# Predicting Enemy Squad Positions Based on Environmental Data, Unit Objectives, Tactics and Observed Entities

Kenneth Maroon[1(✉)] and Christian Darken[1]

[1] MOVES Institute, Naval Postgraduate School, Monterey, CA USA
{kjmaroon1, cjdarken}@nps.edu

**Abstract.** The Intelligence Preparation of the Battlespace (IPB) allows company commanders to grasp key aspects of the environment they are operating in, as well as the elements and objectives of the adversary they are facing. This is accomplished through analysis of geospatial intelligence (GEOINT), open source data, and other intelligence streams to produce situation estimates, known adversary tactics, and enemy courses of action (COA). The products developed during the IPB can also be used in combat simulations to improve artificial intelligence for automated planning and improved wargaming. This paper describes a process to represent IPB results as navigation mesh annotations and position evaluation functions. These can then be used for scoring opposing force formations, based on objectives, tactics, and terrain data. Since it is reasonable to assume that an enemy force will position its units to have the greatest ability to accomplish its objectives, formations that maximize our scoring function act as an educated prediction of enemy unit positions. These predictions can then support more robust automated planning and improved combat modeling. We illustrate this method through the use of WOMBAT XXI, a research prototype combat model.

**Keywords:** Military Planning, GEOINT, Tactical Movement, Combat Models.

## 1 Introduction

Modeling how human commanders predict opponent positions is a difficult task for constructive combat simulations. Humans use their experience, understanding of the local terrain, environmental conditions, and enemy intelligence estimates to make informed decisions about where they believe their opponents are. Simulation designers attempting to have AI-controlled bots make similar predictions will often use scripted behaviors or give bots extraneous data not available to human participants [1]. This could be considered "cheating," and involves such things as perfect knowledge of enemy positions or predefined tactical advantage points. We propose a method to have bots use the same

environmental data and intelligence products available to humans to make their predictions. In the following sections, we will identify previous research into opponent position detection used to develop our predictor, outline the Intelligence Preparation of the Battlespace (IPB) and its products that we use for positional evaluation, demonstrate the use of our predictor in a research prototype combat model called WOMBAT XXI, and close with possible improvements and future work.

## 2 Related Work

There is a rich body of work regarding realistic opponent position tracking that we build our predictor from. Tozour [2] and Isla [3] introduce space-based methods for opponent tracking using occupancy maps. These methods represent possible opponent positions as the probability that a node is occupied on a grid or navigation graph. Probability is diffused between nodes over time and updated as nodes become visible to tracking entities. Alternatively, Bererton [4] proposed the use of particle filters to reflect possible positions as a finite set of weighted samples that move randomly in the search space (particles). Darken and Anderegg [5] further refined this method by using defined motion models called "simulacra" for their particles, producing predictions that can reflect different behaviors expected in opponents.

Additionally, we looked at ways of bringing tactical combat experience into simulations. Straatman et. al. [1] were able to produce dynamic tactical behavior in their bots through the use of waypoint annotations and position evaluation. In their approach waypoints are annotated based on assigned goals and operating areas. Additionally, dynamic information such as cover, lines-of-fire, and spacing related to other friendly entities are calculated for each waypoint. Position evaluation functions are then used to determine scoring for each waypoint and entities pick ones with the highest score. We also looked at research on fireteam movement conducted by Darken et al. [6] for use in our prediction models. Hladky and Bulitko [7] also tested hidden semi-Markov models and particle filter predictors in the first-person shooter, *Counter-Strike: Source*.

Our predictor builds on previous work through the use of navigation meshes that are annotated based on known opponent tactics and intelligence. We use position evaluation as a scoring function for possi-

ble opponent positions. Our predictor then searches for formations that maximize this scoring function.

## 3 Problem Formation

For this paper we will use the term "entity" to refer to a single simulated combat element. Each entity is part of a unit. Units are made of one or more entities and can also contain sub-units. One entity in each unit can be designated as the unit leader. The set containing the positions of each entity in a unit (including the unit's sub-units) is the unit's formation.

Our prediction problem can be broken up into two parts. First, we need to create a navigation graph of the operational area and a scoring function for each node on the graph. The scoring function needs to reflect expected adversary tactics and objectives, as well as environmental considerations. Each entity's position can be evaluated by the scoring function, and a unit's formation score is the sum of all of its entity evaluations. The second part of the prediction problem is identifying entity positions that maximize their unit formation score. While searching for this set of entity positions is not trivial, it can be accomplished in many ways. Techniques such as hill climbing, simulated annealing, and genetic algorithms can be used to find high scoring formations. However, we are not guaranteed to find the absolute best set of positions. At best, we can use these algorithms to determine sets of positions that are local maxima in the search space.

In the next sections, we will identify the key products for our scoring function and present a combat simulation environment to test our predictor.

### 3.1 Intelligence Preparation of the Battlespace

Intelligence Preparation of the Battlespace (IPB) is "the systematic continuous process of analyzing the threat and environment in a specific geographic area" [8]. This evaluation includes analyzing available GEOINT and open source data, as well as studying threat patterns, doctrine and known standard operating procedures (SOP). This analysis produces a graphic representation of enemy tactics called an Adversary Template [9]. Enemy Course of Actions (ECOA) are another integral product of the IPB, as they describe expected enemy actions and objec-

tives. Multiple ECOAs are developed to reflect what the enemy may be doing at the start of the operation and during key events. These IPB products are the main way real world intelligence and data are represented in our prediction model. The IPB provides us with the geographical data to build the navigation graph on the operational area. We can use adversary templates as a means to evaluate unit geometry and annotate graph nodes consistent with enemy objectives outlined in ECOAs.

### 3.2 Wombat XXI

WOMBAT XXI (WXXI) is a combat simulation built by LtCol Byron Harder, USMC [10] to support his development of an automated fire support planner. Harder based WXXI on the Combined Arms Analysis Tool for the 21st Century (COMBATXXI) simulation used by the U.S. Army and Marine Corps. WXXI is built on the Unity3D game engine. While we did not use the automated planner built by Harder, the unit templates, terrain-meshes, and visibility calculations in WXXI were vital to the development of our predictor.

## 4 Building the Formation Scoring Function

The first step in developing our predictor is to generate a model of the terrain of our operational area. In WXXI, actual elevation data is converted to a Unity terrain object. The terrain object represents the real-world data as a triangle-mesh. This terrain object is then processed to create a navigation graph using the Astar Pathfinding Project [11]. The navigation graph uses the triangles of the terrain object as its nodes. For our testing we used a terrain object created by Harder based on a 2 $km^2$ area from Cayucos Creek, CA. For this terrain, there were 8192 total nodes in the navigation graph [10].

### 4.1 Annotate Navigation Graph Nodes

Once the Navigation Graph nodes have been created, they can be annotated according to the outputs of the IPB. Terrain types, weather effects, and visibility can be precalculated for each node to assist in later scoring evaluations. Additionally, ECOAs can be represented by marking nodes that are operationally relevant. These include nodes that must be patrolled, are tactical key points, or are part of a Field of Fire

(FoF). For our testing scenarios, we precalculated the visibility for each node to all other nodes.

In our scenarios, we are also able to assign enemy units preferred targets and designate desired FoF. The targeting assignments reflect the objectives of the ECOAs developed in the IPB. These assignments only affect entities in the current unit, and entities in a unit's sub-units can have different assignments. Entities able to target the node that a preferred target is on, will have their position score increased. Likewise, each node in a FoF that can be targeted by an entity will also increase the entity's position score. We refer to the increase in position score for targeting a particular node as "targetability" for this paper. Targetability for a node is dependent on multiple factors in Wombat XXI. However, for our testing, the closer an entity is to the desired targeted node, the greater the targetability.

Entities will also try to avoid being targeted themselves. If an entity's node can be targeted by the opposing force, their position score will be decreased by the opponent force's targetability of that node.

### 4.2 Determine Expected Enemy Formation Geometry

Enemy entities will try to maintain a desired distance to their unit leader as in Darken et. al. [6]. If the entity is outside of the acceptable range of the unit leader, the unit score will be decreased. The further outside the range to the leader, the greater the penalty to the unit formation score. Entities will also try to maintain enough dispersion from other entities to avoid simultaneous attacks. For each entity that is less than a set minimum distance away from another entity, the unit score is decreased. The penalty is increased the closer units are together. Finally, entities will seek to maintain unit cohesion to provide mutual support to other unit members. For each entity outside of a set range to another unit member, unit score is decreased. The greater the distance they are apart, the higher the penalty. The desired ranges and penalties for distance to unit leader, dispersion, and cohesion would be determined using the Adversary Templates generated during the IPB.

### 4.3 Scenario Formation Scoring

For our testing scenario the formation scoring algorithm is:

**Algorithm 1.** The Formation Scoring algorithm

```
calculateTotalScore (unit)
  formationScore = 0
  for all entities in unit
    formationScore += calculateEntityScore (entity)
  for all subunits in unit
    formationScore += calculateTotalScore (subunit)
  return formationScore

calculateEntityScore (entity)
  entityPositionScore = 0
  for all preferredTargets assigned to entity
    entityPositionScore += preferredTarget's node targetability
  for all FoF nodes assigned to entity
    entityPositionScore += node targetability
  if(distance to unit leader > max distance to leader range)
    entityPositionScore -= outside range to leader penalty
  if(distance to other entities > max cohesion range)
    entityPositionScore -= cohesion penalty
  if(distance to other entities < min dispersion range)
    entityPositionScore -= dispersion penalty
  if(entity's node is targetable by opponent)
    entityPositionScore -= targetability penalty
  return entityPositionScore
```

## 5 Finding Maximizing Formations through Hill Climbing

While there are a number of approaches to determining entity positions that maximize unit formation score, we developed a method using simple hill climbing from random positions. We assign entities to a random node on the map within an identified area of operation for their unit. Then, each entity identifies neighbor nodes and evaluates what its position score would be at these nodes. The entity then moves to the node with the highest position score and evaluates its position score at the new neighbor nodes. This continues until all entities have moved to a node with a higher position score than all of its neighbors. Since an entity's position score is dependent on the location of the other entities in the unit, multiple rounds of hill climbing are conducted for the unit. We have found that generally after two rounds entities have found a

maximum position relative to other entities. Additionally, we found that running the first round of hill climbing without a dispersion penalty prevented entities from prematurely blocking another entity's hill climb. When all units have moved to their maximizing locations, we have identified a possible maximizing formation. This process can be repeated multiple times, and the different initial randomization of entity positions will generate different formations at each iteration. We can then either select the highest scoring formation generated, or sample from a distribution of possible formations.

An entity whose position is known from prior observation is simulated by holding its position constant during randomization and hill climbing. The other undetected entities will take positions that maximize unit formation score in relation to the detected entity.

In developing our hill climbing algorithm we ran an experiment, testing it over 1000 runs. The formation scoring function was kept the same, but random starting entity positions were selected for each run. We found that our hill climbing method increased formation scores by 251% on average, while decreasing the relative standard deviation of scores. Full results are shown in Table 1.

**Table 1.** Hill Climbing method performance test results.

| | Starting Score | Ending Score | Change % |
|---|---|---|---|
| $\mu$ | 12.320 | 38.414 | 251 |
| $\sigma$ | 4.483 | 7.757 | 146 |
| $\sigma/\mu$ | 0.364 | 0.202 | -55 |

## 6 Scenarios/Results

It is important to note that while the terrain used in the scenarios is from real-world data, the scale of distance between units is exaggerated for clearer visualization. Additionally, the scoring function and penalty weights were scaled. While they do not reflect exact military doctrine, they do simulate basic tactical positioning.

For these scenarios, the enemy unit consists of a squad leader and two four-entity fire team subunits. Each fire team has a commander entity. The two commander entities will try to stay in range of the squad leader. Fire team members will attempt to stay in range of their respective commanders. Each fire team is also assigned a designated

FoF. Nodes in each FoF are marked with teal and green squares. The targetability of nodes in the FoF is represented with a purple sphere that increases in size as targetability increases. Starting and ending positions for our first scenario are shown in Figures 1 and 2.

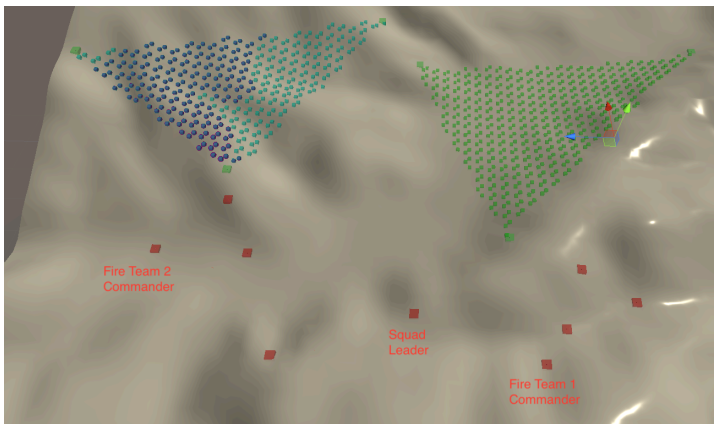


**Fig. 1.** Scenario 1 starting random positions. Fire Team 1's view of its FoF is completely blocked by a ridge. Fire Team 2 starts the scenario with partial visibility into its FoF.

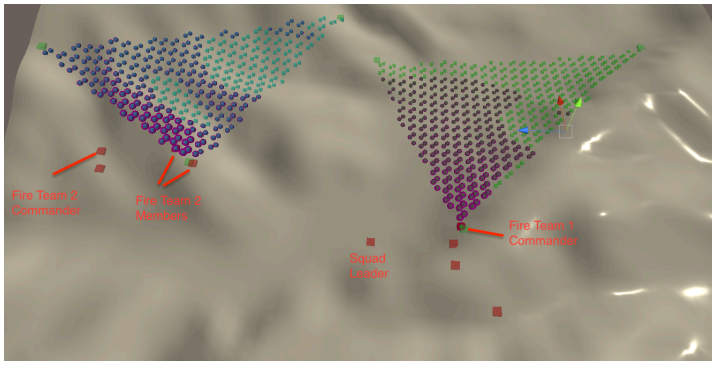


**Fig. 2.** Scenario 1 ending positions after hillclimbing. Fire Team 1 has moved over the ridge blocking their FoF and are now able to target a large portion of it. Fire Team 2 has moved up into a split formation that allows it to better target the undulating terrain of its FoF. Both Fire Teams have tightened unit formations.

Our second scenario demonstrates how our predictor works when an enemy entity is detected. This scenario focuses on a single five-entity unit. One entity in the unit is detected by an opposing blue unit as shown in Figure 3. Nodes that are targetable (simulating visibility for this scenario) by the blue unit are marked with a blue sphere.

Formation scoring for the red unit is the same as in the previous scenario, but red entities will avoid nodes that are targetable by the blue unit. The unit will attempt to maximize targetability FoF nodes and maintain spacing consistent with the previous scenario.

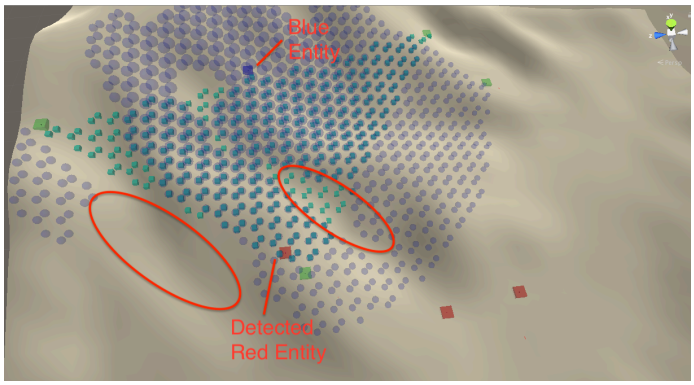


**Fig. 3.** Scenario 2 overview. A red unit is detected by a blue entity. Two other red entities are undetectable by the blue entity. Note how terrain features produce gaps in blue's detection area, these are marked by the red ovals.

For this scenario we conducted 50 total hill-climbing iterations holding the detected entity's position constant for all iterations. The highest scoring formation represents the blue entity's prediction of the red unit's entity positions and is shown in Figure 4.

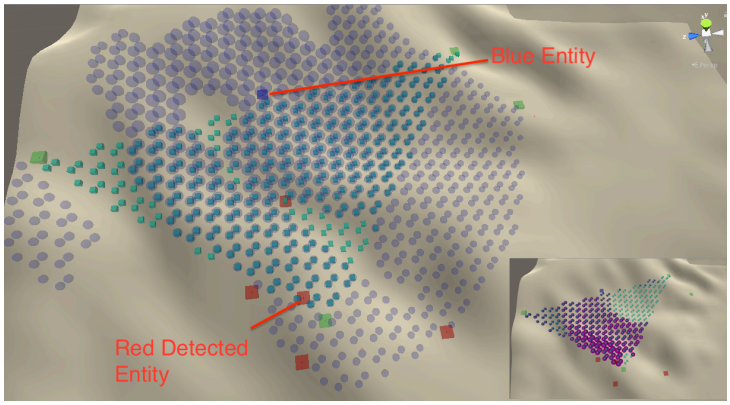


**Fig. 4.** The highest scoring candidate formation. Red units are predicted to be located in the gaps and edges of blue's detection area. This formation maintains red unit spacing and maximizes targetability of the FoF (shown bottom right).

## 7 Conclusions and Future Work

In combat, commanders do not always have an exact position for their enemy threats. They depend an understanding of the tactical environment and enemy objectives to estimate where threats are. Our method simulates this understanding by building formation scoring functions based on the same products developed during the IPB. This allows the AI to make estimations of threat locations using the same data available to human commanders, providing for a more realistic decision-making process. In the future we would like to validate a more complex scoring function against human predictions. We would like to incorporate tactics such as avoiding lines of fire and use our predictor as part of the automated planning functions currently in WXXI.

## Reference


1. Straatman, R., van der Sterren, W., Beij, A. "Killzone's AI: Dynamic Procedural Combat Tactics." In: *Proceedings of the 2005 Game Developers Conference*. San Francisco, CA (2005).
2. Tozour, Paul. "Using a Spatial Database for Runtime Spatial Analysis." *AI Game Programming Wisdom 2*, Charles River Media. (2004).
3. Isla, D. and Blumberg, B. "Object Persistence for Synthetic Creatures." In: *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*. (2002).
4. Bereton, C., "State Estimation for Game AI Using Particle Filters," In *Proceedings of the AAAI Workshop on Challenges in Game AI*. (2004)
5. Darken, C. and Anderegg, B. "Particle Filters and Simulacra for More Realistic Opponent Tracking." *Game AI Programming Wisdom 4*. Charles River Media. (2008).
6. Darken, C., McCue, D., Guerrero, M. "Realistic Fireteam Movement in Urban Environments." In *Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. (2010).
7. Hladky, S. and Bulitko, V. "An Evaluation of Models for Predicting Opponent Positions in First-Person Shooter Video Games." In *Proceedings of Symposium on Computational Intelligence and Games*. Perth, Australia. (2008).
8. U.S. Army. "Intelligence Preparation of the Battlefield/Battlespace." *Army Techniques Publication 2-01.3.* Washington, DC: Headquarters Department of the Army. (2015). https://armypubs.army.mil/epubs/DR_pubs/DR_a/pdf/web/atp2_01x3.pdf
9. U.S. Marine Corps. "Infantry Company Operations." *Marine Corps Warfighting Publication 3-10A.2.* Washington, DC: Headquarters United States Marine Corps. (2014).
10. Harder, B. "Automated Battle Planning for Combat Models with Maneuver and Fire Support." Monterey, California: Naval Postgraduate School. (2017)
11. Granberg, A. The A* Pathfinding Project. Ver. 3.8.2. Unity software asset. (2016). http://arongranberg.com/astar/.