{T}\right)}$$

To see an example, assume an average time between detections is 5, the time scale is 1.0 seconds, and the uniform random number (0, 1) is .38. The following calculation is completed to determine if the agent is detected.

$$1.0 - e^{\left(\frac{-1.0}{5}\right)} = .18$$
$$.38 > .18$$

In this time step the agent is not detected. In the next time step another random number is generated and the calculation is completed again. When the T value is set to zero this implies a probability of detection close to 1.0 indicating that as soon as an agent is within sensor range it is detected.

2. Elevation

Elevation's role in MANA is similar to the role it plays in actual combat. The sensor height of the agent determines the line of sight when maneuvering through undulating terrain. If an opposing agent is maneuvering on the other side of an elevation feature that blocks the line of sight from the agent, then the agent is unable to identify or classify his opponent. Unless specified by the controller, the weapon system is able to penetrate through a terrain feature with direct fire. However, high explosive ammunition types are able to engage targets by firing "over" the hill or increased elevation.

3. Classification

The probability of classifying an agent once it has been detected is simple within MANA. A probability at a specified distance can be set by a user in the advanced sensor menu. If the probability is set to 1 then once the agent is detected it is be correctly classified. If the probability is less than one another random number is generated where if it is less than the probability specified then it is classified.

B. SHOOT, DIRECT FIRE

Once an agent is detected and classified as an enemy, the opposing agent attempts to engage it with its assigned weapons. MANA models two types of weapons: kinetic and

explosive [4]. Kinetic weapons are bullets, tank rounds, and an anti-tank weapon such as a TOW missile. Explosive weapons within MANA are the traditional indirect fire weapons such as artillery fire and missiles from aircraft. MANA differentiates between the two types by how the probability of being hit is measured. Kinetic weapons calculate hit from the shooter. Explosive weapons calculate hit from the point of impact [3].

In MANA, an agent can be equipped with up to six weapon types with each weapon having its own characteristics and priority of targets [3]. Only one weapon can be designated as being the primary weapon for the agent [3]. In the advanced weapon menu the ranges and probabilities for hit can be specified for each weapon system.

Considering a M1 main battle tank, there are four weapons that the tank can employ. The primary weapon is the 120mm main gun. This is employed against other tanks and armored personnel carriers. The second is the coaxially mounted M240C. While mounted with the main gun the coax (as it is known) is used against a completely different type of enemy, such as dismounted infantry. The third weapon is the commander's M2 heavy machine gun. This is primarily utilized against light vehicles and infantry. The final weapon is the loader's M240B, which is used to target infantry. In MANA when establishing the weapon systems for the M1, the user will select the weapons tab in the "edit squad properties" window.

In the "weapons" tab, the user has a multitude of adjustments for the weapon systems of the agent. The first observation is in the upper left corner for the status of weapons is located. At the start only the number one is in green. As the user creates more weapons the corresponding status of weapons number will turn green. The weapons can be labeled, however, that does not affect the scenario and is only for the benefit of the user. The current weapon is indicated just under the status of weapon designator. Each weapon can be designated as a primary or alternate weapon with only one being able to be designated as the primary weapon. The user can use the simple or the advanced adjustments for each weapon. Primarily, in the simple tab the user will specify the type of weapon as well as the hit probabilities at specified distances for the weapon. Figure 11 depicts the simple weapon settings.

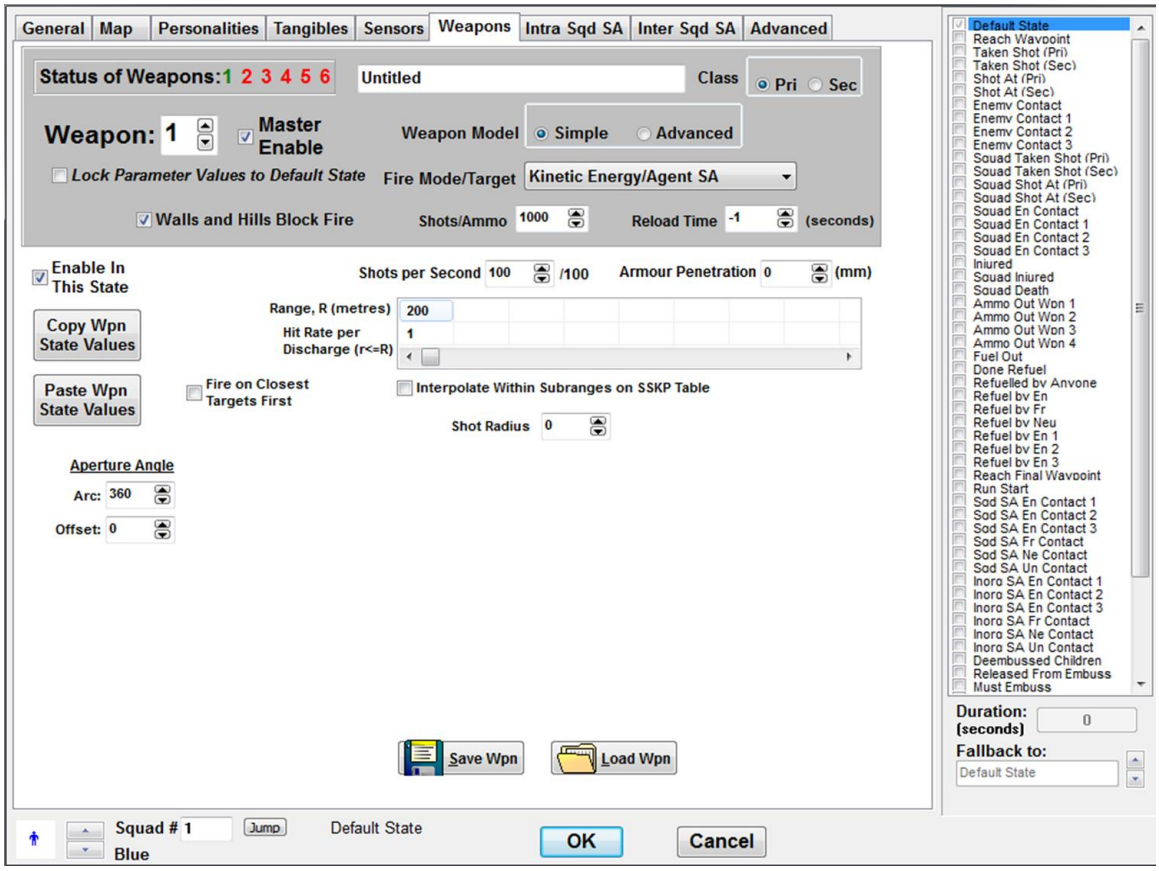

Figure 10. Weapons Tab under "Edit Squad Properties," from [21].

When more specificity is called for in the scenario the user can select the advanced settings for greater control in how the weapon performs. Figure 11 depicts the advanced weapon options.

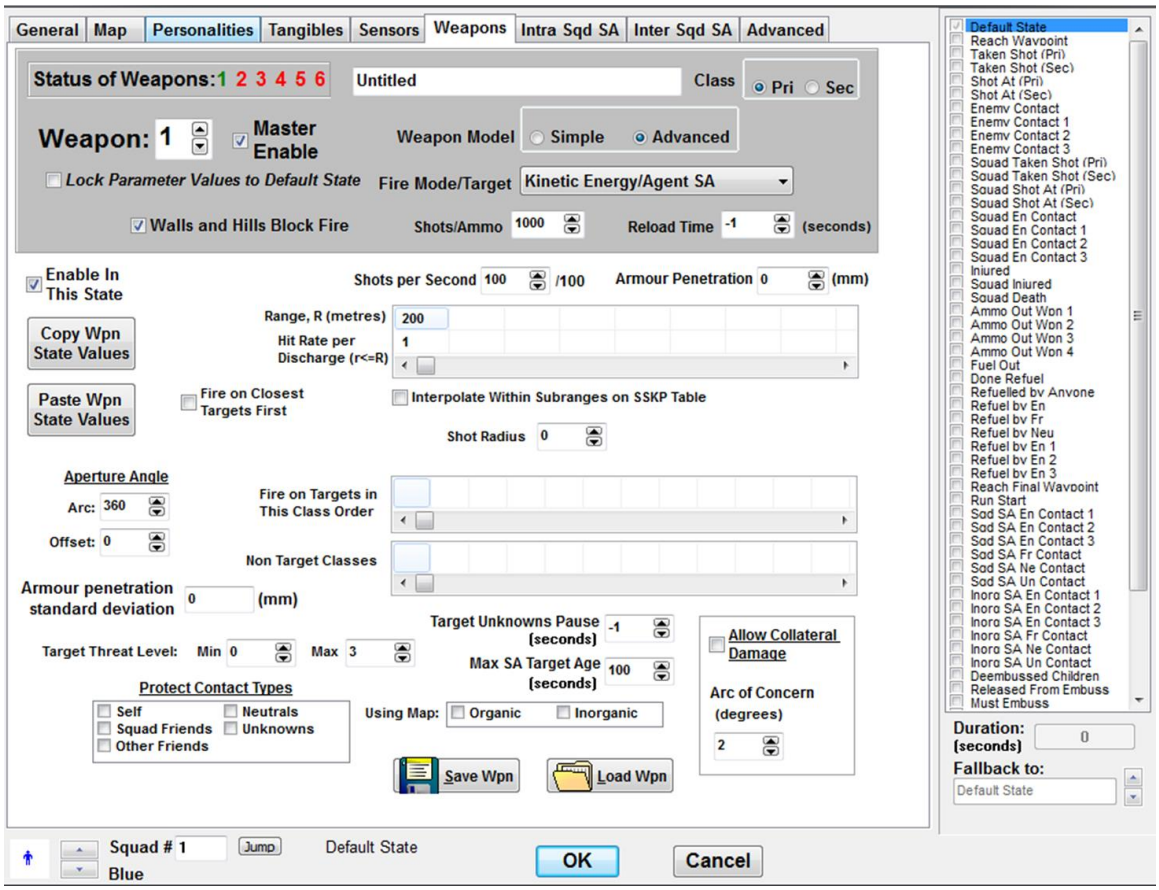

Figure 11. Advanced Options for Weapons Tab, from [21].

The differences between the options are quite clear once the user toggles between the two. The most obvious is that now a user can establish a priority of targets per weapon system as well as the armor penetration. The user can also establish the priority of targets the weapon will and will not be used against. This prevents behavior such as firing the main gun of the tank against dismounted infantry. Figure 12 is an example set up for a tank main gun.

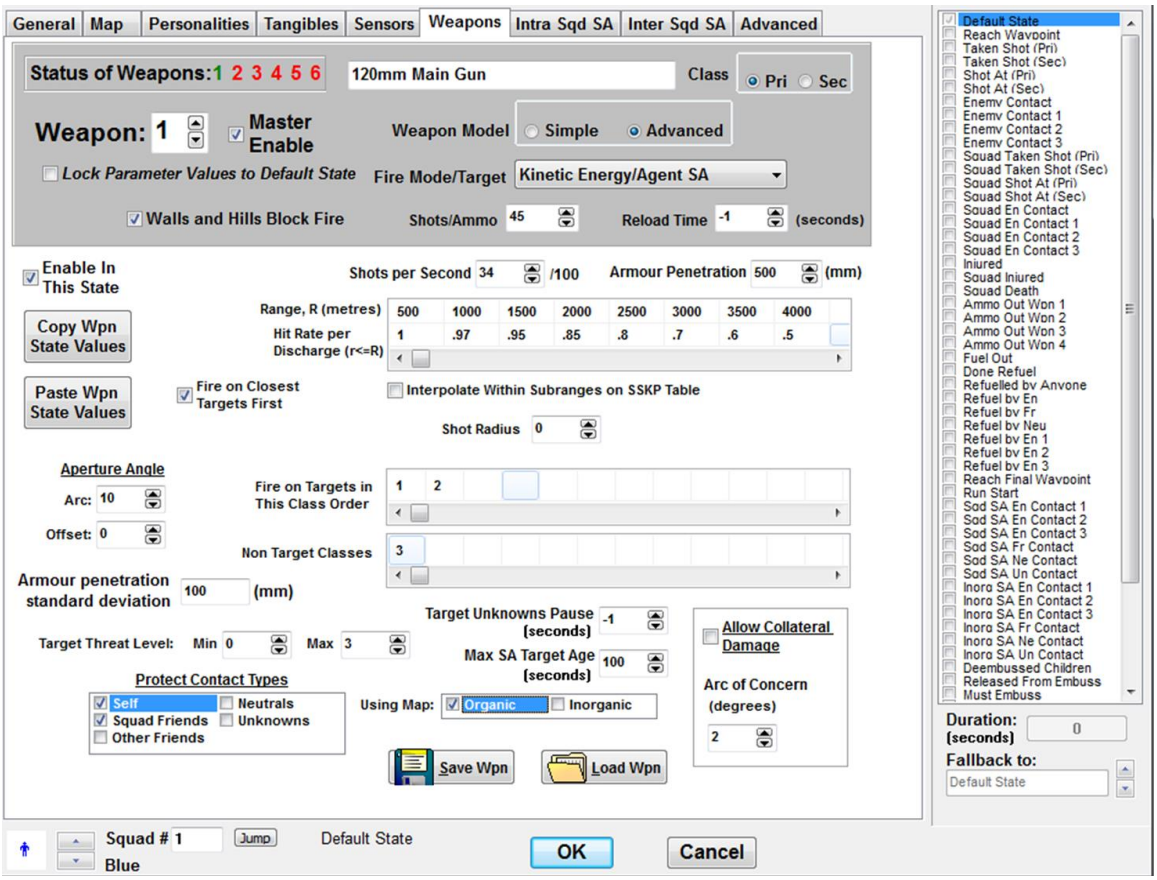

Figure 12. Weapon Tab Example, from [21].

Figure 12 is an example set up for the main gun of an M1 Abrams Tank. The values inputted into the tables are not exact and a user should conduct research into which hit probabilities and specifications are appropriate to meet the intent of the specific scenario. The setting for the main gun should be the kinetic energy/ Agent SA which indicates that the tank develops its own understanding of the enemy and is firing a kinetic energy weapon. In contrast, an artillery piece would need the reporting of enemies by other agents since it is an indirect fire weapon and would be set as a High Explosive/Inorg. SA. A mortar system that is a part of an infantry company could be modeled as the Explosive/SQD SA as they would be close enough to the enemy to develop their own situational understanding of the battlefield.

The ammo setting is for 45 rounds, which is the basic load for the tank. Setting the value -1 for the reload time prevents the tank from being reloaded during the scenario

and MANA treating the tank like it was a riffle being carried by an infantryman where they could swap magazines once the ammo was out. Setting the shots per second to 34/100 means that the tank is only able to shoot about every three seconds. In actual combat, it is likely that a tank would engage the closest enemy first as that would be assumed to be the greatest threat. MANA allows this priority to be established within the scenario by clicking the appropriate box. One assumption made by MANA is that the armor penetration is standard for all distances of engagement. While the M829A3 SABOT round may be able to penetrate over 600mm of armor at 1000 meters, it cannot do it at 4000 meters. This assumption can be mitigated somewhat by adjusting the armor penetration standard deviation which puts a normal distribution over the penetration with a standard deviation of a specified amount. Adjusting these values to fit the scenario is important for a user that seeks to model the right behavior.

In this example, the target priorities have been set for classes 1 and 2, where they are other tanks and armored personnel carriers. Agent class 3 is prevented from being shot at as that class has been specified as being dismounted infantry. A user could also specify airborne agents such as planes and helicopters as classes that the tank should be prevented from engaging. The setting for maximum and minimum threat levels are important for a user to consider. This prevents a weapon system from engaging too few enemies or too many enemies at a time. In this case, it would not be logical for a tank to seek to engage a cluster of enemies where he was outnumbered by a significant amount. Also, a user may not want to employ a limited number of artillery rounds against individual infantry soldiers. Setting the minimum threat level to more than 1 or 2 would force the agent to fire at groups of enemies rather than individuals. In this case, the tank would engage a single opposing tank or infantry fighting vehicle, but would avoid groups of more than 3 clustered together. It is important for the user to ensure that there are not conflicting characteristics specified.

The other weapon systems on the tank can be established in the same way, with one exception. When a user establishes a shot radius for kinetic weapons this is to simulate the dispersion associated with machine gun fire. This would be set much lower for the coaxially mounted M240B than the M2 fired by the tank commander.

The method for determining if an agent is hit by a direct fire weapon is relatively straight forward. Once and agent shoots at an opposing agent a random number between 0 and 1 will be called. If that number is less than the probability of hit specified at that distance, then the agent is "shot." If the random number is greater than the probability, then the agent has missed.

C. SHOOT, INDIRECT FIRE

Indirect fire is similar to direct fire except that the probabilities for being hit are calculated from the point of impact of the shot rather than from the shooter [3]. When observing from the point of impact the process for scoring a hit or miss is the same as with the direct fire if an agent is within the specified blast radius of the round. The high explosive rounds will also have a specified armor penetration value so agents within the hit radius that have armor ratings that exceed the penetration value would not be affected by the shot, even if they are hit [3].

MANA makes a simplification in the modeling of indirect fire by not putting a probability distribution for the accuracy of the high explosive weapons. The high explosive weapon always hits in the target location and the probability of hit is then calculated from the point of impact to the agents within the hit radius [3, 19]. While this assumption may decrease the time necessary to evaluate engagements, it is not very close to reality. Artillery is not a pinpoint weapon system and the distributions of impact areas are well known and could easily researched by an individual utilizing MANA to model a combat scenario. It seems that MANA could easily incorporate a bivariate normal distribution for the impact area by having the user input the desired standard deviations in the set up menu.

D. HIT/KILL

MANA is not a physics-based simulation and attempts to simplify some of the interactions within the scenarios [3]. One simplification is in the calculations of how agents are "killed" within the model. In other models there is a Bayesian calculation for the probability that an agent is killed given that it is hit. However, in MANA each agent is designated to poses a certain number of hits before it is "killed." Thus, if an agent is

hit then it is considered “wounded [3].” This status remains until the agent is “killed.” Though this prevents some uncertainty within the model, it may not impact the distribution of results over many iterations of the scenario. If a weapon is not specified to have the ability to penetrate an agent’s armor, the agent is not wounded when hit by that weapon [3].

THIS PAGE INTENTIONALLY LEFT BLANK

V. COMMUNICATE

In this section, how the agents within MANA transfer their individual understanding of the battlefield amongst their forces to build a collective understanding of the enemy is shown.

The ability to communicate is vital as a force coordinates maneuver and intelligence across the battlefield. This is especially so as the U.S. military is becoming increasingly reliant on technology to communicate orders and intelligence between maneuvering entities. Conversely, one primary objective of battle is to degrade the ability of the enemy to communicate and coordinate maneuver effectively. Rarely is it that a force has perfect communication and a perfect understanding of the battlefield environment. The biggest assumption that a user can make in a combat model is that each force has perfect communication where exact intelligence is instantaneously relayed between agents allowing them to maneuver with perfect understanding of the battlefield. Of course, this situation never occurs in real combat situations and it is important that a combat model have the ability to simulate varying levels of communication proficiency. Understanding how increased communication affects the battle is essential, as is the understanding of how losing the ability to communicate affects units. MANA possesses a robust ability to model communications on the battlefield.

Communication in MANA is primarily observed by the user through the situational awareness map. This map represents the spatial understanding of the battlefield for agents within squads and between different squads. The situational awareness map allows a user to see what information is being passed within the force. Once an agent identifies and classifies an opposing agent it relays that information within its own squad and then to the other squads within its force. With perfect communication and no delay the information is instantly shared and all agents know where and what is maneuvering as soon as it is identified. This is not a common occurrence in actual combat so a user would desire some level of delay in the communication as well as some probability of incorrect information to be allowed to more closely replicate the “fog of

war." The importance of correctly quantifying the communication attributes is clear for the model to accurately represent real world situations.

In MANA a controller can specify the following communication characteristics [3]:

- Latency: delays in communication
- Reliability: likelihood information sent is correct
- Capacity: how much information can be sent per time step
- Filtering: which information is sent to different squads

Within squads a user determines the communication delay, contact persistence as well as conflict resolution with repeated reports and large numbers of unknown agents of the battlefield. The inorganic situational awareness tab in the edit squad properties is how squads report information between each other. There is greater flexibility for a user to manage different levels of competency of communication flow between different squads. The MANA 4 manual provides detailed descriptions of each attribute of the communication between squads. The stochastic nature of communication is observed in the reliability and accuracy settings of inter squad communications. These two settings are established as percentages where at each time step a random number is drawn against them [3]. If the random number is less than the established percentage, then the message is sent successfully and with the correct information. If the random number is greater than the established percentage, then either the message is not sent or it is sent with incorrect information. Of course, a message not sent with incorrect information is also possible, but it is indistinguishable from the message not being sent with correct information. A controller can also determine if a message is stored if an agent is out of communication range with the intended recipient [3].

VI. CONCLUSION

MANA is a collection of simple ideas and concepts that when put together form a powerful and important tool for military operations research. All models, including MANA, have limitations and unless the scenario is properly established can provide meaningless results. MANA as a combat model will not provide the "answers to the test", in that it will not exactly predict the future. However, when properly established it can reveal great insight into the trends that are displayed over multiple iterations as the user manipulates the subject parameters of the research within in the model. Understanding why changing one parameter affects the outcome is important for a user when interpreting the results of the scenario. The underlying mathematics within the model are the "engine" that drives the results. Though relatively simple, when a user does not understand how the model achieves its results the user cannot understand the results. The aim is that this thesis has added to the established user manual and provided users additional insight into how MANA operates and improves the understanding of results.

While the way that MANA generates random numbers has benefits in terms of speed, there exists correlation in the sequences of numbers that are generated. It should be investigated whether this correlation negatively affects the model and if a more "random" sequence needs to be generated. The comparison needs to also weigh any changes to the generating method in terms of additional computing power required. Making the generation of random numbers more complicated will increase the time needed to generate them. Would this additional time cause the model to run too slowly and degrade the usefulness of the model?

MANA has been designed to have a simple interface with which a user can intuitively set up and execute a scenario. While this is true, there are still hundreds of settings that have to be made, each of which can prevent the scenario from providing meaningful results if not quantified properly. In many cases, the settings can conflict with each other and prevent the model from running. Thus, there exists a learning curve for a user in establishing the proper settings for a scenario to reach meaningful results. A user needs to have an understanding of how each parameter affects the movement,

perception, and communication of each agent and squad within the scenario to properly establish the parameters. The tutorial in Appendix A should provide a baseline example of how a new user establishes a scenario and allows the user to more rapidly navigate the intricacies of MANA.

APPENDIX. MANA TUTORIAL

A. INTRODUCTION

MANA is an agent-based model where the agents within the scenario represent individual pieces of equipment or personnel on the battlefield. The agents react to the terrain and enemy without direction from the user where the resulting behavior is a phenomenon known as "emergent behavior." MANA is an important and powerful tool to simulate combat to gaining an understanding of the effects of influencers on combat scenarios. Many of the physics-based attributes associated with other combat models have been removed in MANA which allows users to focus research onto the behavior of the agents within the simulation. This also simplifies the computation behind the model which allows a user to conduct more iterations of a simulation in the same amount of time. Many of these physics-based elements are difficult to quantify and model correctly and only add to the computational complexity for the computer running the model. MANA has limitations (as do all models), however, the ways in which it can be utilized for research is vital to the military community.

Despite the effort of MANA to create a user friendly interface, the model can be overwhelming for a new user to get started. The interface is designed to be quick and easy to learn (and for the most part it is); however, there are hundreds of adjustments that a user can make which directly affect the ability and movement of the agents within the scenario. If the settings are not established correctly, the scenario may perform strangely with results such as fratricide or enemies passing each other and not engaging in combat. This leads to a steep learning curve for the user when starting to build scenarios within the model. This tutorial utilizes a simplified mission to conduct a walkthrough of the steps to build the environment and squads so a user can evaluate the end results of the scenario. In the end, the reader should have a basic understanding of how to load a background image, build terrain and elevation representation maps, establish and define weapon, communication, and movement characteristics, and run multiple iterations extracting data from which to conduct analysis.

This tutorial will be organized in a logical manner where a user is instructed in starting a new simulation, loading the environment, establishing squads of agents, defining weapons, defining communication abilities, executing multiple iterations and exporting the data.

B. GETTING STARTED

When a user initially opens the program MANA the first screen is the user interface with a blank "game board". The user clicks on the file menu button on the top of the screen and select new to begin establishing a new scenario. Selecting "new" from the "file" menu allows the user to define some of the basic characteristics of the environment. This is depicted in Figure 13.

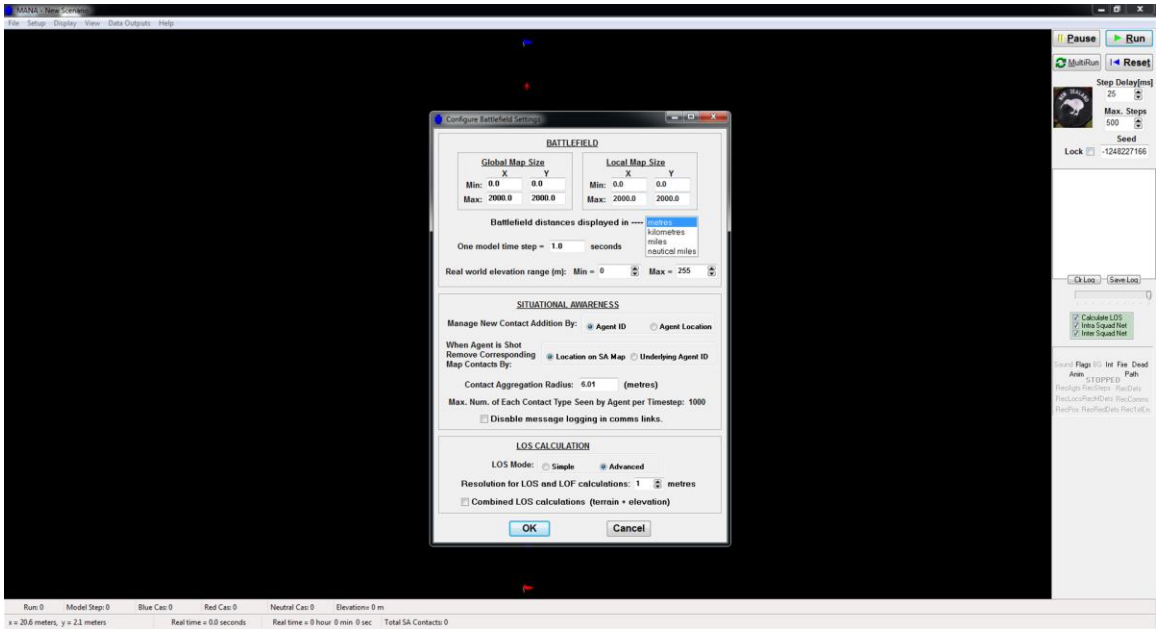

Figure 13. Starting Menu, from [21].

The user defines the dimensions of the battlefield, scale of the time steps, basic situational awareness characteristics and how line of sight is calculated.

On the right hand side of the user interface are buttons for starting the scenario, stopping the scenario and resetting the scenario. A user can also click the multi run button to execute the scenario multiple times without having to manually resetting and

starting the scenario each time. The user can also specify the step delay and the maximum number of time steps that the model will execute. The seed value is displayed where the user can enter in a number manually or have it randomly generated. This is important if the user wants to replay the scenario with the exact same results to gain further insight into the results of a specific iteration of the scenario. On the bottom of the interface MANA displays which iteration of the scenario is being executed, the current time step, the number of casualties for red, blue, and neutral, the elevation where the cursor is currently located, local grid location, contacts and real time equivalent.

Once the basic environmental elements are defined, the user selects the set up menu button on the top of the screen, shown in Figure 14, to begin defining agent characteristics and building the environment.

Figure 14. MANA Setup Menu, from [21].

The set up menu allows the user to adjust specific scenario elements of the agents and squads as well as load background images and develop terrain and elevation maps, as displayed in Figure 15. When the user makes the selection for editing squad properties, a window with several tabs is opened allowing the user to specify all the elements of the agents and squads to include weapon characteristics, communication capabilities, number of hits to kill, starting position and orientation, route way points, and movement formations.

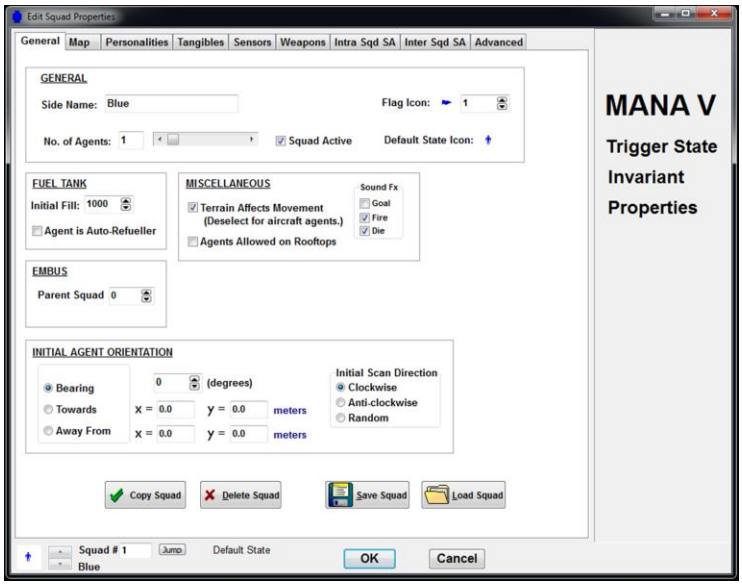

Figure 15. MANA Edit Squad Properties Menu, from [21].

The final initial setting that the user can establish is the stop conditions. This is selected through the setup tab at the top of the user interface, shown in Figure 16. Once into the selection window, the user defines the stop conditions of the scenario.

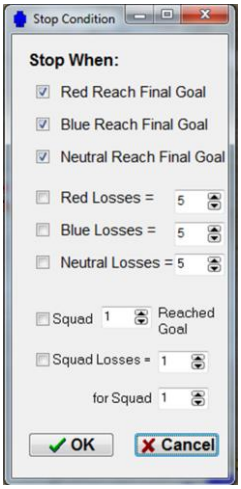

Figure 16. Stop Conditions, from [21].

C. THE TUTORIAL SCENARIO

The situation is that a Marine Expeditionary force is establishing a beach head in the vicinity of Badar Abbas, Iran. The Iranian forces are poised to attack south and

disrupt, or defeat the Marines before they can mass their forces and attack north. A battalion from the 82nd Airborne Division has established a blocking position north of the beach head with the task of conducting a guard for the Marine forces along the beach and preventing the Iranian attack south from affecting the beach landing. [22]

In this tutorial, a mechanized company of Iranian forces against a platoon (+) of light infantry from the 82nd that have additional anti-tank capabilities and artillery support is modeled. The Iranian mechanized company consists of four T-72 tanks and six BMP armored personnel carriers. They are supported by two 105mm towed howitzers. They will not employ dismounted infantry. The 82nd platoon consists of four squads of ten soldiers each with two, single shot anti-tank weapons per squad and machine guns. They are supported by two towed 105mm howitzers. This is not necessarily how either force would doctrinally organize or equip, but should demonstrate how to implement the characteristics into the model and run the scenario.

The 82nd platoon establishes defensive positions along the main road oriented north and south. From these positions they employ their anti-tank weapons against the tracked and wheeled Iranian vehicles as well as calling for indirect fire support from their supporting artillery section. The Iranian forces seek to maneuver against the 82nd fixing them in their positions and defeating them with overwhelming firepower. They suppress their positions with supporting indirect fire and attempt to move past the guard to engage the marines on the beach to the south. Can the 82nd platoon attrite the Iranian forces enough to prevent them from advancing south to attack the Marines massing on the beach?

1. Establishing the Battlefield

The first step in developing the scenario is to build the environment. Utilizing Google Earth is utilized for the imagery displayed as the background image. While the terrain and elevation maps can be built within MANA, the best course is to build the maps utilizing Microsoft Paint.

In this scenario the battlefield will be 15 km x 10 km, displayed in Figure 17. The scenario sets the stop conditions of the battle as 10 red force causalities and 40 blue force causalities.

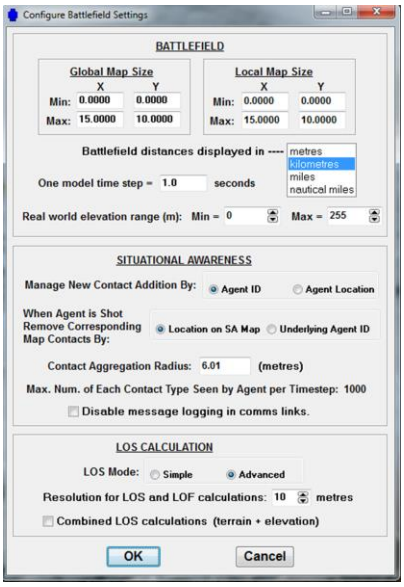

Figure 17. Tutorial Set up Settings, from [21].

Building the battlefield is one of the most important aspects of the development of the scenario. It is also one of the most difficult, and time consuming. Building an accurate depiction of the terrain is vital to get an accurate representation of movements of the agents within the scenario. This step alone could take weeks if the battlefield is complex with varied vegetation and large elevation changes.

The user selects the area that the simulated battle takes place. It is important to note the dimensions of the battlefield and align the area of the image with the setting of MANA. Figure 18 is the battlefield that serves as the background for the scenario.

Figure 18. Tutorial Imagery, from [23].

After the background image is loaded, the elevation map is created. Elevation in MANA is represented by a grey scale with white being the highest elevation and black being the lowest or no elevation. There are 256 different levels of grey to depict changes in elevation. This process can be long and tedious, but is important for the model. Figure 19 is a simple elevation representation in grey scale. Utilizing Microsoft Paint to build the map, a user should note that the image needs to be saved as a bitmap to be loaded properly by MANA.

Figure 19. Tutorial Elevation Map, from [24].

The next piece to build is the terrain map. The area where the battle is going to take place is mountainous and there is little vegetation. Though the names of the default terrain features include dense and light bush, this scenario can utilize these features since the only function of the terrain features is to provide modifiers for movement, cover, and concealment for the agents. If the scenario necessitates changes to the default values for the terrain or to create new terrain features, the process is straight forward. In this case we will utilize the existing terrain features, putting dense and light bush modifiers over the mountainous regions. Figure 20 is the terrain modification window within MANA. Microsoft Paint has the ability to create custom colors to use. Inputting the exact color specifications from MANA into Paint will allow the user to create the terrain map such that MANA understands what the colors indicate as terrain characteristics.

To edit terrain follow these steps

- Click the setup button then select scenario map editor.
- Specify the size of the map needed.
- Select edit terrains of the right hand side of the user interface.

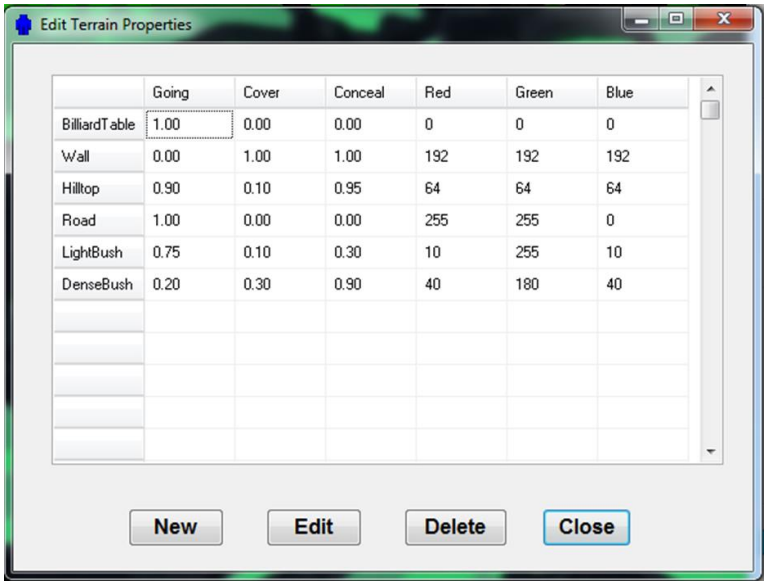

Figure 20. Scenario Map Editor Window, from [21].

The edit terrain properties window allows the user to view the values for going, cover and concealment, as well as the red, green, blue values to create the color. Double clicking on the values within the table allows the user to modify the characteristics as necessary. Utilizing Microsoft Paint, the terrain map for the scenario is created and depicted in Figure 21. In the Paint program, the background image can be loaded and the user can "paint" the terrain on top of the image. This helps to ensure that the terrain aligns correctly with the image.

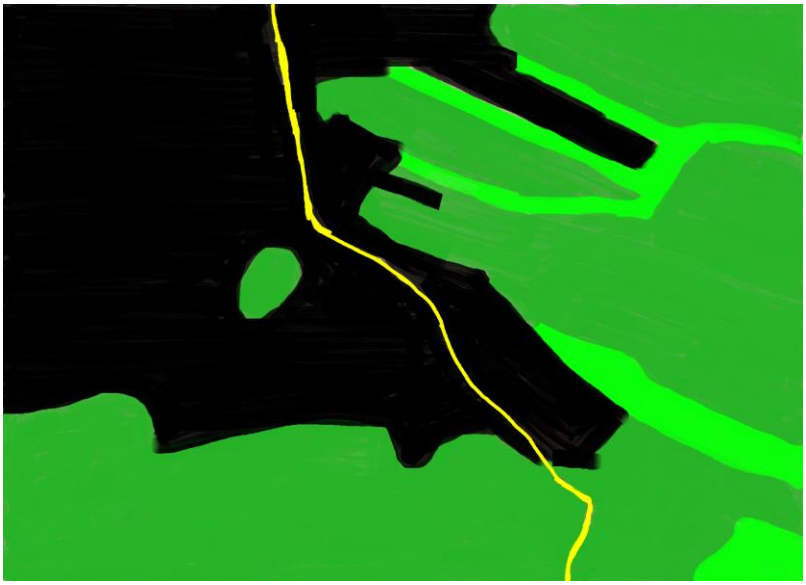

Figure 21. Tutorial Terrain Map, from [24].

Once the terrain and elevation maps are created, they can be loaded into MANA by utilizing the load terrain map and load elevation data selections under the setup button at the top of the user interface. To ensure that the scale is correct, each of the terrain and elevation maps can be loaded as the background image to ensure that they align correctly with the background. The user should move the cursor across the battlefield, observing the elevation reading in bottom of the user interface to ensure that the elevation coincides with the image in the background.

2. Creating the Blue Force Infantry Squad

Now that the battlefield is established with the terrain and elevation data loaded into the scenario we can begin to build the forces and place them on the map. The platoon of infantry from the 82nd is created first.

a. Blue Force Infantry Edit Squad Properties

Open "edit squad properties" from the setup menu and begin to establish the characteristics of the 82nd platoon. The size of the platoon is 40 soldiers broken down into four squads of ten soldiers each. The other selections in the general set up window maintain the default settings. Once the first squad is created, the copy squad button can

be used to create the other three squads. The easiest way to do this is to finish all settings for the first squad then copy the squad so the other three have the same characteristics. The general settings are depicted in Figure 22.

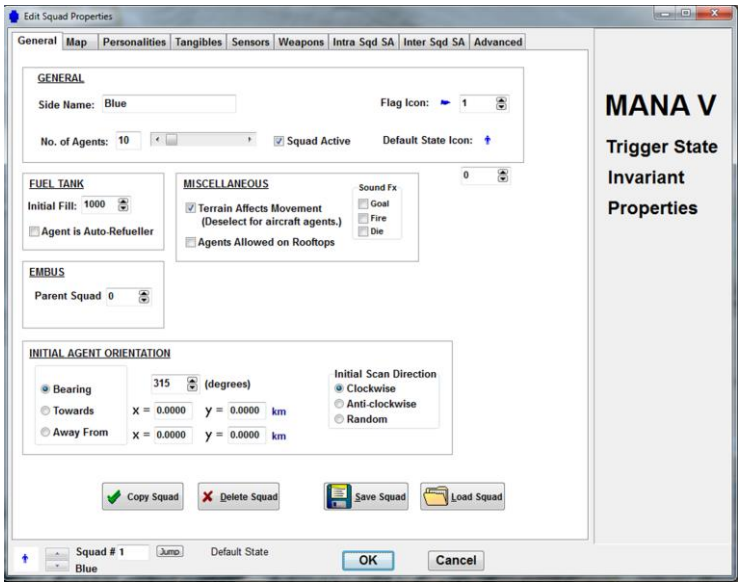

Figure 22. Blue Force Squad General Setup, from [21].

b. Blue Force Infantry Map Placement

The next step is to place the 82nd platoon on the battlefield and establish their waypoints and "goal" destination, as depicted in Figure 23. Since the scenario depicts the platoon in defensive positions, it does not make sense that they would maneuver towards a "goal", so no waypoints are established beyond the "goal'. Click the clear waypoints button on the right hand side of the user interface, and then right click on the map where the desired starting point for the squad is to be located. Next, left click on a point where the "goal" destination should be located. The first left click is designated at the endpoint and each subsequent waypoint is previous to the first established waypoint. In effect the placement of the waypoints is done "in the rears". This must be completed separately for each of the four squads if they are not to start in the exact same location.

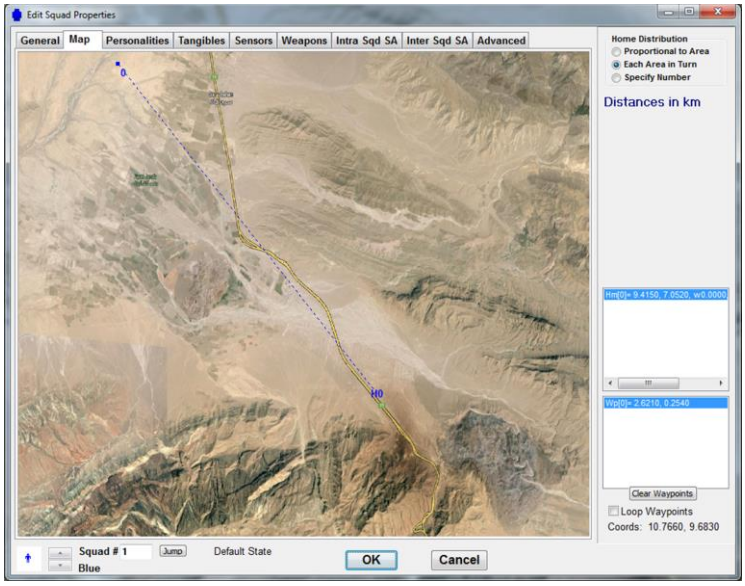

Figure 23. Blue Force Map Set up, from [21].

c. Blue Force Infantry Personality Weightings

The next step is to establish the personality weightings for the squad, depicted in Figure 24. The weightings are broken down into three sections, Agent SA (Situational Awareness), Squad SA, and Inorganic SA. The agent SA is how the individual agents move against what they, individually, senses on the battlefield. The squad SA is how the agents move against what the squad senses and is communicated to the individual agents. The inorganic SA is how the individual agents maneuver against what is perceived and communicated between different squads. Since the platoon is in a defensive position, the settings for movement "towards enemies," "waypoints" and "terrain types" are set to zero. This prevents the squads from moving out of the defensive positions. If a user desires an altered response for a specific event, such as reacting to contact or when a friendly agent is killed, then the corresponding trigger state is checked and the user can specify different personality weightings to take effect once that criterion has been met. In this case no additional trigger states are created.

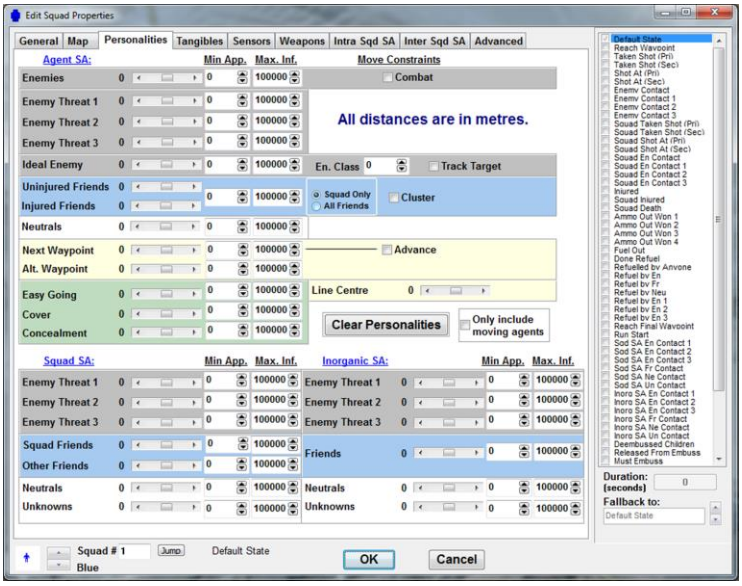

Figure 24. Blue Force Personality Settings, from [21].

d. Blue Force Infantry Tangibles

The next step defines the characteristics under the tangibles tab, depicted in Figure 25. Under this tab, maintain the default icon number and allegiance setting as being oriented to the blue force. Agent threat and agent class define how the enemy agents "view" this agent and are used to determine how they are maneuvered against and what weapons the enemy uses against them. In this case, both are specified as one for the infantry squads. The sensor height is set for 2 meters, which is about the height of a soldier, and the weight remains with the default setting of 0.1. This means that the agents will accelerate rapidly. The only reason to change this would be if the scenario was modeling large entities, like naval ships in the ocean, as it takes time for them to turn and get up to full speed. For this scenario, the assumption is that all entities will be able to reach full speed instantaneously. The speed will be set at 10 KPH. The 82nd do not require fuel so that section will retain the default settings of zero. In the self-protection section the number of hits until kill is set at 5, concealment level at 33%, and armor thickness at 5mm. These values are somewhat arbitrary for this tutorial, but the accurate values would be important for accurate research. Though the 82nd will not maneuver towards waypoints, setting the waypoint radius values at a value greater than one allows the agents to reach a waypoint without having to be right on top of it. The boxes under

the embossing behavior will be unchecked as there are not forces riding within other agents, such as mechanized infantry dis-embarking from the back of a Bradley Fighting Vehicle. Within the angular movement section of the menu the default status are utilized, except for the "in absence of enemy" section where the button for "scan back and forth" will be checked and the arc of scanning will be set at 180 degrees. This makes the agents scan west to north to east every second.

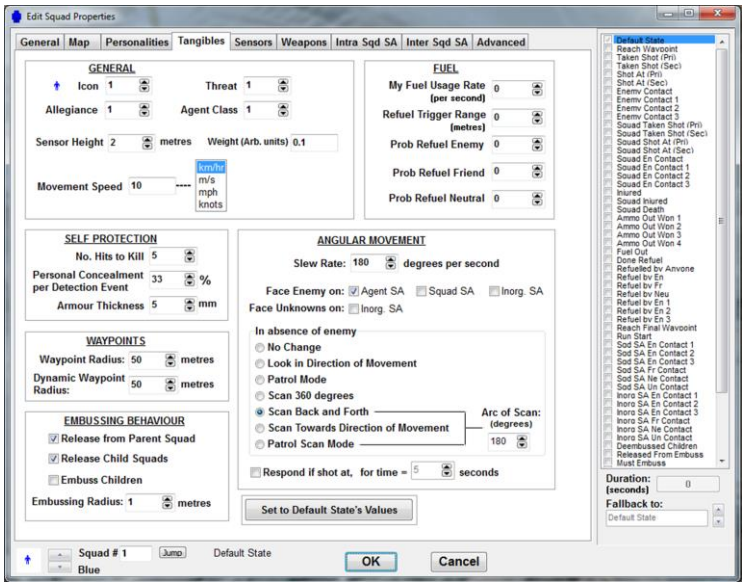

Figure 25. Blue Force Tangibles Settings, from [21].

e. Blue Force Infantry Sensor Settings

The next setting is the sensor setting, depicted in Figure 26. The blue forces are outfitted with only one sensor (presumably their eyes and organic optics). The advanced settings are utilized and the sensor is set as a class "A" sensor. The ranges of detecting and classifying agents are established. The "target specific classes" button is unchecked so the platoon attempts to identify all classes of enemy.

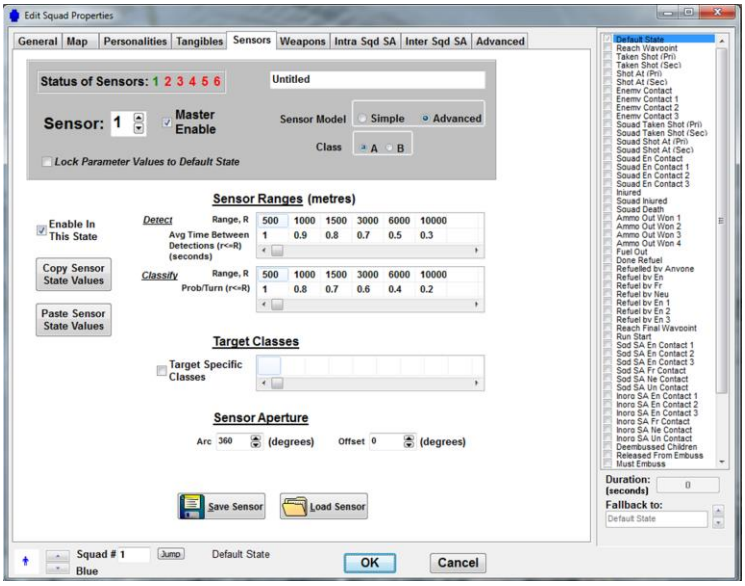

Figure 26. Blue Force Sensor Settings, from [21].

f. Blue Force Infantry Weapon Settings

After the sensors are established the weapons need to be defined for the platoon. The squads have two weapons assigned, a machine gun, and an anti-tank weapon. The anti-tank weapon is established as the primary weapon and the machine gun is assigned as a secondary weapon. All weapons fire on the closest targets first and be established as kinetic weapons with squad SA. The anti-tank weapon is set for two rounds with no re-load capability. The "shots per second" setting is set at 200. This indicates that this weapon can shoot two targets per second. The penetration is set at 200mm, which allows the weapon to defeat the tank; however, the standard deviation for the penetration is 25mm which indicates that the weapon might hit the tank and not damage it. While the weapon can hit a target out to 4000m, the probability of hitting is only 0.3. The primary targets are the T-72s, but this weapon is also used against the BMPs as second in the priority of firing. Placing a 1 in the non-target class indicates that the anti-tank weapon will not be used against dismounted infantry. All other default settings are maintained. The second weapon, the machine gun, is established as a secondary weapon. In this case the machine gun can fire at two targets per second and has an armor penetration of 50mm. This allows the weapon to be effective against the BMPs, but the standard deviation indicates that the weapon may hit the target but not damage it on occasion. The

machine gun is not used against the tanks. It is important to note that the box "enable in this state" on the left hand side of the window above the copy weapon values is checked, or the weapon will not be used. It is also important to note the values for the "target threat level". In this case, the setting determines what concentration of forces the weapon will or will not be employed against. If the threat level of the tank is set to 3 and there are 4 tanks in close proximity; then that threat level is 12. If the max setting is less than 12, then the squad will not engage those targets with that weapon. If the minimum stetting is greater than 12 the weapon will not be employed against the group. This would make sense in reference to artillery where a unit would not want to waste rounds on individual targets, opting instead for larger groups clustered together. All other default settings are maintained. Figures 27 and 28 depict the weapon settings for the blue forces [25].

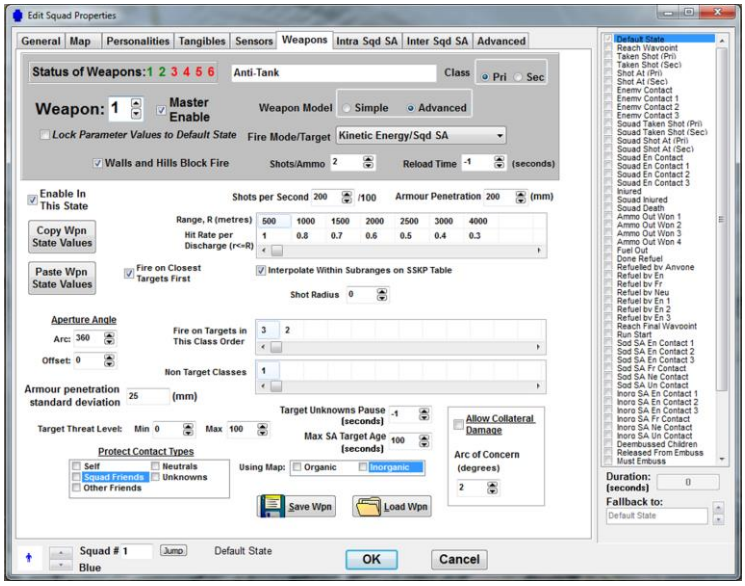

Figure 27. Blue Force Weapon 1 Settings, from [21].

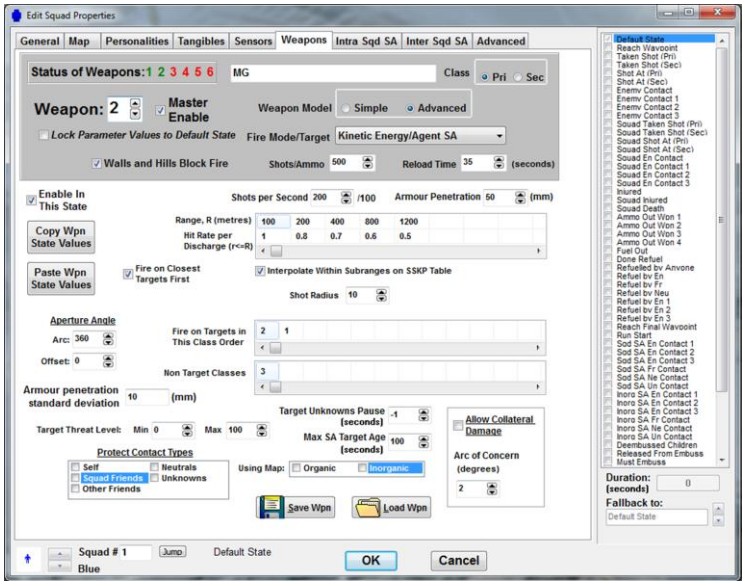

Figure 28. Blue Force Weapon 2 Settings, from [21].

g. Blue Force Infantry Squad Communications

Communication in MANA is simulated through the "intra" and "inter" squad situational awareness (SA) tabs. The first communication section is the Intra Squad tab which controls how the agents communicate within their squad. This essentially depicts how quickly identified and classified agents are added to the squad situational map and how long agents remain aware of contacts after they can no longer "observe" them. The communication delay value is set for 5 seconds which can be thought of as the time it takes to submit a contact report. All other defaults within this tab are maintained. The second communication section is the Inter Squad tab. This tab manages how the squads communicate between each other as well as with the supporting artillery. The default settings for the inbound organic settings section are maintained. In the outbound communication link a new "channel" is created between the infantry squad and the artillery. The user has to create a default artillery squad to build this link. The characteristics for the artillery squad are established later. Each link is effective to 10000 meters, and possesses the ability to send 1 message per time step. There is a 15-second delay sending messages to the artillery section. This simulates the time needed to perform a "call for fire", or send a contact report. The reliability of the messages is set to 90% between infantry squads and 80% to the artillery section. The priority is high for all

reports to the artillery. The intra- and inter-squad situational awareness settings are depicted in Figure 29 and 30. The communication link setting is displayed in figure 31.

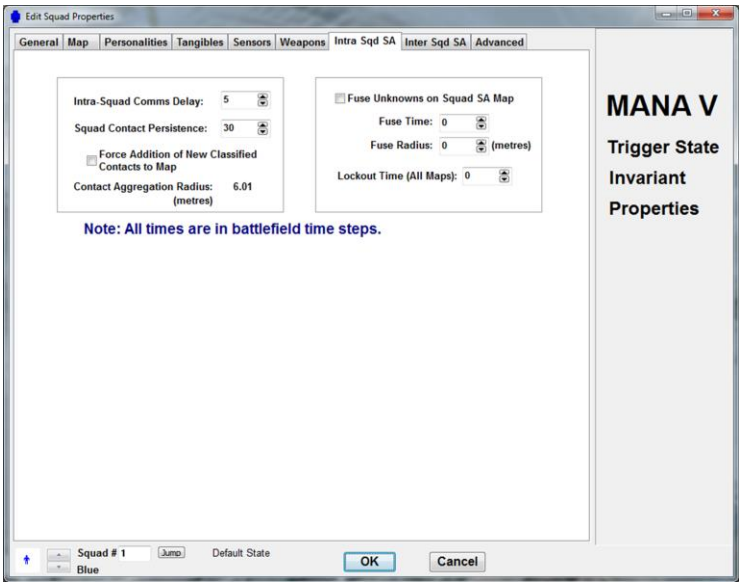

Figure 29. Blue Force Intra-Squad Communication, from [21].

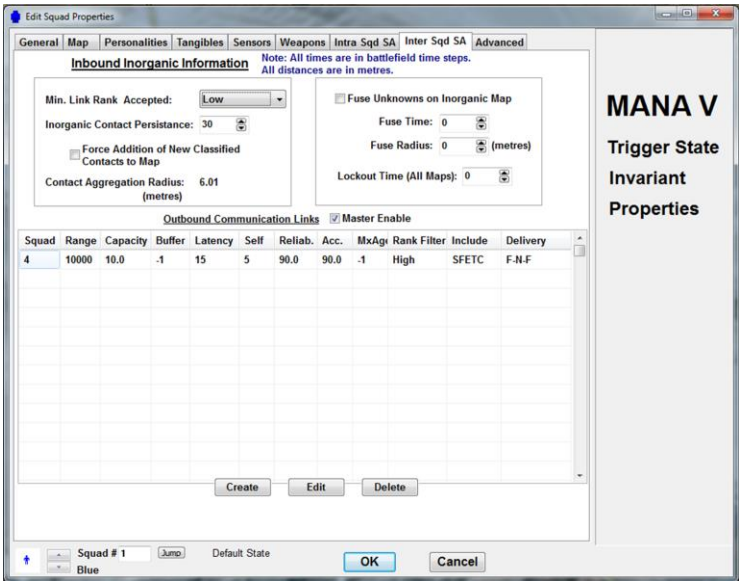

Figure 30. Blue Force Inter-Squad Communications, from [21].

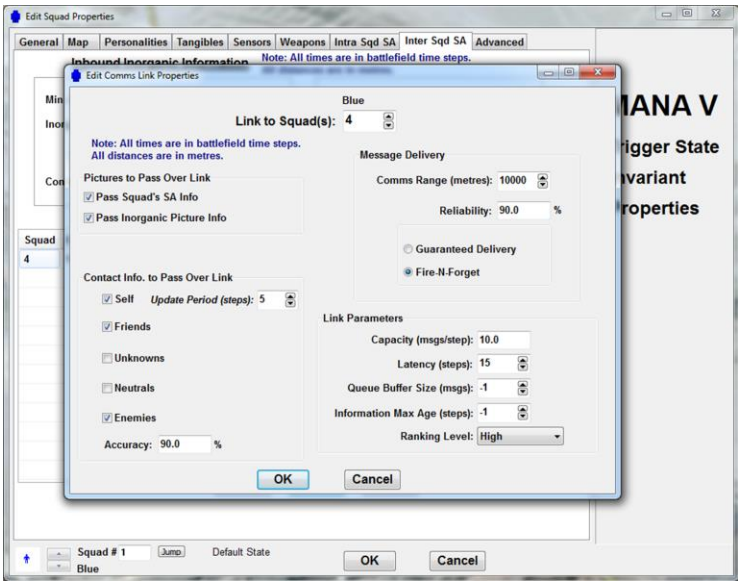

Figure 31. Blue Force Communication Link Setup, from [21].

h. Blue Force Infantry Advanced Settings

The advanced settings tab, depicted in figure 32, allows the user to specify some tactical attributes of the squads. This includes movement formations, inter-agent separation, prioritizing attacking enemies when in a flanking position, and how far away from a designated path among other features that could be important depending on the specific scenario being modeled. In this tutorial the squads assume the default statuses for most options. The infantry squads take the formation of a line and have a slight positive desire to maneuver into that formation. The artillery section does not make any adjustments in the advanced setting tab.

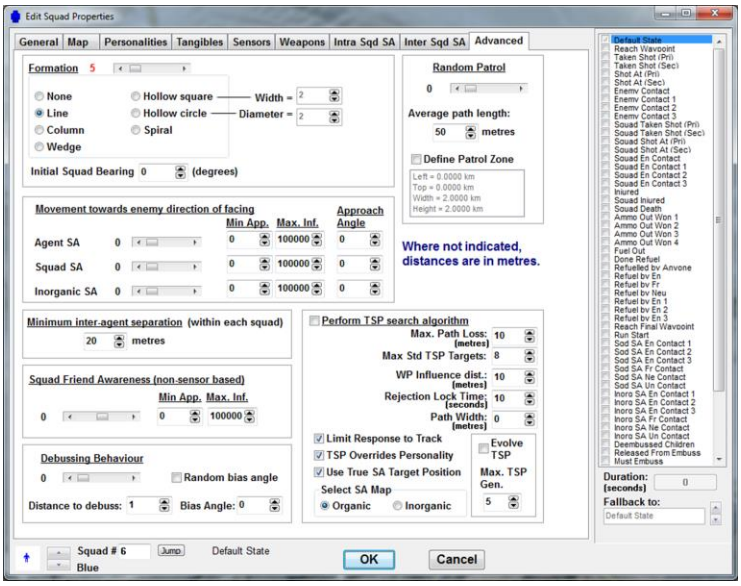

Figure 32. Blue Force Advanced settings, from [21].

3. Blue Force Artillery Settings

The following figures, 33-37 display the settings for the Blue Force artillery.

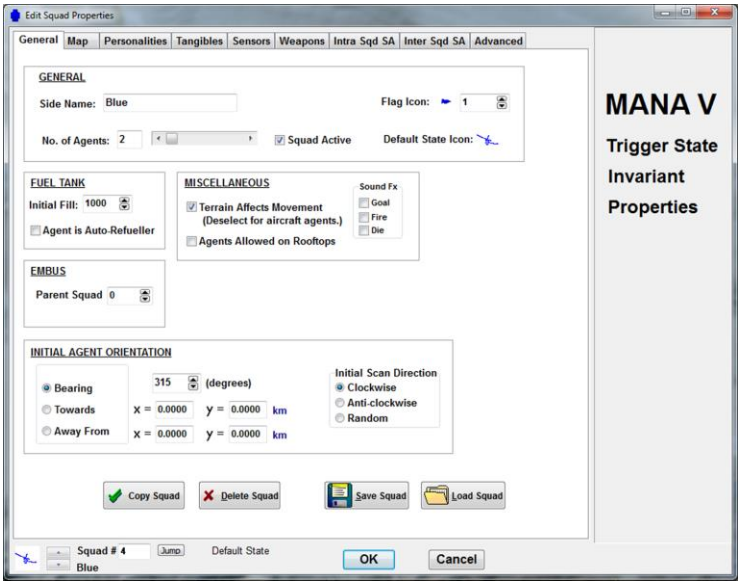

Figure 33. Blue Artillery General Settings, from [21].

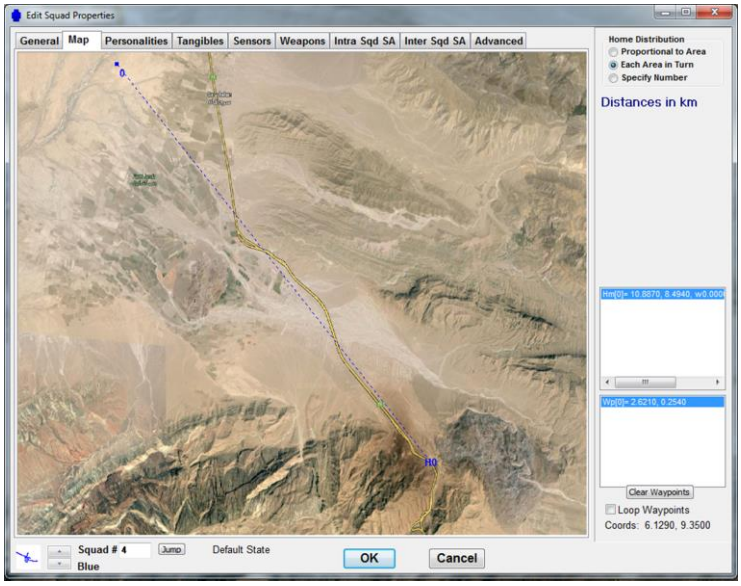

Figure 34. Blue Artillery Map Placement, from [21].

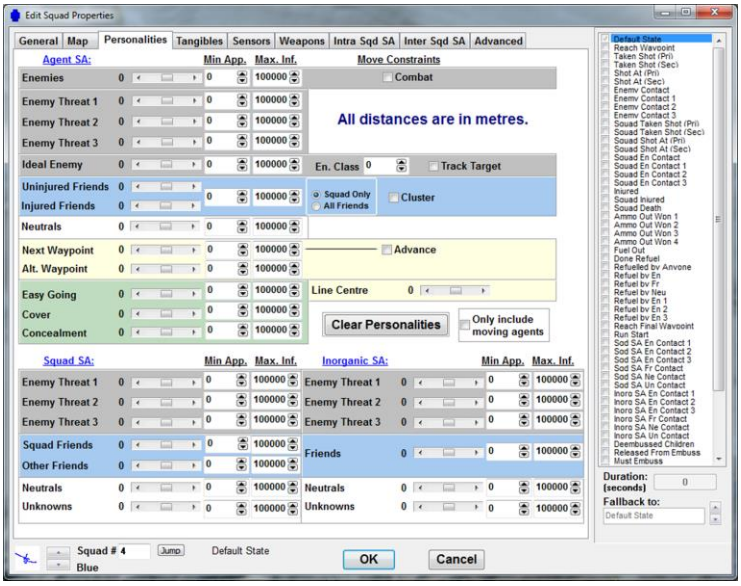

Figure 35. Blue Artillery Personality Settings, from [21].

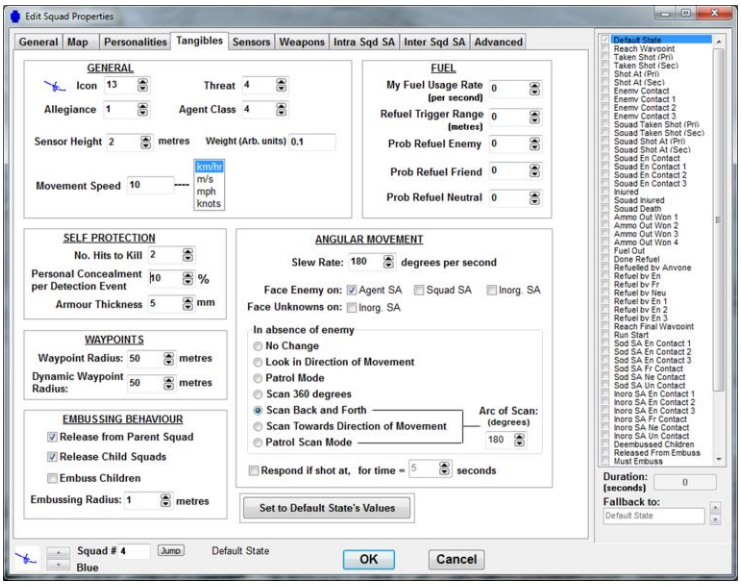

Figure 36. Blue Artillery Tangibles Settings, from [21].

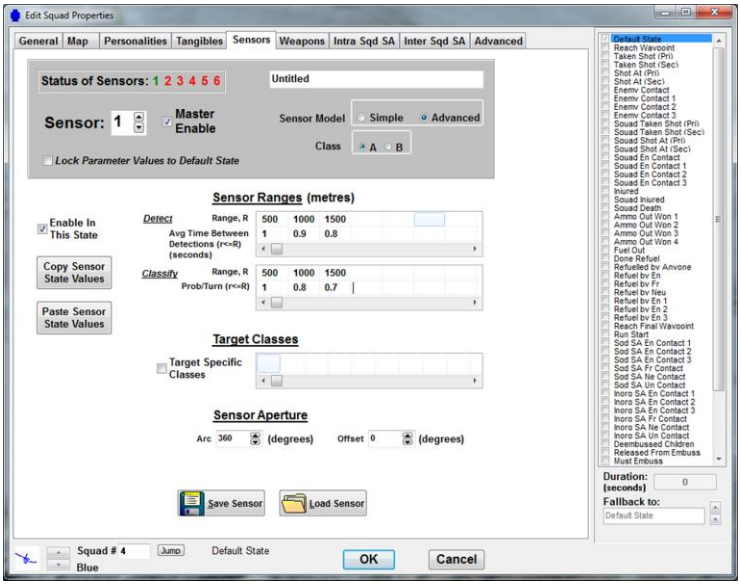

Figure 37. Blue Artillery Sensors Settings, from [21].

The major difference between the settings for the artillery and those of the infantry squads is how the weapons are designated. In the case of the artillery the type of weapon is specified as being a high explosive/inorganic situational awareness weapon. This indicates that the probabilities of hit are calculated from the point of impact and not from the shooter. Using the inorganic situational awareness setting indicates that this

weapon engages targets based upon what the infantry squads are "reporting". This is similar to calling for fire in combat as the artillery does not necessarily need to observe the target themselves to engage it. There is also a minimum target range setting that can be specified for the artillery as well as a maximum range. In MANA the artillery always hit the targeted area and the probability of hit of agents within the blast radius is specified within the table. It is also important to pay attention to the protect contact types. Checking the boxes for squad friends and other friends implies that the artillery will not engage targets if there are friendly forces within the blast radius of the round. Not checking this box would allow the infantry to call "danger close" and there would be some probability that they would be hit as well as the enemy. The settings for ammo and reload time are also important since the artillery is not a machine gun. However, these would be important to research prior to conducting an in-depth scenario research project. Figure 38 displays artillery weapon settings. Figure 39 displays intra-squad situational awareness settings.

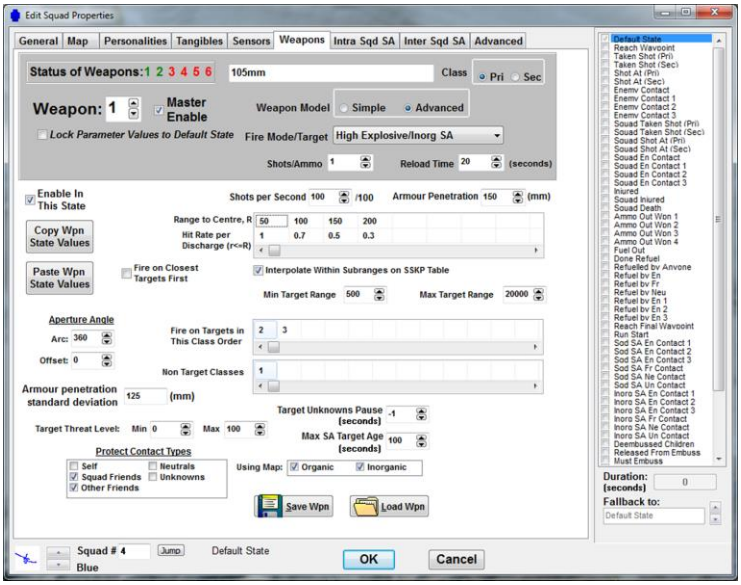

Figure 38. Blue Artillery Weapon Settings, from [21]

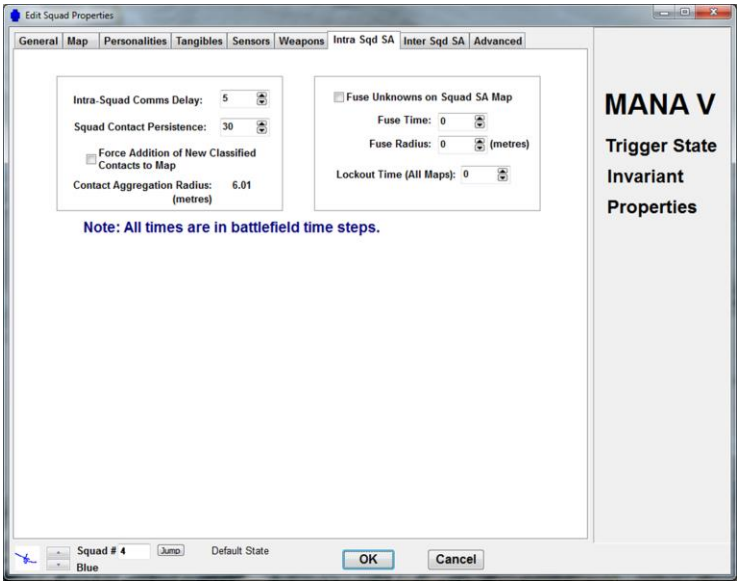

Figure 39. Blue Artillery Intra Squad SA Settings, from [21].

Since the artillery section will not maneuver they will not need to send reports to the forward infantry sections. The default settings are maintained for the inter squad SA tab, displayed in Figure 40.

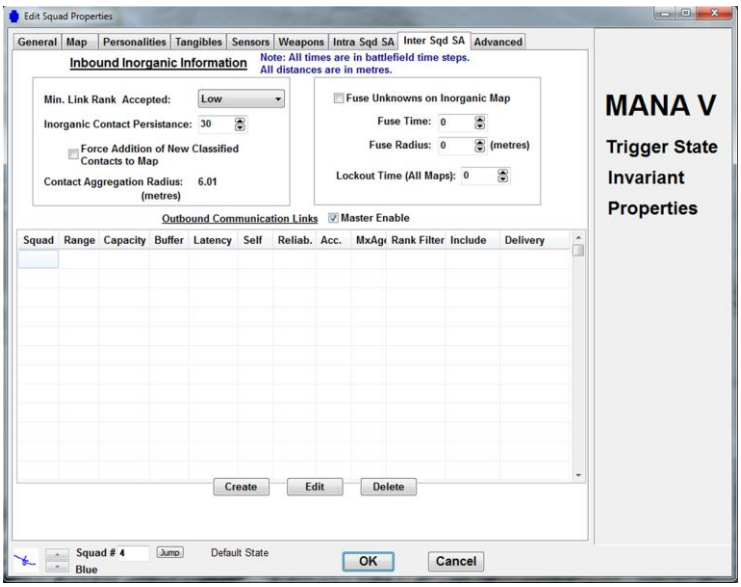

Figure 40. Blue Artillery Inter Squad SA Settings, from [21].

The artillery section maintains all default settings in the advanced tab.

4. Red Force Artillery Settings

The Red Force Artillery has the same settings as the blue force artillery except that they target agent class 1 and are located in the northern part of the battlefield. The red artillery will not maneuver and will receive "reports" from the maneuvering tank and BMP platoons. Figure 41 displays map locations for the Red Force Artillery. Figure 42 displays the weapon settings for the Red Artillery.

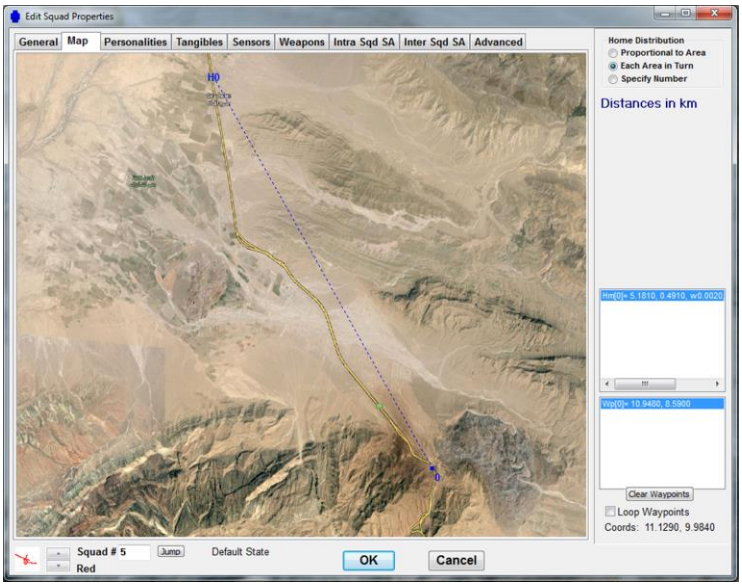

Figure 41. Red Artillery Map Location, from [21].

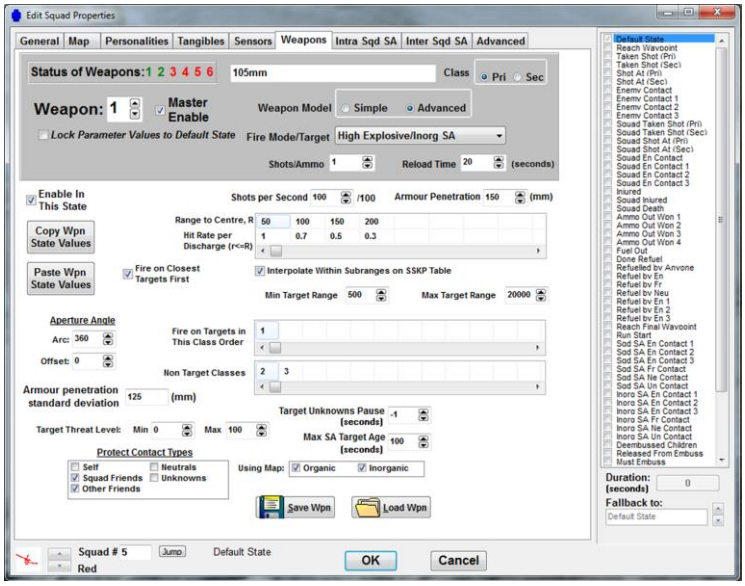

Figure 42. Red Artillery Weapon Settings, from [21].

5. Red Force Tank Platoon Settings

The tank platoon maneuvers south and possesses different settings than what was established for the defensive minded infantry squads. The general settings are displayed in Figure 43.

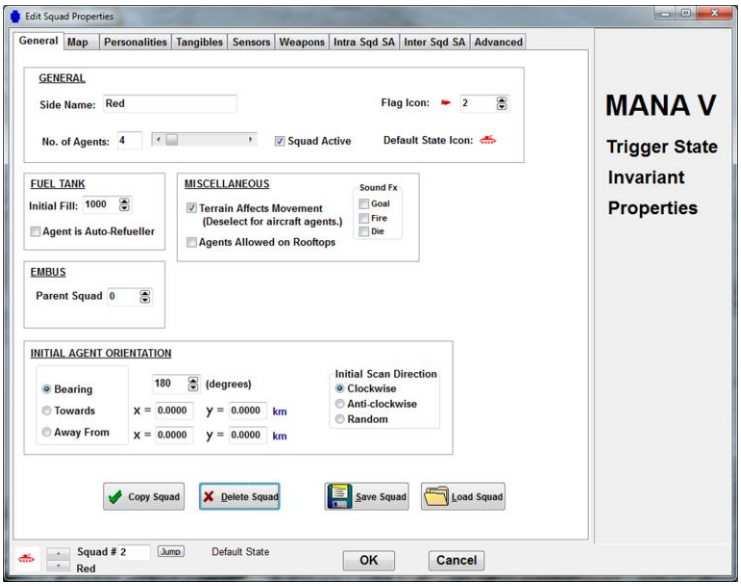

Figure 43. Red Force Tank General Settings, from [21].

Since the platoon is maneuvering against the blue forces, it is important to provide direction for the movement. This is done by the user in placing waypoints along a specified route, displayed by Figure 44. The user can then later determine how closely the platoon follows to the rout and how far off the route the platoon is allowed by altering specific settings within the "advanced" tab. In this case, the default settings are maintained.

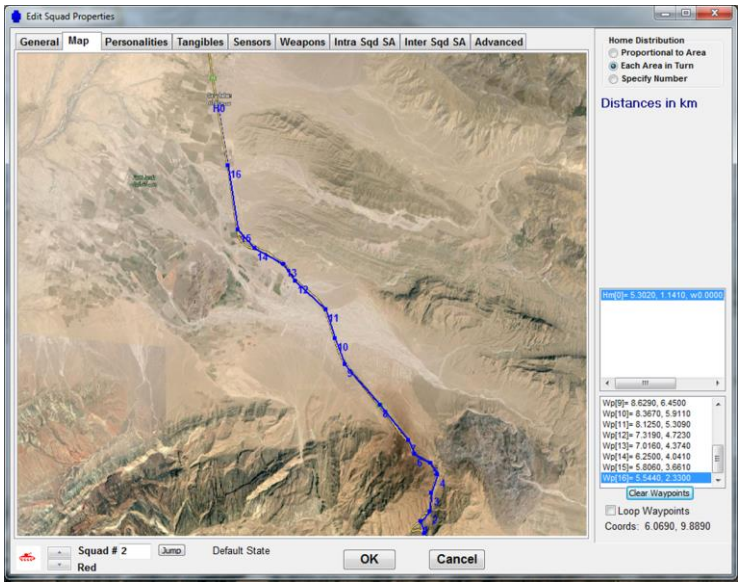

Figure 44. Red Force Tank Map Location, from [21]

The personality weightings are where the biggest difference between the defensive infantry squads and the maneuvering tank platoon. The user defines how the tanks maneuver with respect to how the agents perceive the world around them. The desire to maneuver against enemies; however, the desire to continue along the specified route towards the end objective is far stronger. It would not be expected that a tank would want to attempt to maneuver into the mountainous areas so there is a large negative against moving in those areas as they are defined by their cover and concealment values in the terrain map.

If there were units maneuvering in different areas the user could define different personality weightings for the squad and inorganic situational awareness. Thus, if a

different squad came into contact with an enemy in another area, the tank could be made to have a desire to change direction to maneuver closer to that reported enemy. In this case the agents only maneuver against what they individually perceive. Figure 45 shows established personality settings.

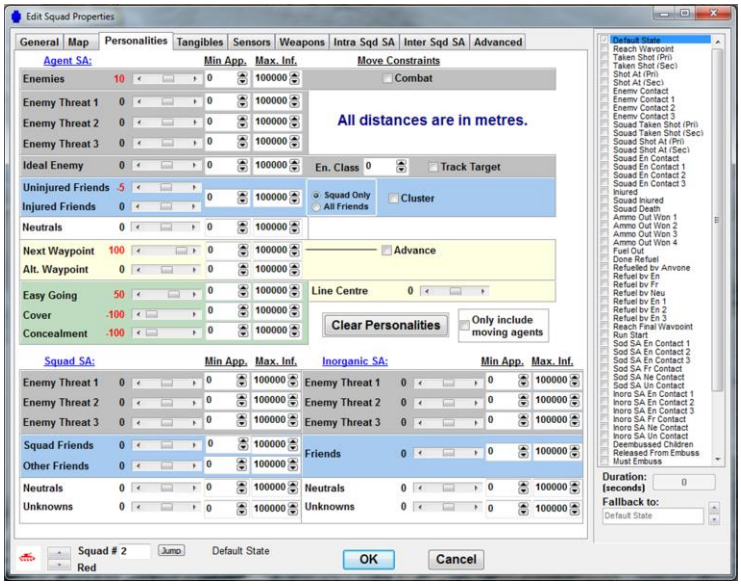

Figure 45. Red Force Tank Personality Weightings, from [21].

The number of hits to kill is set at two based upon the type of weapon being employed against the tanks. An important attribute within the tangibles tab is the waypoint radius setting. This should set relatively large so that the agents can "reach" the waypoint without being right on top of it. Setting this too small slows the maneuver of the platoon as each agent attempts to get within the specified radius. This may conflict with minimum distance required between agents and prevent the platoon from reaching the waypoint and continuing the movement. The tangible settings are displayed in Figure 46.

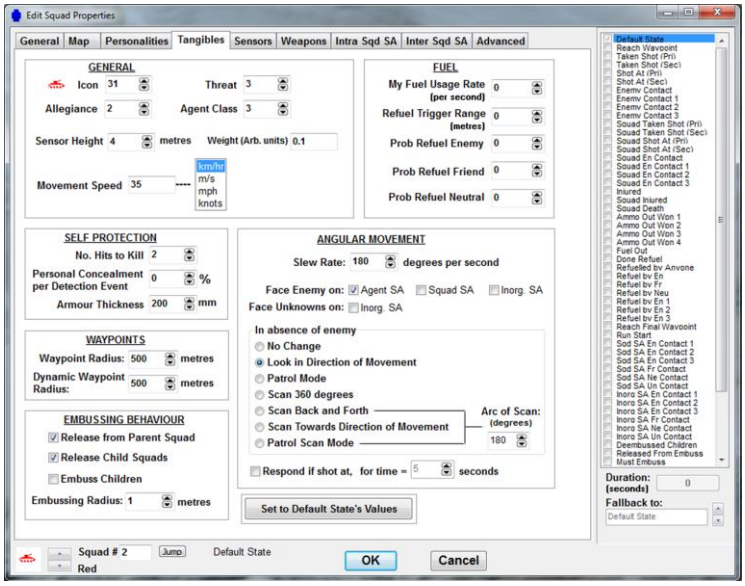

Figure 46. Red Force Tank Tangibles Settings, from [21].

In this tutorial the simple settings are used for the sensor, displayed in Figure 47. The detection distance is set to 6000 meters with the classification being 5000 meters. It makes sense that the defense would be able to detect the attacking force earlier than the attacking force would be able to detect the defensive force due to the fact that the defensive force is "dug in" and has established observation on avenues of approach and the attacker is not aware of the defense. These values could play a vital role in the outcome of the battle and a user would have to take extreme care that the capabilities being modeled in the scenario were accurately quantified.

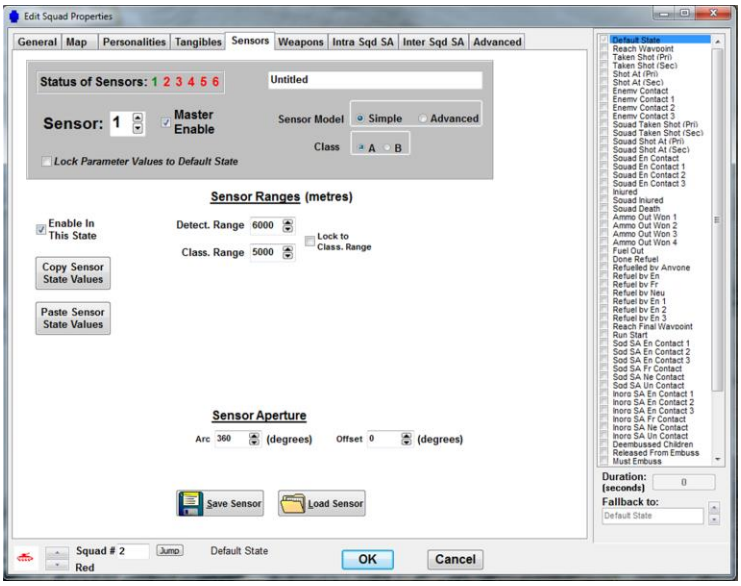

Figure 47. Red Force Tank Sensors Settings, from [21].

The tanks have two assigned weapons; the main gun and the coaxially mounted machine gun. Since they are not attacking any mounted units the primary weapon is established as the machine gun with the secondary gun being the main cannon of the tank, depicted in Figures 48 and 49. This could easily be reversed without any effect to the scenario if the user introduced mounted units on the blue side. In that case, the user would switch to the advanced settings and ensure that the main gun was not utilized against dismounted infantry.

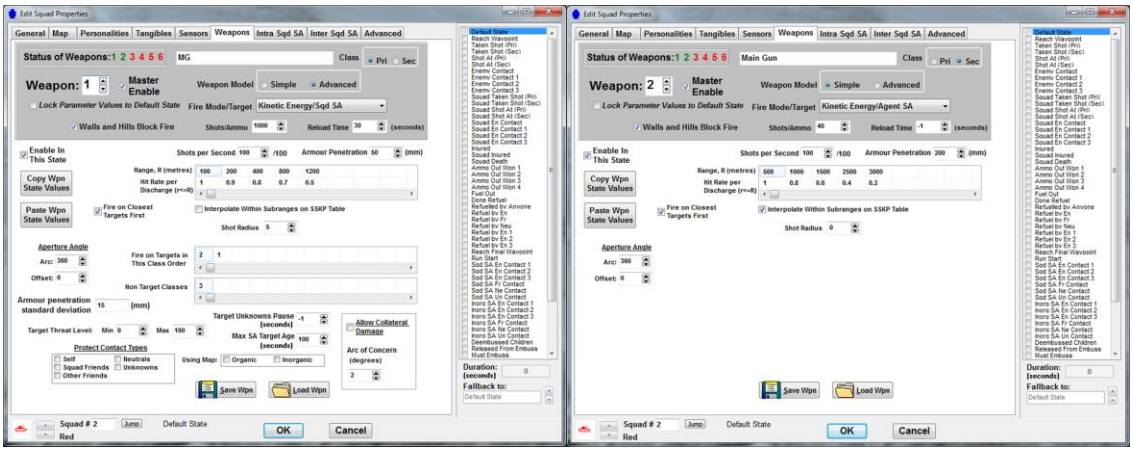

Figure 48. Red Force Tank Weapon Settings, from [21].

The intra-squad situational tab, shown in Figure 49, establishes a 5 second delay to simulate the time it takes to send a contact report, but all other default settings are maintained. The inter-squad settings establish two links; one between the tank platoon and the BMP platoon, and one between the tank platoon and the artillery section.

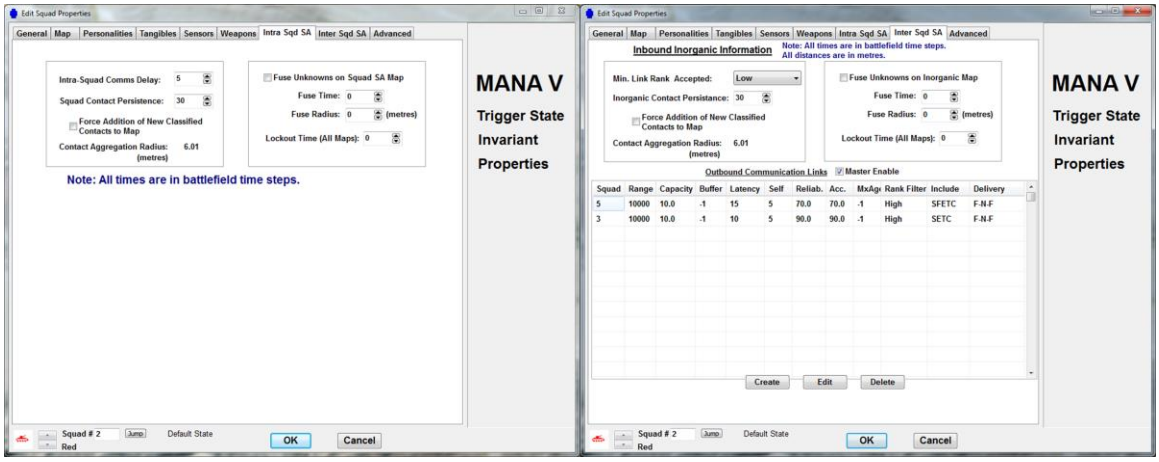

Figure 49. Red Force Tank Situational Awareness Settings, from [21].

In the tangibles tab, shown in Figure 50, the user establishes a wedge as being the preferred movement formation. Keeping the personality weighting for the formation to zero prevents the squad from retreating and reforming into a wedge after they have taken a causality. Instead, they continue to move forward and reform on the way. All the other defaults are maintained.

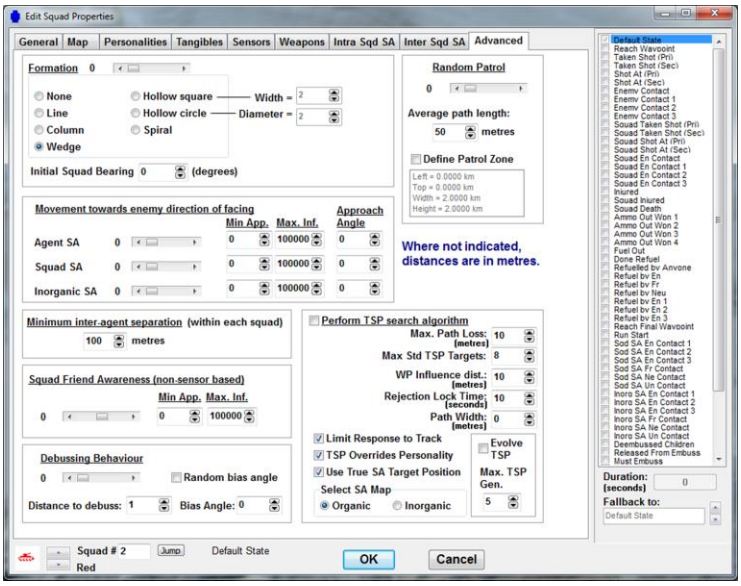

Figure 50. Red Force Tank Advanced Settings, from [21].

6. Red Force BMP Platoon Settings

The BMP platoon has six vehicles and is placed to maneuver to the rear of the tank platoon along the same route. They have the same personality weightings determining how they maneuver, but have differences in the amount of armor the agents possess. The BMP only has 50mm of armor. The general settings and map placement for the BMP platoon are shown in Figure 51. Personality and tangible settings are displayed in Figure 52.

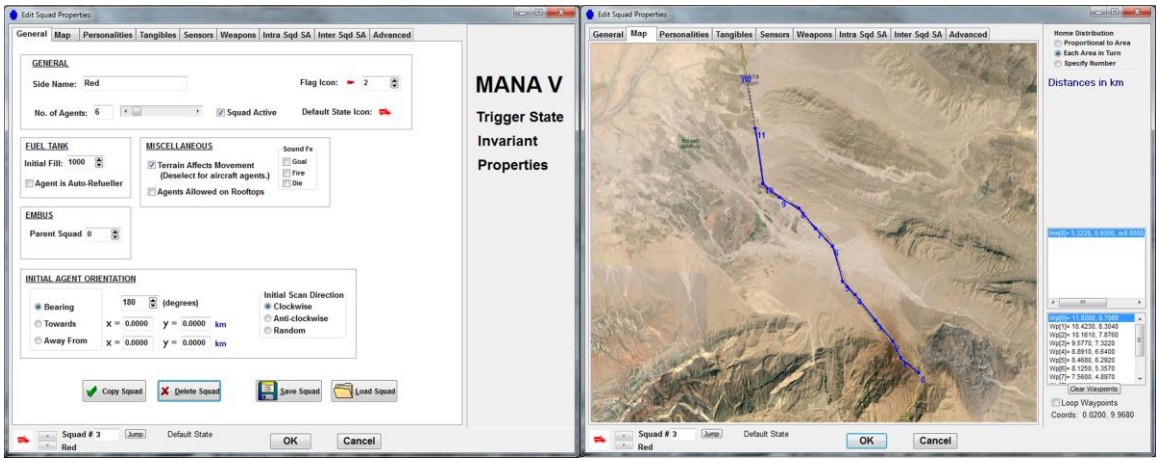

Figure 51. Red Force BMP General Settings and Map Placement, from [21].

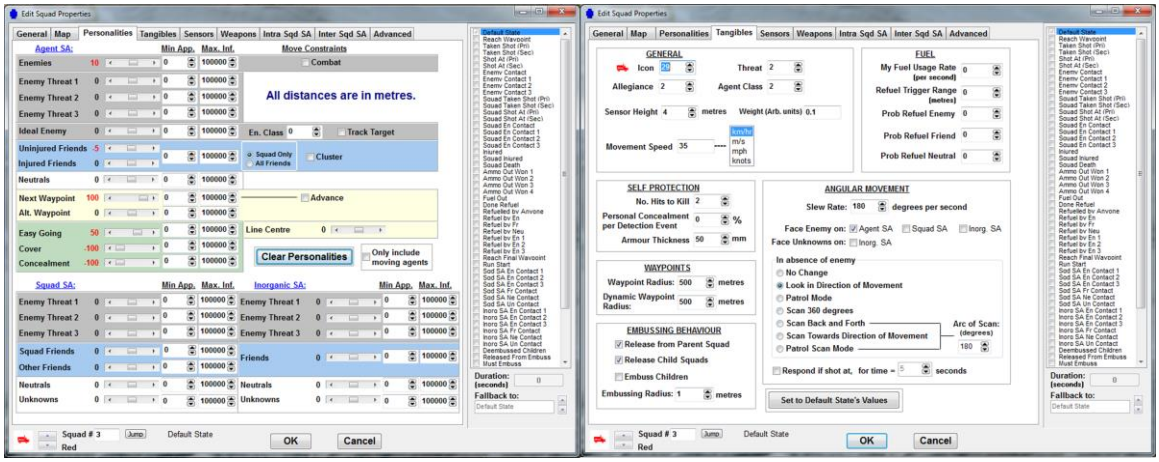

Figure 52. Red Force Personalities and Tangibles Settings from, [21].

The BMP platoon has the same sensors settings as the tank platoon, shown in Figure 54. The primary difference between the red force platoons is in the weapons being used, shown in Figure 53. The BMP platoon has one weapon, the 30mm cannon. This is similar to the main gun of the tank, but is has some different characteristics. The BMP has an ammo capacity of 250 rounds with the ability to reload in 30 seconds. The armor penetration is 50mm with a standard deviation of 15mm. The range is less than that of the tank main gun, only able to reach to 2000 meters. The 30mm cannon also possess a shot dispersion radius of 5 as it is an area fire weapon similar to the machine gun. Advanced settings are displayed in Figure 55.

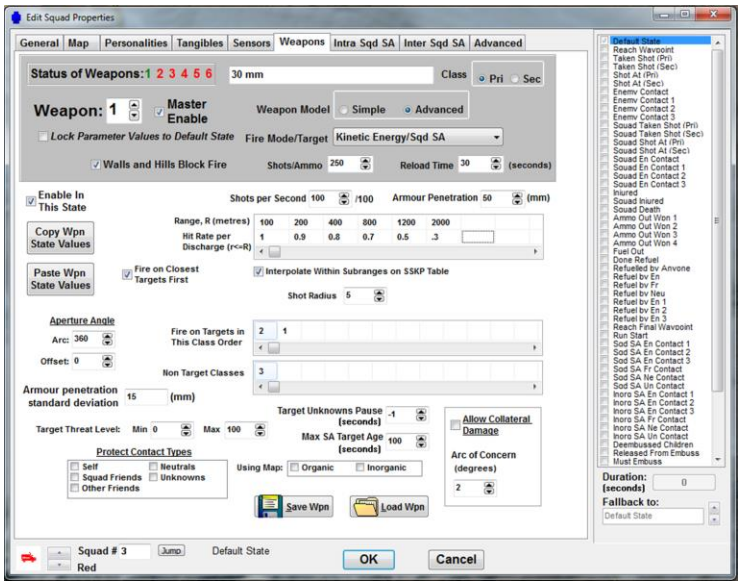

Figure 53. Red Force BMP Weapon Settings, from [21].

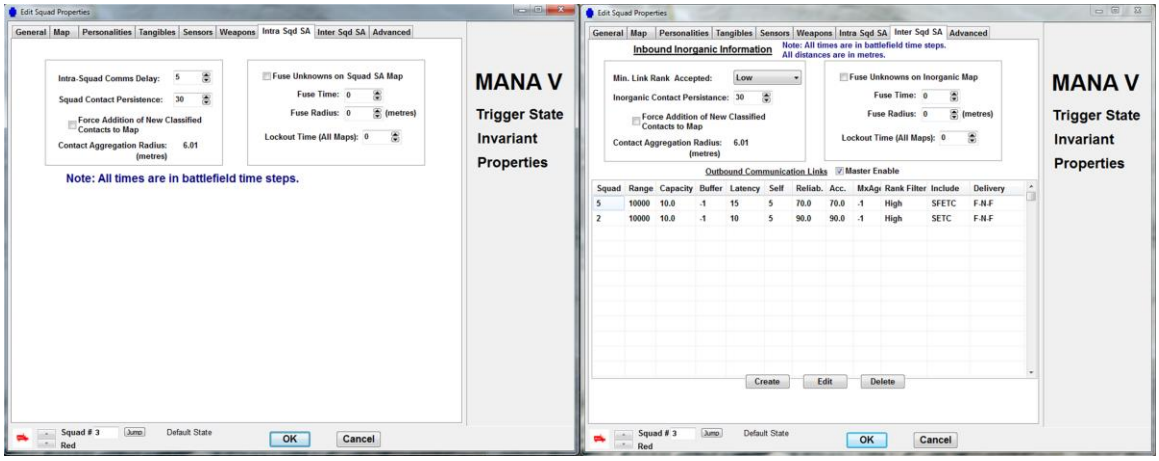

Figure 54. Red Force BMP Situational Awareness Settings, from [21].

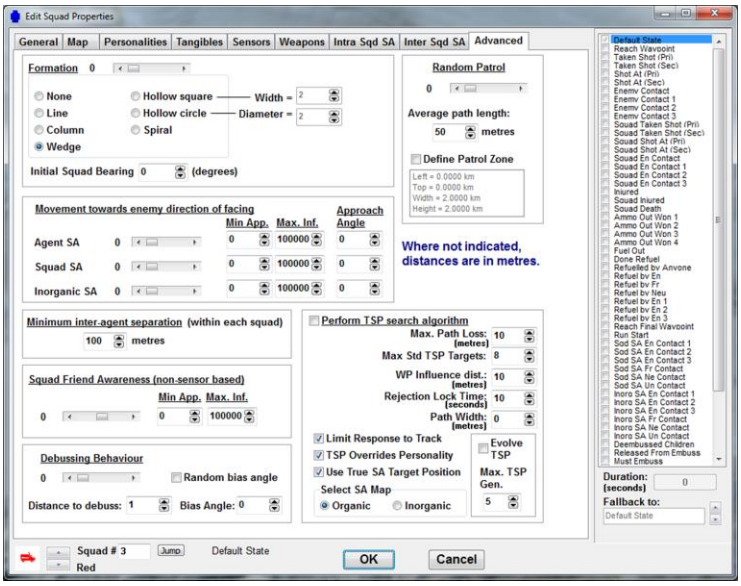

Figure 55. Red Force BMP Advanced Settings, from [21].

7. Running the Model

Once all the platoons are established, the battlefield defined and characterized, and the stop conditions specified the user is able to begin the scenario. By selecting the multi-run tab under the set up menu, the user establishes how many iterations of the scenario will be executed. Once this selection is made, pressing multi run button on the upper right hand side of the user interface causes MANA to begin executing the iterations of the scenario. If the user has made selections from the "Data Outputs" menu, MANA begins to store specified data the individual iterations. The user is then able to conduct a statistical analysis of the data.

D. DATA EXTRACTION

Data extraction is very easy while utilizing MANA V. At the top of the user interface selecting data outputs opens the drop down menu where the user can select from nine choices for MANA to display data obtained from the scenario. A check mark shows next to the selections that the user has made. MANA creates an EXCEL file that is filled with the selected data. When selecting multiple runs MANA creates a summary file (displayed in Table 8) that includes means and standard error for red and blue causalities, if either squad reached the goal, and run time. The other files that are commonly utilized

are; record causality location data, record agent state data, and record multi-contact detections. However, the other selection could be important depending on what the user is trying to gain insight on from the particular scenario. [26].

# MANA MultiRun Summary File																							
# Version: 5.01.06																							
# Machine Name: IT150979																							
# Scenario	\\special\jrwilli1S\Desktop\MANAV_32bit_20130710[1]\tutorial_scenario.xml																						
# MultiRun Started	5/7/2014	9:21:01 AM																					
Run	Seed	Alleg1Cas(Blue)	Alleg2Cas(Red)	Blue Reach Goal	Red Reach Goal	Steps	Sqd1Cas	Sqd2Cas	Sqd3Cas	Sqd4Cas	Sqd5Cas	Sqd6Cas	Sqd7Cas	Sqd8Cas	Sqd1Inj	Sqd2Inj	Sqd3Inj	Sqd4Inj	Sqd5Inj	Sqd6Inj	Sqd7Inj	Sqd8Inj	
1	-1857509513	40	9	No	No	747	10	3	6	1	0	10	10	9	0	1	0	0	0	0	0	1	
2	-1164441075	14	10	No	No	622	7	4	6	0	0	5	2	0	0	0	0	0	0	4	5	5	
3	1379616596	6	10	No	No	371	0	4	6	0	0	6	0	0	4	0	0	0	0	3	8	0	
4	-71173002	14	10	No	No	627	7	4	6	0	0	2	5	0	2	0	0	1	0	7	4	5	
5	-1228942222	5	10	No	No	389	0	4	6	0	0	5	0	0	9	0	0	0	0	2	8	0	
# MultiRun Finished	5/7/2014	9:24:46 AM																					
		BlueCas	RedCas	BlueGoal	RedGoal	Steps	RealTime																
Means		15.8	9.8	0	0	551.2	551.2																
Std Errors		6.343501	0.2	0	0	73.4428	73.44277																
# Output Ends																							

Table 8. Summary Output Data from Multi Run Scenario, from [27].

The casualty location data file, displayed in Table 9, displays each agent that was killed in chronological order. It displays where they were killed, at what time step, which agent killed them and with which weapon, and their location.

# MANA Casualty Location Results File																
# Version: 5.01.06																
# Machine Name: IT150979																
# Run Star	9:11:10 AM															
RandSeed= 0																
ID	x-casualty	y-casualty	time	Cas Squad	Cas Squad	Cas alleg	subsquad	state	shooter ID	Squad ID	Sqd name	weapon cl	weapon ID	x-shooter	y-shooter	shooter alleg
17																
12	5558	2627	171	Red	2	2	0	Default St	20	4	Blue	Primary	1	10887	8438	1
18	5448	2481	171	Red	3	2	0	Default St	20	4	Blue	Primary	1	10887	8438	1
19	5617	2691	192	Red	3	2	0	Default St	21	4	Blue	Primary	1	10887	8551	1
15	5445	2590	192	Red	3	2	0	Default St	20	4	Blue	Primary	1	10887	8438	1
24	8851	7246	278	Blue	6	1	0	Default St	23	5	Red	Primary	1	5221	531	2
14	6262	3786	343	Red	3	2	0	Default St	20	4	Blue	Primary	1	10887	8438	1
17	6359	3880	343	Red	3	2	0	Default St	21	4	Blue	Primary	1	10887	8551	1
27	8951	7215	357	Blue	6	1	0	Default St	22	5	Red	Primary	1	5142	452	2
10	6673	4003	358	Red	2	2	0	Default St	43	7	Blue	Primary	1	9089	7185	1
26	8870	7240	362	Blue	6	1	0	Default St	23	5	Red	Primary	1	5221	531	2
30	8891	7233	362	Blue	6	1	0	Default St	23	5	Red	Primary	1	5221	531	2
31	8971	7209	362	Blue	6	1	0	Default St	23	5	Red	Primary	1	5221	531	2
13	6582	4139	368	Red	2	2	0	Default St	42	7	Blue	Primary	1	9151	7183	1
11	6875	4284	400	Red	2	2	0	Default St	44	8	Blue	Primary	1	9796	7007	1
33	8932	7221	425	Blue	6	1	0	Default St	22	5	Red	Primary	1	5142	452	2
32	8911	7227	467	Blue	6	1	0	Default St	23	5	Red	Primary	1	5221	531	2
16	7262	4667	485	Red	3	2	0	Default St	21	4	Blue	Primary	1	10887	8551	1
# Run End																

Table 9. Agent Casualty Location Data File, from [27].

The agent state data output, depicted in Table 10, provides the end status of each agent on the battlefield. It displays where the agent was at the end of the scenario, the location of its goal, how many hits it took, the amount of ammo remaining, and what trigger state it was in if different than the default status.

# MANA Agent End State Results File																		
# Version: 5.01.06																		
# Machine Name: IT150979																		
# Run Star	9:21:01 AM																	
RandSeed= -1857509513																		
ID	name	squad	subsquad	x	y	status	fuel	goalx	goaly	hits	state	trig_step	ammo1	ammo2	ammo3	ammo4	ammo5	ammo6
0	Blue_agen	Blue	0	9333.4	7066.4	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
1	Blue_agen	Blue	0	9395.5	7054.4	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
2	Blue_agen	Blue	0	9313.3	7070.3	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
3	Blue_agen	Blue	0	9459	7042	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
4	Blue_agen	Blue	0	9499.9	7034.1	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
5	Blue_agen	Blue	0	9373.9	7058.6	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
6	Blue_agen	Blue	0	9436.7	7046.4	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
7	Blue_agen	Blue	0	9478.8	7038.2	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
8	Blue_agen	Blue	0	9416.5	7050.3	Dead	1000	2621	254	5	Default St	0	0	498	0	0	0	0
9	Blue_agen	Blue	0	9353.7	7062.5	Dead	1000	2621	254	5	Default St	0	0	500	0	0	0	0
10	Red_agen	Red	0	9031.6	6613.1	Injured	1000	9556	7290	1	Default St	0	945	0	0	0	0	0
11	Red_agen	Red	0	8657.1	6474.3	Dead	1000	9556	7290	2	Default St	0	994	0	0	0	0	0
12	Red_agen	Red	0	6868	4249.2	Dead	1000	7319	4723	2	Default St	0	1000	40	0	0	0	0
13	Red_agen	Red	0	8823.8	6522.2	Dead	1000	9556	7290	2	Default St	0	956	0	0	0	0	0
14	Red_agen	Red	0	8698.4	6511	Dead	1000	9577	7322	2	Default St	0	120	0	0	0	0	0
15	Red_agen	Red	0	5543.3	2560.1	Dead	1000	5786	3708	2	Default St	0	250	0	0	0	0	0
16	Red_agen	Red	0	5481.1	2467.1	Dead	1000	5786	3708	2	Default St	0	250	0	0	0	0	0
17	Red_agen	Red	0	8723.2	6406.3	Dead	1000	9577	7322	2	Default St	0	128	0	0	0	0	0
18	Red_agen	Red	0	8631.9	6320.5	Dead	1000	9577	7322	2	Default St	0	143	0	0	0	0	0
19	Red_agen	Red	0	8761	6305.9	Dead	1000	9577	7322	2	Default St	0	135	0	0	0	0	0

Table 10. Agent State Data File, from [27].

The multi-contact detection data file, depicted in Table 11, provides details on when and where agents were detected and by whom.

# MANA Multi-Contact Detection Results File								
# Version: 5.01.06								
# Machine Name: IT150979								
# Run Star	9:21:01 AM							
RandSeed= -1857509513								
Step	Squad of Detector	Squad of Classified Agt	Detector Agent	Classified Agent	x	y	Range	Detector Deadtime
246	1	5	13	32	8803	7242	4994	0
247	1	5	13	28	8823	7241	4997	0
248	1	5	13	25	8845	7240	5000	0
250	1	5	12	31	8865	7240	4991	0
251	1	5	12	24	8887	7239	4994	0
252	1	5	12	33	8908	7238	4997	0
253	1	5	12	26	8928	7237	4999	0
255	1	5	12	27	8949	7237	4993	0
256	1	5	12	29	8971	7236	4996	0
257	1	6	12	36	9028	7196	4991	0
258	1	5	12	30	8991	7235	4989	0
259	1	6	12	34	9071	7194	4998	0
259	1	6	13	39	9049	7195	4998	0
261	1	6	12	43	9091	7193	4991	0
261	2	5	17	28	8823	7241	4996	0
261	2	5	17	32	8803	7242	4984	0
262	1	6	12	42	9114	7193	4995	0
262	2	5	17	25	8845	7240	4999	0
263	1	6	12	37	9134	7192	4998	0
265	2	5	17	31	8865	7240	4982	0

Table 11. Multi-Contact Detection Data File, from [27].

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] A. Tolk, *Engineering Principles of Combat Modeling and Distributed Simulation*, Hoboken, NJ: Wiley & Sons, 2012.

[2] "MANA overview," lecture notes for OA4655, lecture notes, Dept. of Operations Research, Naval Postgraduate School, Monterey, CA, Nov. 2013.

[3] G. C. McIntosh, D. P. Galligan, M. A. Anderson, and M. K. Lauren, *MANA (Map Aware Non-Uniform Automata) version 4 user manual*, Defense Technology Agency, Auckland, New Zealand, Tech. Note 2007/3 NR1465, May 2007.

[4] A. Lancinskas and J. Zilinskas, "Methods for generation of random numbers in parallel stochastic algorithms for global optimization," *Journal of Young Scientists*, vol. 27, no. 2, pp. 118–123, 2010.

[5] "Linear, congruential random number generators," lecture notes for GS 510, Dept. of Mechanical Engineering, Colorado State University, Fort Collins, CO: Fall 2004.

[6] D. E. Knuth, "Random numbers," in *The Art of Computer Programming*, 3rd ed. Boston, MA: Addison-Wesley, 1998, ch. 3, sec. 1–2, pp. 1–40.

[7] K. Entacher, "A collection of selected pseudorandom number generators with linear structures," Aug. 1997.

[8] C. N. Zeeb, and P. J. Burns, "Random number generator recommendation," Dept. of Mechanical Engineering, Colorado State University, Fort Collins, CO: 1997.

[9] W. Trappe, and L. C. Washington, "Classical cryptosystems," in *Introduction to Cryptography with Coding Theory*, 2nd ed. Upper Saddle River, NJ: Pearson Education, 2006, ch. 2, sec. 10, pp. 41–43.

[10] J. Wehrwein, "Random number generation," senior thesis, Middlebury College, Middlebury, VT, 2007.

[11] K. H. Rosen, "Number Theory and Cryptography," in *Discrete Mathematics and its Applications*, 7th ed. New York: McGraw-Hill, 2012, ch. 4, sec. 1, 3, pp. 237–244, 257–272.

[12] S. K. Park, and K. W. Miller, "Random number generators: good ones are hard to find," *Communications of ACM*, vol. 31, no. 10, pp. 1192–1201, Oct. 1988.

[13] J. Boyar, "Inferring sequences produced by pseudo-random number generators," *Journal of the Association for Computing Machinery*, vol. 36, no. 1, pp. 129–141, Jan. 1989.

[14] A. W. Gill, "Improvement to the movement algorithm in the MANA agent-based distillation," *Journal of Battlefield Technology*, vol. 7, no. 2, pp. 19–22, July 2004.

[15] A. W. Gill, and D. Grieger, "Validation of agent-based distillation movement algorithms," DSTO Systems Sciences Laboratory, Edinburgh, Australia, 2003.

[16] D. Grieger, "Comparison of two alternative movement algorithms for agent-based distillations," Land Operations Division Defense Science and Technology Organization, Edinburgh, Australia, 2007.

[17] G. C. McIntosh, *MANA-V (Map Aware Non-Uniform Automata –Vector) Supplementary Manual*, Defense Technology Agency, Auckland, New Zealand, Tech. Note 2009/7 NR 1525, Sept. 2009.

[18] J. Stewart, "Vectors and the geometry of space," in *Calculus Early Transcendentals*, 7th ed. Belmont, CA: Cengage Learning, 2012, ch. 12, sec. 2, pp. 791–798.

[19] G. C. McIntosh, private communication, Feb. 2014.

[20] J. Strickland, "Physical models of attrition for passive targets," in *Mathematical Modeling of Warfare and Combat Phenomenon*, 1st ed., Colorado Springs, CO: Lulu, 2011, ch. 2, pp. 17–50.

[21] Michael K. Lauren, *Map Aware, Non-Uniform Automata version 5.0*. [Licensed download]. New Zealand Defense Technology, Sept. 2009.

[22] "Bader Abbas Lanchester Equation Project," lecture notes for OA4655, Dept. of Operations Research, Naval Postgraduate School, Monterey, CA, 2012.

[23] Google, *Google Earth*. [Online]. Palo Alto, California: Google, 2014.

[24] Microsoft, *Paint*, [Installed software]. Redmond, WA: Microsoft, 2010.

[25] M. McDonald, *Edit squad—weapons*, Mpeg4 video. Cle.nps.edu. Oct. 18, 2013.

[26] M. McDonald, *Other data outputs options*, Mpeg4 video. Cle.nps.edu. Oct. 18, 2013.

[27] Microsoft, Microsoft Excel. [Licensed download]. Redmond, WA: Microsoft, 2010.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California