15th ICCRTS

"THE EVOLUTION OF C2"

Plans Validation using DES and Agent-Based Simulation

Topic: Modeling and Simulation

Mr. Andy Ong Kim Soo
Defence Science and Technology Agency
1 Depot Road,
Defence Technology Tower A, Singapore 109679
Telephone: +65 63736078
Andy_Ong@dsta.gov.sg

Mr. Andrew Wong Teck Hwee
Defence Science and Technology Agency
1 Depot Road,
Defence Technology Tower A, Singapore 109679
Telephone: +65 63736513
wteckhwe@dsta.gov.sg

Dr. Christian J. Darken
Naval Postgraduate School
WA-382
Monterey, CA 93943
Telephone: (831) 656-2095
cjdarken@nps.edu

Dr. Arnold H. Buss
Naval Postgraduate School
ME-380
Monterey, CA 93943
Telephone: (831) 656-3529
abuss@nps.edu

Abstract

Military plans validation is typically a long and costly process requiring planners to validate their plans using anticipated scenarios or through military exercises. While military exercises provide realistic simulation of the plan, it is often the most expensive way of validating a plan. On the other hand, although using anticipated scenarios is relatively cheaper, the robustness of the validated plans is dependent on the scenarios against which they are validated. This, in turn, depends on the experience of the planners that crafted the scenarios.

This paper describes research on an alternative way of plans validation in the context of air defence done at Naval Postgraduate School as part of a postgraduate course of study. It explores the possibility of using a multi-agent system (MAS) to analyse air defence plan and generate potential air strike plans that exploit weaknesses in the air defence plan. The resulting plans are fed into a low resolution Discrete Event Simulation (DES) based air defence simulator to simulate the effects of the air strike plan against the air defence plan. A prototype was developed and has demonstrated the ability to validate air defence plans using MAS-generated strike plans and a low resolution DES-based simulator successfully.

1. Introduction

Much of military planning today, whether offensive or defensive, is based on expected adversary course of action, tactics and doctrine. If the adversary manages to produce an unexpected course of action which is not anticipated in the plan, this will lead to the adversary gaining a tactical advantage, and in the worse scenario, allow the adversary a strategic edge.

Military plans are often difficult to validate and verify. Unless they are put to use in a live exercise or operation, it is difficult to know the actual effectiveness of the plan.

This paper describes research conducted in Naval Postgraduate School to explore the feasibility of using software agents modelled after hypothetical adversary's behaviour to validate against military plans through discrete event simulations. A prototype based on an air defence scenario was developed to demonstrate the concept.

2. Methodology

This section describes the overall design of the system comprised of the agent-based strike plan generator and the Discrete Event Simulation (DES) based simulator.

2.1 Overall System Design

The plans validation system is comprised of a DES engine, an agent-based strike plan generator, a display interface, and other supporting components. The DES engine models the abstract behavior of air defence assets, air strike aircraft, and the interaction between them.

The DES engine serves as a platform for validating air defence plans. The plans are evaluated by simulating the effects of agent generated strike plans against the defence plans. In other words, the DES engine forms the environment in which the aircraft agents operate. Environmental updates are communicated to the aircraft agents through User Datagram Protocol (UDP) messaging. Similarly, when the aircraft agents perform evasive maneuvers, they communicate their new waypoints back to the DES engine via UDP messages. Such communication allows real time interactions between the agents and the environment. The overall system design is shown in Figure 1.

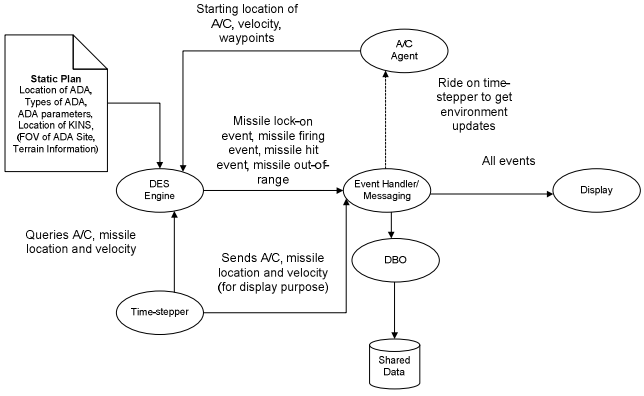

Figure 1: Overall Architecture of the System

2.2 Agent-based Plan Generation

The Agent-Based plan generator exploits the weaknesses in the air defence plan to generate potential adversary plans that are used to support the validation of the defence plans through the use of DES-based simulator. In this section, we will describe the agent architecture that was designed to mimic the command and control structure of a hypothetical air strike group.

Agent-Based Model Architecture

The goals of the agent based model are to generate appropriate strike plans for the agents representing strike aircraft, and to implement a behavior model for the agents in the simulation environment. The architecture of the Agent-Based Model is based on a hierarchical decision making process similar to the Hierarchical AI approach [1, 2]. Instead of having a single agent making plans, deciding where to strike, and determining how many aircraft formations should be created, the idea is to breakdown the decision-making process into levels. This is roughly analogous to the chain of command in an army where broad mission objectives at the strategic level are broken down into specific tasks at tactical levels, with tactical commanders making decision on the best approach to carry out such tasks to achieve the larger mission objectives.

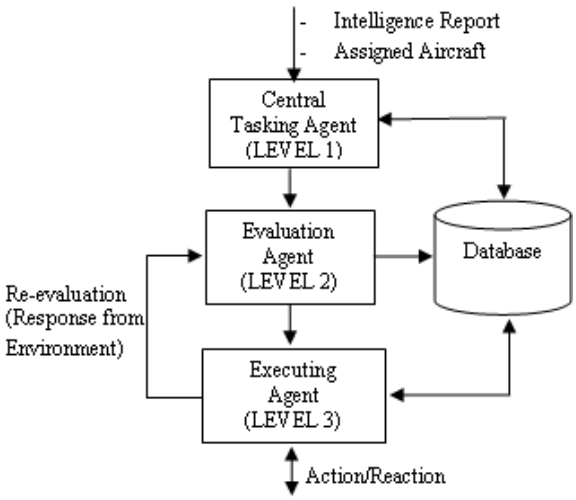

Figure 2: Agent-Based Model Architecture

In Figure 2, the agent-based model architecture consists of three levels of decision-making processes. At the highest level is the Central Tasking Agent, which is responsible for generating the number of participating air formations and assigning area of operations and targets to them. The assignment is based on intelligence information gathered on the target's air defence and the number of aircraft that are assigned to the strike operation.

At level two of the hierarchy is the Evaluation Agent which receives information on air formations, assigned area of operation and targets from level one agent. It will generate the participating aircraft in the air formation and also generate a suitable course of action for the air formation.

At level three of the hierarchy is the Executing Agent which is the lowest level in the agent architecture. The Executing Agent is like the foot soldier in the army, receive specific tasking orders from the Evaluation Agent, such as target objective and approach to the objective. This agent will receive real-time information from the simulation environment, and based on the information received, it will act on it and at the same time relay the information back to level two of the decision-making process so that the decision making agent at level two can re-evaluate the course of action.

The concept of how the agents exploit an air defence plan will be discussed later in this paper.

2.3 DES-Based Simulator Design

The DES-Based simulator provides a platform for evaluating air defence plans against the strike plans generated by the agent-based plan generator. It takes in the air defence plan crafted by human planners and the air strike plan generated by the agent-based generator. The DES simulator would then construct models of the components before starting the simulation. The DES simulator was designed based on the concept of Listener Event Graph Objects (LEGOs) framework [3], which allows a more complex model to be built in phases by linking smaller components together in a loosely coupled manner. The LEGO model of the DES simulator is shown in Figure 3. The DES simulator was developed based on Simkit [4]. The main elements in the DES simulator are: Surface-to-Air Missile (SAM) system and missiles, anti-aircraft guns and aircraft. Other supporting components include: mediators, adjudicators and adapters. Details on how various components are modelled will be described in later part of this paper.

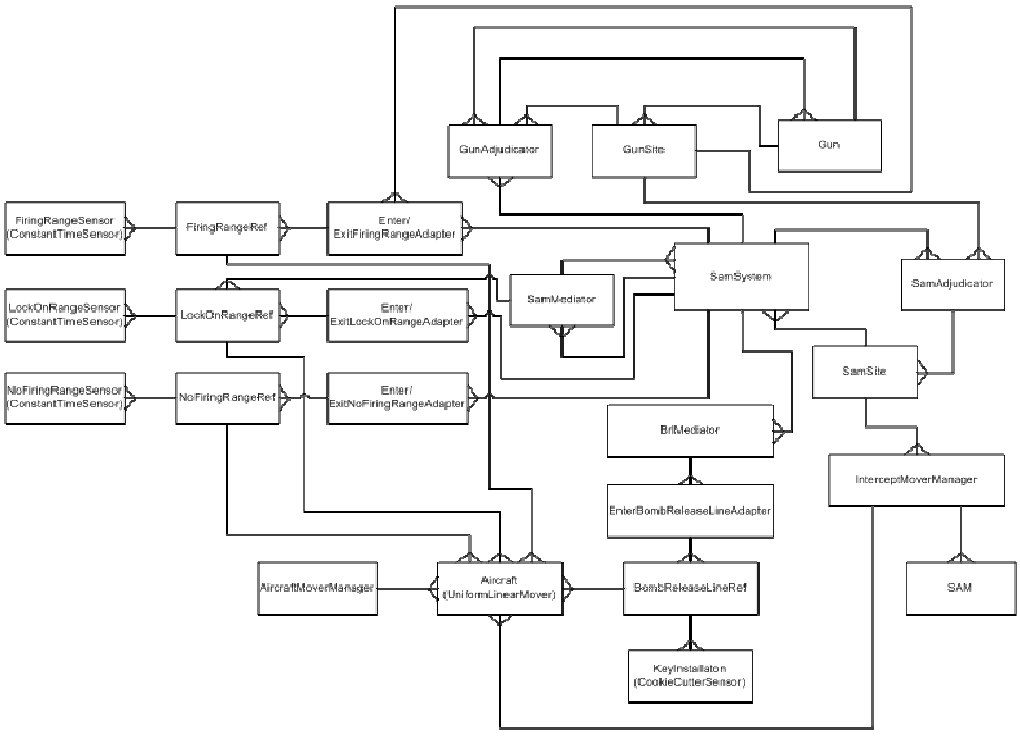

Figure 3: LEGO Model of the DES Engine

3. Prototype Implementation

3.1 DES-Based Simulator

The DES-Based simulator provides a platform for evaluating the air defence plan against the strike plan generated by the agent-based plan generator. The main models in the DES engine are: SAM system, Anti-Aircraft gun and aircraft. Other supporting components include: mediators, adjudicators, adapters and communications.

Modeling the SAM System

In the model, there is only one SAM system, which is the command and control unit of all the SAM sites in the DES engine. The SAM system has the overall situation awareness of the defended area and performs target handover from a SAM site to the other whenever a target gets out-of-range of a SAM site. The system ensures that at any one time, a target is engaged by only one SAM site. The firing option adopted in the model is the SHOOT-LOOK-SHOOT, which means, firing a missile once locked-on, observe the result, and fire another missile if the earlier missile missed the target.

Modeling the SAM Site

Each SAM site is comprised of a sensor and a missile launcher. In the model, a SAM site is only capable of locking and engaging one target at a time. A simple target selection algorithm is implemented based on the first-come-first-serve principle. Thus, when more than one target enters the sensor range, the first target will be locked-on, while subsequent targets will be put on the watched list. Whenever the first target is destroyed or gets out-of-range, the next target on the list will be acquired and locked. The missile launcher is loaded with a configurable number of missiles. The number of missiles in the launcher is decremented whenever a missile is fired. When all the missiles in the launcher are expended, a reload time will be incurred to reload the missile launcher. The model assumes perfect command and control so that a target is always handed over to the next SAM site that has the target in its sensor range.

Modeling the SAM Sensor

The use of DES for simulation of sensors and movers is not new. Buss and Sanchez [5] detail how movement and sensing can be modelled using DES. While the Gun sensor is modelled after the concepts presented in that paper, the SAM sensor is developed by extending the concept for sensing mover at multiple ranges.

A typical SAM sensor is capable of sensing targets that enter or exit its sensor range (lock-on range), weapon range (firing range) and the weapon's no-firing zone. In the SAM sensor model, three different sensors are used to model the various sensor ranges with their center aligned to the same static mover, which simulates the platform that the SAM sensor is installed physically. Figure 44 depicts how the SAM sensor is modelled, with the blue triangle indicating the location of the common platform.

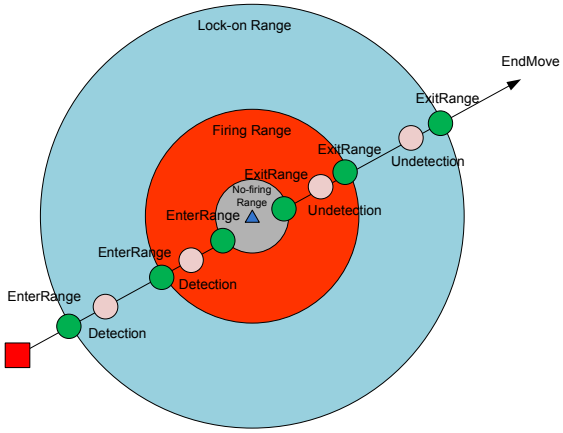

Figure 4: Various SAM Ranges

When an aircraft enters the "Lock-on Range" sensor, it will be detected and locked by the SAM system after a fixed amount of delay (SAM reaction time). Upon entering the "Firing Range", a missile will be fired at the target after a certain amount of delay (SAM engagement time). When a target enters the "No-firing Range", the SAM will not engage the target as there is too little time for the missile to be launched and catch up with the target.

The same "EnterRange" event triggered by the three sensors creates an issue as all "EnterRange" events are to be handled by the same mediator. The mediator is unable to differentiate which sensor initiated the event, since they have the same event name. Buss [6] proposes a simple but elegant way of overcoming such situation with the use of an "adapter" class. The "adapter" class listens to an event and triggers a new event. The same mechanism is used to overcome our problem as shown in Figure 5. The adapter works by adapting the same "EnterRange" event produced by the three sensors into specific events: EnterLockOnRange, EnterNoFiringRange and EnterFiringRange.

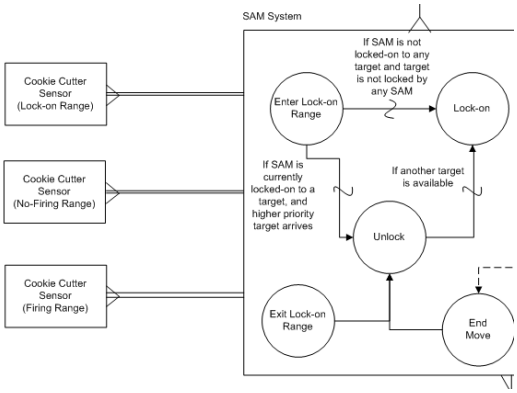

Figure 5: Use of Adapters to Differentiate EnterRange Events

Modeling SAM Trajectory

A simple path mover manager, which moves the SAM to an extrapolated interception point, is insufficient to model the trajectory of a SAM realistically, as a missile needs to respond to its target's manoeuvre by changing its own trajectory. The SAM model is built based on the concept of the intercept mover manager by Buss and Ahner [7].

The strength of an intercept mover manager is that it tracks the location and velocity of its target at regular time interval, re-computes the projected interception point and moves the missile towards the revised interception point. The intercept mover manager continues to track and re-project new interception point, until the mover is within certain proximity of the target.

In the event that a SAM flies out of its maximum range, the DES engine simulates a loss of command link and the SAM will be self destructed after a certain delay. This is a common feature available in current SAM systems.

SAM Probability of Kill

Instead of using a single-valued kill probability for SAM interception, a triangular Probability of Kill (Pk) contour was used, making Pk a function of the range of the target, as shown in Figure 6. This Pk contour could be easily substituted with any other more accurate Pk contours in the future when they become available. In the model, it is assumed that the highest Single-Shot Kill Probability (SSKP) is usually achieved at around 70% of the maximum weapon range. As such, the vendor supplied Pk becomes the height of the triangle in the Pk contour.

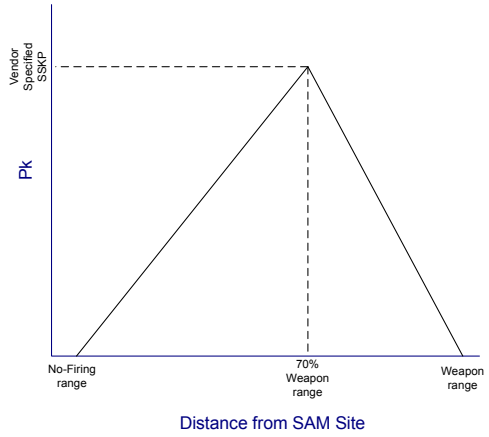

Figure 6. Triangular SAM Pk Contour

During simulation, a target is considered hit when the SAM is within certain proximity of the target. A uniform(0, 1) random number is generated using the Mersenne Twister (MT) random number generator. MT was chosen due to its long period of $2^{19937} - 1$ and a low working memory of 624 words [8]. The random number is then compared against the Pk value corresponding to the range of the target. Any number smaller than the Pk value is considered as a kill. Otherwise, it is considered as a miss and another missile will be scheduled to engage the target again.

Modeling the Anti-Aircraft Gun

According to the Field Manual 44-43 [9], a high volume of fire is desired to increase the probability of kill when engaging aerial targets with guns. Thus, unlike the SAM system, there is no restriction on the number of anti-aircraft guns allowed to engage a target simultaneously. The guns will engage any targets that get within the weapon range of the gun. As in the SAM system's case, the gun will engage its targets on a first-come-first-serve basis. Subsequent targets that enter the weapon range of a gun will be kept on a list. When the first target is destroyed, the gun will engage next target on the list sequentially.

As in the case of SAM system, the anti-aircraft gun also adopts the SHOOT-LOOK-SHOOT option. When a target enters the weapon range of a gun, the gun will release a burst of 50 to 100 rounds of munitions at the target. If the target is killed, the gun will move on to engage other targets in its target list, otherwise it will release another burst. The number of rounds to be released in each burst is based on a uniform(0, 1) generated by the MT random number generator and scaled to a number between range of 50 to 100. For each round of munitions released, the ammunition count is decreased by one. When the munitions run out, a delay is incurred for the reload event.

As the anti-aircraft gun in the model is not associated with any fire control radar, their target detection event could be modeled simply using a constant time sensor [5] as shown in Figure 7.

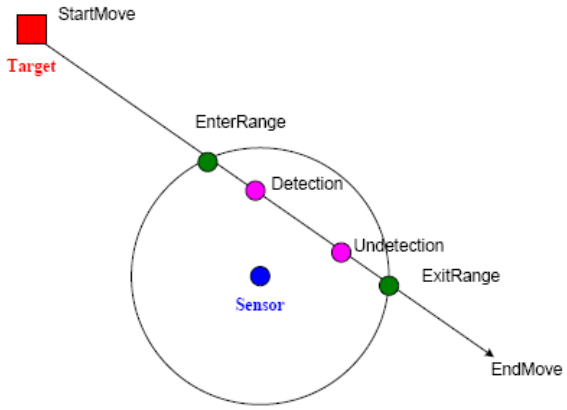

Figure 7: Anti-Aircraft Gun's Sensor

The trajectory and the probability of kill for each round of gun munitions are not modelled explicitly in the resolution if this model. Instead, the probability of kill for each burst of bullets is used to adjudicate the effects of the munitions on the target. In the gun's Pk contour shown in Figure 8, it is assumed that the number of rounds released at the target directly affects the Pk. When the number of rounds released exceeds 50, the Pk will be capped at the vendors's specified SSKP. As in the SAM system, the Pk contour could be substituted with more accurate ones when they become available.

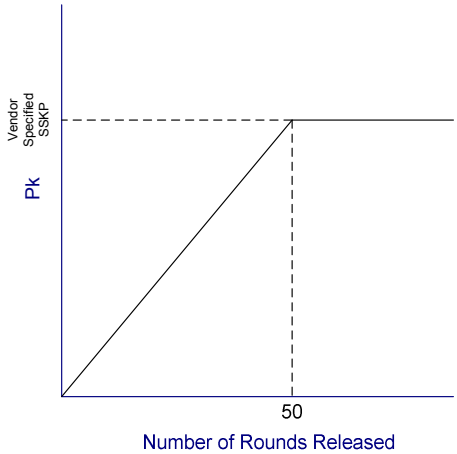

Figure 8: Pk Contour of Anti-Aircraft Gun

Modeling the Strike Aircraft

Strike aircraft are simply modelled using Uniform Linear Movers which are controlled by Path Mover Managers. In the current DES model, the aircraft are able to vary their speed at each waypoint. However, acceleration and deceleration are not modelled currently.

3.2 Agent-Based Strike Plan Generation

Approaches for Planning and Control

In the earlier section, the agent architecture model described a hierarchical approach for making decisions. The central tasking agent decides how to best conduct a strike into the strike area, the evaluation agents plan the actual movement of strike aircraft with this information, and the executing agents execute the movement plan accordingly. In the following paragraph, the algorithms that were used by these agents will be discussed.

When determining how to best approach the strike area, a proposed technique similar to position evaluation function described in KillZone AI [10] was used. Position evaluation functions are well known in computer chess, where the (Artificial Intelligence) AI generates possible board positions and evaluates these board positions to select the strongest series of moves.

In this technique, there is a need to find out the best approach vectors to the strike area. The various factors that have to be considered include the air defence coverage, overlapped air defence coverage, exposure time to the air defences, exposed distance to air defence before reaching the strike area and the speed of the aircraft. The number of approach vectors also depends on the various types of tactics to use. To generate a suitable approach vector, the strike area has to be determined first. From the strike area, straight lines are generated for every 10 degrees. The eventual result will look like a spokes of a wheel as shown in Figure 9. For every spoke line, the total expose distance to the air defence was determined. The normalized value will be used as a score. The exposed distance calculation is based on the line-intersection of the air defence coverage. The total exposed distance of a single spoke line is obtained by adding the exposed distance that a spoke line intersected with individual air defence coverage:

$$\text{Expose distance of a single spoke} = \sum \text{Distance of spoke intersecting air defense coverage}$$

The exposed distance is normalized by the following formula:

$$\text{Normalized Exposed distance} = \frac{\text{Exposed distance of a single spoke}}{\sum \text{Exposed distance of spokes}}$$

Secondly, there is also a need to determine the exposure time to the air defence coverage for each of the spoke lines. This is due to the fact that the exposed distance alone is not good enough to determine the best approach for overlapped air defence coverage as this also depends on the exposure time over this composite air defences

coverage. The strike aircraft is required to cross this exposed distance as quickly as possible. Therefore, the exposure time calculation is based on the following formula:

$$\text{Exposure Time} = \frac{\text{Exposed distance of the composite air defenses coverage}}{\text{Speed of the strike aircraft}}$$

The exposed time over the composite air defences coverage has to be normalized and this is obtained by:

$$\text{Normalized Exposure Time} = \frac{\text{Exposure time of a single spoke}}{\sum \text{Exposure time of the spokes}}$$

The normalization of these 2 scores is to create a unified metric for selecting the best approach. These 2 normalized scores are then added together to represent the weight of the approach vector and the best approach will be the vector that has the highest scores.

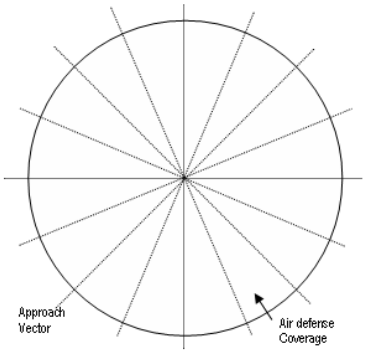

Figure 9: Position Evaluation for Approach Vectors

In the approach for movement planning, the cell decomposition approach is chosen as describe in Movement Behavior for Soldier Agents on a Virtual Battlefield [11]. The idea is to represent free space and air defence coverage as a grid of small uniform cells that are square in shape as shown in Figure 10. Although the cell cannot represent the shape of the air defence coverage exactly, it is possible to vary the size of the cell to either increase or decrease the details of the representation. The size of the cell is always inversely proportional to the detailed level of the representation. The movement planning on the grid is by searching through the cells.

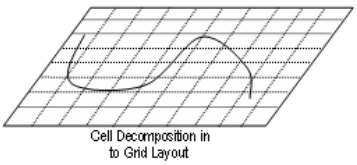

Figure 10. Cell Decomposition Search space

Once the area of operation is represented in a grid, which can also be known as the threat map, the A* algorithm is typically used to control the search from start to destination, with the straight line distance to the destination as a heuristic function. A simple A* search is not used; instead a technique described as Tactical Path-Finding with A* [12] was used. This search algorithm still follows the generic function of the A* search algorithm where G_x is the cost function, H_x is the heuristic function and F_x is the sum of the cost functions and the heuristic given by:

$$F_x = G_x + H_x$$

In this technique, additional considerations are factored in the cost function, G_x, of the algorithm which include the exposure to air defences. The exposure cost is based on the type of air defence unit covering the area. For overlapping coverage of two or more air defence units, the total exposure cost is computed by adding the exposure cost of the overlapping air defence unit together. The heuristic, an estimate of the minimum distance from start to end, uses the Euclidean distance function which is an application of Pythagorean Theorem between start point, (S_x, S_y) and end point (E_x, E_y) is given as:

$$\text{Euclidean distance} = \sqrt{(S_x - E_x)^2 + (S_y - E_y)^2}$$

Hence, the costing structure of the air defence type that is deemed to be suitable is as follow:

Cost Structure	
SAM	10
GUN	5
MOVEMENT COST	1

Table 1: Cost Structure for path planning algorithm

Lastly, behavior mechanisms for responding to the state changes in the environment are added to the individual agent. This behavior mechanism includes a set of actions and a steering behavior. The typical states convey back to the agent from the environment with reference to the air defence system are "Lock-on", "Lock-off", "Incoming missile" and "Gun firing". The current actions implemented, which can be taken by the agent include evasive process and strike process in response to the state of the environment. Each of the agents keeps track of its own current action and process an action if the current action is not suitable of the change of state received from simulation environment. The current action of the agent can be the following "Lock-on Action", "Lock-lost Action" and "Evasive Action" actions.

The process action of “Evasive Action” is undertaken when the agent received a state change message from “Lock-on” state to “Incoming missile” state from the simulation environment and the action of the agent is not ‘Evasive Action”. The agent will initiate an evasive process when its action is “Evasive Action”, and will generate a series of waypoints out of the strike area based on its current heading, and the direction away from the target area, the waypoints are then sent to the simulator which will reflect the agent steering behavior.

The process action of “Lock-lost Action” occurs when the agent received a state change message from “Incoming missile” state to “Lock-off” state from the simulation environment and the agent action is not “Lock-on Action” and “Evasive Action”. The agent will initiate a strike process when its action is “Lock-lost Action” and will generate waypoints back to the target area. The waypoints will be sent to the simulator which will reflect the agent steering behavior

4. Experiment and Results

A scenario was crafted as a basis for the conduct of an experiment. The scenario assumed that the attacker has good intelligence on the locations and types of weapon systems deployed, and has planned an attack route using the agent-based strike plan generator that has selected the safest route of approach. A graphical representation of the scenario is shown in Figure 11. The question posted is: “How sensitive is our current attack plan to the variation in the weapon systems?” Such variations are caused by imperfect intelligence which is common in any intelligence gathering process. The Measure of Effectiveness (MOE) is based on the number of leakages. Number of leakages refers to the number of enemy aircraft that are successful in reaching the Bomb Release Line (BRL). In the simulation, it is assumed that any aircraft that reaches the BRL will be able to launch its bomb without fail.

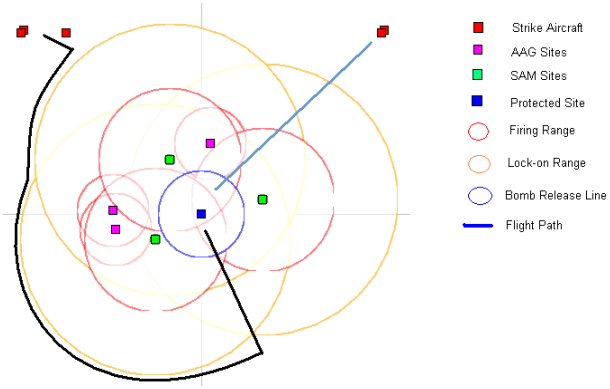

Figure 11: The Attacker's Scenario

Based on the current model, there are a total of 15 potential main effects that could affect the effectiveness of the air defence plan. The 15 potential main effects and the range over which each effect could be varied are shown in 02.

Potential Effects	Min	Max	Units
SAM Reaction Time	5.0	60.0	s
SAM Loading Time	120.0	360.0	s
Missile Per Launcher	2	8	
SAM Min Range	5.0	10.0	'00 m
SAM Max Range	50.0	150.0	'00 m
SAM Max Speed	5.5	8.0	'00 m/s
SAM Engagement Time	3.0	10.0	s
SAM SSKP	55	90	%
Gun Reaction Time	4.0	8.0	s
Gun Loading Time	30.0	60.0	s
Gun Min Range	1.0	5.0	'00 m
Gun Max Range	30.0	40.0	'00 m
Gun Rate of Fire	600	1200	
Gun Magazine Size	200	400	
Gun SSKP	75	85	%

Table 2: Potential Main Effects of the Model

To conduct an experiment with 15 potential effects which has continuous values for most of the effects will be very time consuming using traditional experimental design. For instance, using a conservative estimation of full factorial 2 level design, there will be 2^15 = 32768 design points. If 50 runs are conducted for each design point, a total of 32768 x 50 = 1,638,400 runs will be required.

This experimental design has adopted the Nearly Orthogonal Latin Hypercube (NOLH) by Cioppa & Lucas [13]. Using the NOLH spreadsheet [14], a total of 65 design points were identified, which has drastically reduced the number of runs required from 1,638,400 runs to 3250 runs (65 x 50).

The values in the design points proposed by the NOLH spreadsheet were fed into the model. For each design point, the model was run 50 times to get an unbiased mean value for the number of leakages.

The data collected was fit into a partition tree as shown in 02. It can be observed that the mean number of leakages is quite close to 4 with small variances.

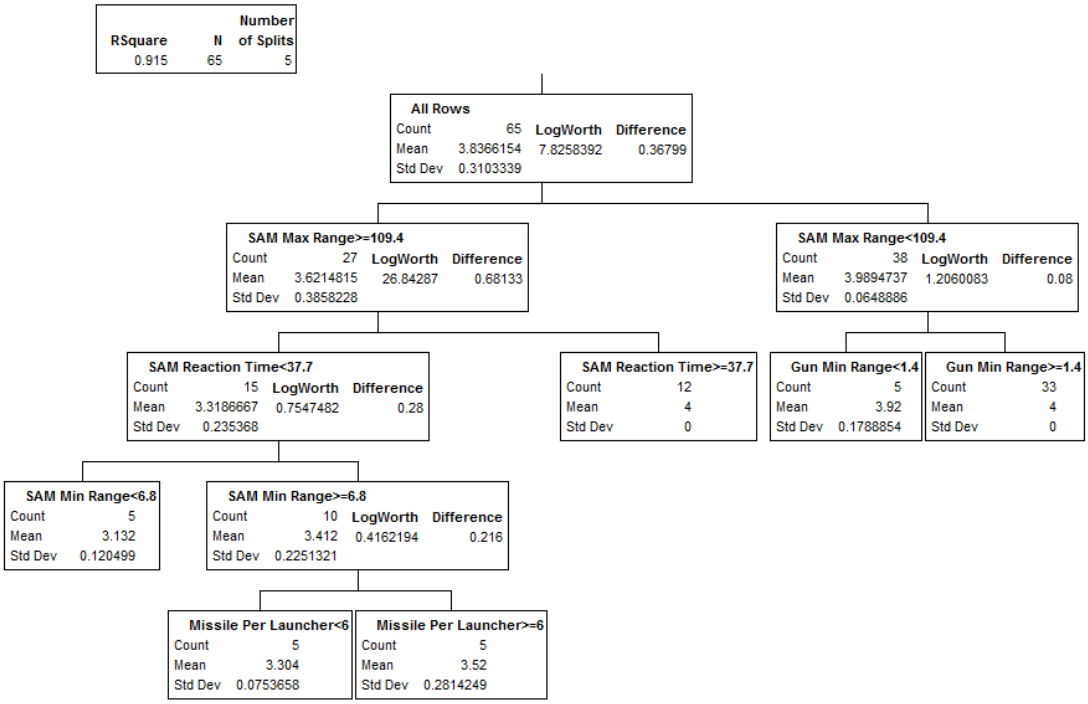

Figure 12: Partition Tree for Attacker's Scenario.

There are two branches that would guarantee with certainty (standard deviation of 0) that 4 aircrafts will complete their mission. For instance, if the attacker feels his intelligence knows the SAM range and SAM reaction time accurately, and it says those are below 109 (10.9km) and above 37.7 (37.7s) respectively, then the attacker should feel very confident that four of his aircrafts will definitely breach the MRL.

A worst-case scenario from the attacker's point of view, is the lower left branch with a mean of 3.132 (on average, close to two aircraft is lost). However, since the variability is not too big (standard deviation of 0.12), the attacker knows that even under this worst case scenario there is a high probability of at least three out of five aircraft would achieve their goal.

If the attacker's only consideration is to get at least one plane to the target, the data analysis clearly shows that this objective is highly achievable, although the attacker might lose some aircraft while doing so.

The regression model of the data collected is shown in 0. The regression model is described by the following equation:

$$\hat{Y} = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + b_4X_4 + b_{12}X_1X_2 + b_{13}X_1X_3 + \ldots$$

where:

$\hat{Y}$	– Number of breaches of the BRL (dependent variable)
b_0	– Intercept
$b_1, b_2, b_3, b_4 \ldots$	– Coefficient for the independent variables
X_1	– SAM Reaction Time (independent variable)
X_2	– SAM Max Range (independent variable)
X_3	– SAM SSKP (independent variable)
X_4	– SAM Max Speed (independent variable)

The main effects showed up in the regression model are inline with that of the partition model, confirming the importance of these key SAM parameters when planning for airstrike operations.

Summary of Fit

RSquare	0.847953
RSquare Adj	0.801408
Root Mean Square Error	0.138296
Mean of Response	3.836615
Observations (or Sum Wgts)	65

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	3.9042878	0.23991	16.27	<.0001*
SAM Reaction Time	0.0108063	0.001233	8.77	<.0001*
SAM Max Range	-0.00523	0.000642	-8.14	<.0001*
(SAM Reaction Time-32.5015)*(SAM Max Range-100.006)	0.0003027	3.732e-5	8.11	<.0001*
(SAM Reaction Time-32.5015)*(SAM Max Speed-6.75077)*(SAM SSKP-72.5077)	-0.000286	0.000123	-2.31	0.0250*
(SAM Max Range-100.006)*(SAM Max Speed-6.75077)	0.0020898	0.000953	2.19	0.0332*

Table 3: Regression Model of Attacker's Scenario

5. Conclusions and Recommendations

This paper has described research that has demonstrated the ability of agent-generated air strike plans to exploit weaknesses in air defence plans which makes it a valuable tool for foreseeing the action, reaction and counteraction dynamics between the attack and defence plans. In addition, the experiment has also shown potential ways of using both the DES engine and plans generator in answering operations research questions. It is hoped that the tools developed in this research could be further refined to assist air defence planners in creating consistent and highly robust defence plans.

The LEGO framework adopted in the design of the DES engine allows individual components to be further refined with little or no impact to other components in the system. The sensors used in this research are mainly constant time based or simple cookie-cutter based. While they have served well to facilitate the rapid construction of a proof-of-concept (POC) model for this research, the sensors should be refined to reflect more realistic sensor characteristics in an actual air defence setup. Potential enhancements include, modeling sensor footprint of irregular shapes and modeling sensor detection/undetection time using the glimpse model. With the framework, the sensors could be replaced with minimal effort.

To keep this POC model simple, the altitude of aircraft and terrain were not considered in the model. While modeling altitude as a continuous variable is more realistic, the introduction of a third dimension is likely to make the model much more complex. Depending on situation, it might be worthwhile to consider abstracting the altitude into discrete height intervals instead of a continuous variable to reduce the complexity of the model. In addition to altitude, acceleration was not considered explicitly in the model. Before the model is extended, one might want to consider if acceleration is necessary for a low resolution model. It is always a good practice to keep the model simple.

For simplicity, the SAM sensors in the current model, acquire a lock on incoming aircraft based on first-come-first-served principle. The sequence of aircraft entering the lock-on range determines the order of how the aircraft are being locked. Although simple, this behavior might not represent air defence doctrines accurately. The model could be enhanced to assess the threat level of incoming aircraft before deciding to lock-on to it or to switch its lock to another more threatening aircraft. For example, if an aircraft is in the lock-on range but not heading towards the BRL, while another aircraft is heading towards the BRL at a high velocity, the sensor might want to lock-on to the later aircraft instead of the first, even though it is in the lock-on range.

For the agent-based model, the path finding algorithm can be improved further by including the additional cost factor such as duration of exposure to air defences which is not currently taken into consideration. In addition, the cost of using A* algorithm can be very expensive as the area of operation for the air formation is expanded. Therefore, a dynamic area of operation should be used for each air formation; this will allow each formation to focus on its own area of operation. Hierarchical path-finding can also be use to reduce the search complexity of the path finding, this is where the entire map of the area of operation is abstracted in several levels and into linked local clusters, where, at the global level, path finding through clusters is traversed in a single big step and the search path is further refine at the cluster level of the abstracted map, which has more details, as it approach its goal.

The agent application can also take in terrain information such as Digital Terrain Elevation Data (DTED) map or vegetation information in form of Shape files for its path finding algorithm. This will make the path planning more viable for use in modeling a real-world environment.

For the individual agent aircraft behavior model, the current implementation only caters to a few actions that the agent can do. Improvement can be made by expanding more actions to allow more dynamic agent behavior. Furthermore, sophisticated

behavior can be implemented to consider information of current position, air-defence site position or even additional incoming threats by using techniques such as a neural network to learn from past actions or Bayesian network to perform inference.

6. References

[1] L. Andrew, "Hierarchical AI," http://www-cs-students.stanford.edu/~amitp/Articles/HierarchalAI.html, September 19, 2008.

[2] E. Mark, "The Clash of Civilizations," http://clash.apolyton.net/models/Model-AI.shtml, September 18, 2008.

[3] A. Buss, and P. Sanchez, "Building Complex Models with LEGOS," in *Proceedings of the 2002 Winter Simulation Conference,* 2002, pp.732-737.

[4] A. Buss, "Discrete event programming with Simkit," *Simulation News Europe,* (32/33), pp. 15-26, 2001.

[5] A. Buss, and P. Sanchez, "Simple Movement and Detection in Discrete Event Simulation," in *Proceedings of the 2005 Winter Simulation Conference*, 2005, pp. 992-1000.

[6] A. Buss, "Component-Based Simulation Modeling," *Proceedings of the 2000 Winter Simulation Conference,* 2000, pp. 964-971.

[7] A. Buss, and D. Ahner, "Dynamic Allocation of Fires and Sensors (DAFS): A Low-Resolution Simulation for Rapid Modeling," in *Proceedings of the 2006 Winter Simulation Conference*, 2006, pp. 1357-1364.

[8] M. Makoto, and N. Takuji, "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 8, (1), pp. 3-30, 1998.

[9] "US Army Field Manual 44-43: Bradley Stinger Fighting Vehicle Platoon and Squad Operations," *http://www.globalsecurity.org/military/library/policy/army/fm/44-43/index.html*, November 2, 2008.

[10] R. Straatman, W. Sterren, & A. Beij. (2006), "Killzone's AI: dynamic procedural combat tactics," http://www.cgf-ai.com/docs/straatman_remco_killzone_ai.pdf, September 18, 2008.

[11] D. Reece, "Movement Behavior for Soldier Agents on a Virtual Battlefield," *Massachusetts of Technology Presence,* Vol.12, (4), pp. 387-410, 2003.

[12] W. Sterren, "Tactical Path-Finding with A*," in *Game Programming Gem 3,* Boston: Charles River Media, 2002, pp. 294-306.

[13] T. Cioppa, and T. Lucas, "Efficient Nearly Orthogonal And Space-Filling Experimental Designs For High-Dimensional Complex Models," PhD's Dissertation, United States Naval Postgraduate School, Monterey, CA, September 2002.

[14] S. Sanchez, "NOLHdesigns spreadsheet," *http://harvest.nps.edu/*, September 18, 2008.

7. Biographies

Mr. Andy Ong is a programme manager in Defence Science & Technology Agency, managing the development of command and control systems for the Singapore Armed Forces. Mr Andy has received the M. Sc in Modeling, Virtual Environment Systems from Naval Postgraduate School, United States. He has previously received the M.Sc. in Defence Technology Systems and M.Tech in Knowledge Engineering from the National University of Singapore and the B.Eng. in Electrical & Electronic from the Nanyang Technological University, Singapore. He has more than 10 years of experience in the development and management of Army, Joint and Navy Command and Control (C2) information systems.

Mr. Andrew Wong is a project manager in Defence Science & Technology Agency, where he manages experimentation projects in the area of command and control systems for the Singapore Armed Forces. Mr Wong has received the M.Sc in Modeling, Virtual Environment Systems and M.Sc. in Defence Technology Systems from the Naval Postgraduate School, United States and National University of Singapore respectively. He also received his B.Eng in Electrical & Electronic from University of Leicester, United Kingdom. He has more than 10 years of experience in the development and management of Army and Navy Command and Control (C2) information systems.

Christian J. Darken, Ph.D., is an Associate Professor of Computer Science at the Naval Postgraduate School, where he also collaborates intensively with the MOVES Institute. Previously he was Project Manager of the Decision Support Systems project and Senior Member of Technical Staff at Siemens Corporate Research in Princeton, NJ, where he was variously associated with the Learning Systems, Adaptive Information and Signal Processing, and Software Engineering Departments. He was also a programmer of one of the first commercial first-person perspective massively-multiplayer games. He received his Ph.D. in Electrical Engineering from Yale University in 1993, and previously received the M.S. and M. Phil. in Physics from the same institution.

ARNOLD BUSS is a Research Associate Professor in the MOVES Institute at the Naval Postgraduate School. He received a BA in Psychology from Rutgers University, an MS in Systems and Industrial Engineering from the University of Arizona, and a Ph.D. in Operations Research from Cornell University. His research interests include simulation modeling and object-oriented software design. He is a member of INFORMS and MORS. (Address: MOVES Institute, 700 Dyer Road, Naval Postgraduate School, Monterey, CA, URL: http://diana.nps.edu/~ahbuss, E-mail: abuss@nps.edu).