A TERRAIN REASONING ALGORITHM FOR DEFENDING A FIRE ZONE

Mikel D. PETTY, Robert W. FRANCESCHINI and Amar MUKHERJEE

1. Background

This section provides brief background information on distributed simulation, computer generated forces, and ModSAF, which set the context for the research described in this paper.

1.1. Distributed simulation

Distributed simulation is an approach to building large-scale simulation models from a set of independent simulator nodes communicating via a network. The simulator nodes each independently simulate the activities of one or more entities in the simulated world and report the attributes (e.g., location and velocity) and actions (e.g., weapons firing) of those entities via the network to the other simulator nodes using a pre-defined communications protocol. In a typical distributed simulation, the simulated entities coexist in a common simulated environment and can interact with each other; their interactions are realized via the exchange of messages in the protocol.

One example of a military distributed simulation is the U.S. Army's Close Combat Tactical Trainer (CCTT). In CCTT, which is a real-time system, the simulator nodes typically represent single vehicles, such as tanks or aircraft. CCTT simulators are substantial devices consisting of a simulation computer, a computer image generator, and a physical simulation of the interior of the vehicle being simulated. The human crew of the simulator maneuver their simulated vehicle in the simulated world and interact with (e.g., fire their weapons at) other vehicles.

During the execution of a simulation, each simulator's simulation computer tracks the position of the vehicle it is simulating, as well as the other entities in the simulation, on a terrain database common to the system. The simulator broadcasts messages

reporting the location of the vehicle it is simulating, which are used by the other simulators to track that location, and may send other messages to mediate inter-entity interactions. The specific network protocol used by the CCTT simulators is a version of the Distributed Interactive Simulation (DIS) IEEE-1278 standard.

1.2. Computer generated forces

Distributed simulations often include a special type of simulator node known as a Computer Generated Forces (CGF) system. A CGF system usually generates multiple battlefield entities (such as tanks, aircraft, and infantry) using computer algorithms rather than human crew to generate and control the real-time actions of those entities. The behavior algorithms are designed to react to the simulation situation and to generate tactically and doctrinally correct actions. CGF systems are connected into the distributed simulation in the same manner as the other nodes and send and receive the requisite network messages for their CGF entities. CGF systems are often used to provide hostile forces and supplementary friendly forces in battlefield training simulations that involve human trainees. CGF systems may also be used for analytical simulations, where they often populate the battlefield completely as needed for the analytical experiment.[1]

1.3. ModSAF

One CGF system of particular interest in distributed simulation is ModSAF (Modular Semi-Automated Forces). ModSAF is the most widely distributed CGF system and is used for many purposes, including training, analytical experiments, and research into CGF algorithms. (However, ModSAF is not used in CCTT, which has its own CGF system). It is a large system, consisting of approximately 1.5 million lines of C code. It includes user interface, network interface, physical modeling, low (entity) level behavior generation, and high (military unit) level behavior generation capabilities.[2]

ModSAF represents the terrain of the battlefield using a data format known as CTDB (Compact Terrain Database Base). In the CTDB format, the terrain surface is primarily represented by elevation posts, which are elevation z values for the terrain at a regular square grid of x, y locations. Triangles are induced from the elevation posts and these triangles make up the terrain surface. Terrain features, such as roads, bridges, buildings, individual trees, tree lines, and tree canopies are also represented in the CTDB format.[3]

2. Introduction to the research

This section defines the problem, briefly mentions some related research, and identifies existing algorithms to solve the problem.

2.1. Problem definition

When a military unit (platoon, company, or battalion) takes up a defensive position, finding good deployment locations for the individual entities of that unit is an important terrain reasoning task. This is true for both real and simulated military units. A military unit that is being generated by a CGF system will consist of some number of entities (typically 3 to 50). A useful CGF capability is to be able to order a unit to take up a defensive position and have a terrain reasoning algorithm determine deployment locations for each of the individual entities of that unit. The CGF system then moves the entities to the selected locations. Such an algorithm in fact exists in the ModSAF CGF system, in a behavior called *Hasty Occupy Position*. However, the Hasty Occupy Position algorithm does not consider observation of the area to be defended. We sought improved performance with an algorithm that employs geometric terrain reasoning to consider observation.

We will refer to the military unit for which deployment locations must be chosen as the *unit*. Its component units at the next lower echelon will be called *subunits*. In this work, we confined our attention to company units and platoon subunits.

The process of selecting specific locations for the individual entities of the unit must begin with certain given information. The *battle position* for a unit is a specification of the region in which its entities must be located; it is given as a chain of segments located on the terrain. The length of the battle position, in both military tactics and in simulation, depends on the size of the unit and the tactical situation. A typical battle position for a company is 1500 meters long.[4] The entities of the company deploy within some radius of the battle position segments, where that radius is another piece of given information. We will refer to points within the given radius of the battle position chain of segments as *in* or *within* the battle position. Also given is the *engagement area*, which is a region of terrain through which an enemy attack is expected. (The expected direction of attack is determined by reconnaissance and military intelligence and the engagement area is determined from that direction by command planning at echelons above the unit. For this problem, the engagement area is simply given.) An engagement area for a company is typically 1000 to 1500 meters wide and 2000 to 3000 meters deep. The entities of the unit should be located where they can observe as much of the engagement area as possible in order to employ direct fire against attacking entities. A *location* is a specific point on a terrain database. *Target reference points* (TRPs) are locations that are used to delimit and partition the engagement area for the subunits of the unit.

Cover is protection from enemy fire and observation by intervening terrain surface and *concealment* is protection by terrain features. Cover or concealment may only partially protect an entity. For example, a tank in hull defilade might be located so

that its hull is covered by a ridge crest but its turret is not, allowing it some protection while still allowing it to use its weapons, a desirable situation.

To select deployment locations for the entities of a unit three types of constraints must be considered. First, the geometry of the terrain surface and the terrain features and their effect on intervisibility obviously constrain how much of the engagement area can be observed from any location in the battle position. The goal is to observe the engagement area; so that forces "... must see the whole battlefield ...".[5]

Second, cover and concealment in the battle position must be considered, as the defending entities should not be unduly exposed to enemy fire. Army doctrine for the defense states that "... fighting forces must be covered and concealed ..." and "Tanks engage enemy [forces] from covered and concealed positions."[6] For the fire zone defense problem, the entities should be deployed in cover and concealment locations where the entity is partially but not fully covered or concealed as the entity must be able to use its weapons to defend the engagement area.

Finally, military doctrine imposes constraints on where the entities may be located. Two are considered here: first, a unit battle position should be partitioned into battle positions for the subunits. According to 1993 U.S.Army Field Manual,[7] in defensive operations "Commanders position their forces in platoon, company, or battalion battle positions ...". Second, to retain tactical cohesiveness the entities of a subunit should be kept together rather than intermingled.[8]

With all these concepts in hand we can state the *fire zone defense* problem:

> Given a military unit, a terrain database, and a battle position, engagement area, and target reference located on that database, find deployment locations within the battle position for the entities of the unit that simultaneously satisfy the intervisibility, cover and concealment, and doctrinal constraints and also maximize observation of the engagement area.

To evaluate solutions to the problem as stated, we need a performance metric that is both militarily meaningful and quantitatively comparable. To define such a metric we turn to military sources, which make the goal of selecting deployment locations very clear. "Weapons are most effective when sited to cover an expanse of terrain over which the attack must come."[9] As stated in the U.S. Marine Corps doctrine, "Weapons must be located where their effects will be greatest and dead space is minimized."[10] ("Dead space" is terrain that can not be observed or fired on from a battle position because of intervening terrain).

In the context of automated terrain reasoning, intervisibility measures for points are suggested by Keirsey and co-workers.[11] The first, *general visibility*, is the cumulative area of terrain visible from a point. The second, *specific visibility*, is the percentage

of a specified area visible from that point. The second measure is most relevant in this problem, where the specified area is the engagement area. Therefore, based on the referenced sources, we define *location observation* as the portion of the engagement area visible from a location and *cumulative observation* as the sum of the location observation values for all the deployment locations selected for the entities of a unit. Algorithms to solve the fire zone defense problem will be compared using the cumulative observation metric.

2.2. Related research

A 2-dimensional algorithm that positions a set of sensors within a set of territories so as to optimally cover the given territories with the envelopes of the sensors is presented by.[12] A symbolic object-oriented method for finding positions for a unit's component subunits (e.g. the platoons of a company); is described by Hieb, Hille and coworkers [13]; the method considers the terrain as a set of symbolic conceptual objects (e.g. hill-863 or avenue-of-approach-2) and consequently does not produce precise deployment locations. In work reported by Rajput and Tu, [14] the ModSAF cover and concealment finding algorithm was enhanced to find assault positions and support-by-fire positions, which are terrain areas used by units when attacking an enemy position.

2.3. Existing algorithm

ModSAF version 2.1 was used for this experiment, as it was the version available to the authors when this work was performed. The ModSAF algorithms described herein, and in general all references to ModSAF in this paper are with respect to that version. More recent versions of ModSAF exist and may differ from these descriptions.

To evaluate the effectiveness of our new algorithm for the fire zone defense we compared it with ModSAF's Hasty Occupy Position (HOP) algorithm. The HOP algorithm is described by Courtemanche and Ceranowicz [15] and in the ModSAF documentation. In the former it is referred to as "Hasty Occupy Battle Position" but it is "Hasty Occupy Position" in the ModSAF implementation and operator interface; we will use the latter term. The HOP algorithm controls the behavior of ModSAF units at both the company and platoon levels. The company HOP algorithm partitions a company battle position, supplied by the operator as a chain of segments, evenly into platoon battle positions and passes them to the platoon HOP algorithm. The platoon HOP algorithm calculates a battle area, which is a rectangular area centered on the battle position, for each entity in the platoon. It then finds a deployment location for the entity within its battle area using the ModSAF cover and concealment algorithm. If no covered or concealed location is found within the battle

area, the entity is placed at the center of the battle area on the battle position segment; these are the HOP default locations.

The cover and concealment algorithm finds covered and concealed locations on the terrain relative to a given location. It does so by extending lines from the given location through the battle area and examining the terrain profile along those lines, using a spacing parameter.[16]

3. Fire Zone Defense algorithm

Overview. The ModSAF HOP algorithm was replaced with a new algorithm, called Fire Zone Defense (FZD), to solve the stated problem of selecting locations for the entities of a unit so as to effectively defend an engagement area. More precisely, specific parts of the HOP algorithm were replaced. This section describes the FZD algorithm and its integration into ModSAF.

Input. The FZD algorithm is invoked from the ModSAF operator interface as a "Hasty Occupy Position" order. The operator provides via the operator interface the battle position (a chain of segments), left and right TRPs for the engagement area (points), and the engagement area itself (a quadrilateral), and assigns a unit to defend that engagement area. As implemented for this work, the FZD algorithm expects a company unit. The algorithm uses that explicit input and the terrain database to find the deployment locations for the entities of the unit.

Locations. A *location* has already been defined as a specific point on the terrain surface. Certain categories of locations are central in the FZD algorithm. *Sighting locations* are locations within an engagement area that should be visible from the battle position so as to effectively observe the engagement area. Ideally, we might want to sight every location in the engagement area, but because there are an infinite number of such locations, it is impractical to consider all locations in the engagement area when evaluating observation. Sighting locations are a subset of the locations in the engagement area chosen to be representative of the entire engagement area, so that observing the sighting locations is equivalent to observing the entire engagement area, to a good approximation. Sighting locations are calculated by the FZD algorithm using simple geometric terrain analysis. In particular, sighting locations are generated for two types of terrain locations. First, the terrain surface at the elevation posts are taken as sighting locations. Recalling that the CTDB is primarily an elevation post format, and that the elevation posts induce triangles (which are of course planar), the idea is that if the elevation post locations can be observed then the triangles between them can be as well. Hence elevation post locations within the engagement area serve as sighting locations. In the CTDB format the elevations posts are spaced on a 125 meter grid, so the number of elevation posts in a typical

engagement area is reasonable. Second, to account for the effect on intervisibility of terrain features, the vertices of terrain features within the engagement area are also taken as sighting locations. Buildings, tree lines, and canopies are included, but individual trees are not. Van Brackle and coworkers [17] introduced in the reconnaissance planning work the notion of sighting locations representing a terrain area.

Battle position locations are locations evenly spaced along the segments of a battle position. The spacing is a parameter to the algorithm. Battle position locations are used to partition a unit battle position into subunit battle positions, as will be seen later. *Defensive locations* are locations within the battle position that are covered or concealed relative to the engagement area, or a TRP within it. Defensive locations are found using the ModSAF cover and concealment algorithm by passing to it a rectangle and TRP reference point calculated from the battle position. *Deployment locations* are defensive locations actually selected by the FZD algorithm as the optimal locations for the defending unit's entities. Note that the set of deployment locations is a (not necessarily proper) subset of the set of defensive locations.

Observation value. The FZD algorithm depends on the notion of observation value for a location. The observation value for a location is an approximation of the quantity previously defined as location observation, which was how much of the engagement area is visible from a location, with two added factors. First, observation value also considers range. Only those sighting locations within a given range of the given location are considered in the observation value. Second, a weight with values between 0.0 and 1.0 associated with each sighting location is included. The weight is a measure of how important it is to observe that sighting location. Initially all sighting locations have weights of 1.0. We shall see how those weights are changed later.

The FZD algorithm computes the observation value for a given location by invoking the ModSAF point to point intervisibility routine, which returns a value between 0 (not visible) and 1 (completely visible). The observation value for a given location is the sum, for every sighting location in the engagement area within a given range of the given location, of the product of the intervisibility value between the given location and the sighting location and the sighting location's weight. Formally, the observation value for location d and range r is defined as

$$\sum_{s \in S} w(s) \cdot v(d,s) \cdot g(d,s,r)$$

where d is the given location, r is the given range, S is the set of sighting locations, $w(s)$ is the weight of sighting location s, $v(d,s)$ is the ModSAF intervisibility value

from d to s, and $g(d,s,r)$ is a function that returns 1 if sighting location s is within range r of d and 0 otherwise.

Given a set D of defensive locations and a set S of sighting locations, we define the *intervisibility* graph as a complete bipartite graph $V = (D \cup S,E)$ where the edges in E connect every defensive location $d \in D$ to every sighting location $s \in S$. For each edge $(d,s) \in E$, the ModSAF intervisibility value $v(d,s)$ is stored in an array, which represents the graph. Each array entry also stores the distance between d and s, denoted $dis(d,s)$. Once the intervisibility and distance values in the array have been filled in, the observation value for a defensive location $d \in D$ can be computed from the array without invoking the point to point intervisibility routine or performing a distance computation. This saves execution time if the observation value for the locations in a known set will be needed repeatedly, as is the case for defensive locations in the FZD algorithm.

Sighting location weights. As mentioned, associated with each sighting location is a weight with values between 0.0 and 1.0. Initially all sighting locations have weights of 1.0. The weight measures how important it is to observe that sighting location. From the viewpoint of tactical effectiveness, the importance of observing a particular sighting location by an entity declines as a function of how many other entities can also observe that sighting location. In other words, for each entity that can observe a sighting location, it becomes relatively less important to deploy another entity to observe that same sighting location as compared to sighting locations observed by fewer or no entities. The idea is that important for each entity to be able to see as much of the engagement area as possible, but it is also important for all or most locations in the engagement area to be observed by some entity.[18] The FZD algorithm reflects this military consideration. It sequentially chooses deployment locations from among the defensive locations for the entities of a unit, and as each one is chosen, the weights of the sighting locations are reduced. Specifically, for a selected defensive location d, and for every sighting location $s \in S$, $w(s)$ is reduced to

$$w(s) - (h \cdot v(d,s) \cdot w(s))$$

The reduction factor h is an algorithm parameter; it was set $h = 0.5$ for this task. Different values would affect the dispersal of the selected deployment locations. The weight reduction calculation depends on the intervisibility value $v(d,s)$ between d and s, so if s is not highly visible from d, its weight is not reduced much.

Algorithm. For a formal statement of the FZD algorithm, see [19]. Informally, it consists of these steps:

Steps 1 through 3 are executed once for the company.

1. Generate a set of sighting locations in the engagement area. This is done by accessing the CTDB, retrieving all elevation posts and feature vertices within a minimal square enclosing the engagement area, and testing each for enclosure within the engagement area.
2. Partition the company battle position into platoon battle positions and compute platoon TRPs.
   - 2.1 Generate battle position locations evenly spaced along the battle position segments. Compute, store, and accumulate the observation value for each battle position.
   - 2.2 Divide the total of the battle position location observation values by the number of subunits to get the subunit share of the total observation value.
   - 2.3 Step along the battle position locations summing the battle position location observation values. When the accumulated sum exceeds a subunit share, that battle position location is the end of one subunit battle position and the beginning of the next. In this way the unit battle position is partitioned into subunit battle positions at the battle position locations so that the sum of the observation values for the battle positions within each subunit battle position is approximately equal to the subunit share computed in step 2.2. This has the effect that portions of the unit battle position which provide relatively poor observation of the engagement area are included in larger subunit battle positions and portions with relatively good observation become smaller subunit battle positions, concentrating more entities into portions with good visibility. The subunit battle positions, like the unit battle positions, consist of chains of one or more segments; some of the segments in the subunit battle positions may be subsegments of segments in the unit battle position. Figure 1 gives an example of this process.
   - 2.4 Compute the subunit TRPs by dividing an arc between the unit TRPs in proportion to the division of the unit battle position computed in step 2.3. (The arc is a subset of the circle centered on the center of the battle position and passing through the unit TRPs.)
3. Invoke the platoon FZD behavior for each platoon of the company, passing to each one a subunit battle position and the left and right TRP pair computed in step 2 as well as an engagement area TRP computed as the centroid of the engagement area polygon.

Steps 4 through 8 are repeated for each platoon.

4. Construct a platoon battle area rectangle on the platoon battle position with its width extending from one endpoint of the platoon battle position to the other and its depth given by a ModSAF parameter.

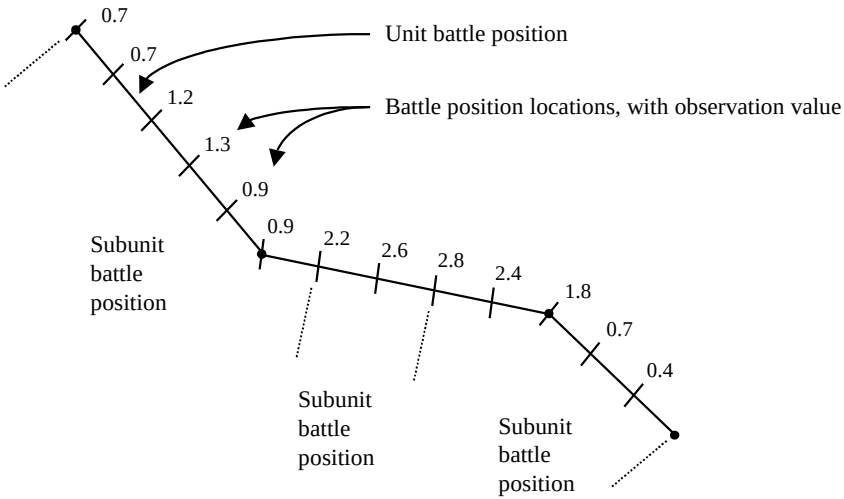

Figure 1. *Unit battle position partitioning example.*

5. Generate a set of defensive locations for the entities of a platoon by invoking the ModSAF cover and concealment algorithm, passing to it the platoon battle position rectangle and the engagement area TRP. This produces a set of defensive locations for the platoon which are the covered or concealed locations within its battle area rectangle.
6. Fill in the array representing the intervisibility graph *V* by determining the point to point intervisibility value $v(d,s)$ and distance $dis(d,s)$ between each defensive location and each sighting location.
7. Select one of the defensive locations as a deployment location for each entity of the subunit using a greedy optimization procedure.
   For each entity:
   - 7.1 For each unselected defensive location *d*, determine the observation value of that location using the intervisibility graph array and the sighting range of the entity. Save the defensive location with the best observation value.
   - 7.2 Select the defensive location with the best observation value; assign it to the entity as its deployment location and mark it as selected.
   - 7.3 Reduce the weights of the sighting locations for the selected defensive location.

8. Invoke the ModSAF route planning and movement functions to move the individual entities of the subunit to their assigned deployment locations.

Optimization comments. The fire zone defense algorithm selects a subset of the defensive locations for the entities. In doing so it attempts to maximize total observation value from the selected defensive locations, i.e., to maximally include the sighting locations in areas that are visible and within range of the selected defensive locations. It does so using a greedy approach, selecting for each entity the defensive location not already assigned to an entity that has the largest observation value.

A strict mathematical maximization of sighting location inclusion that does not consider range and inclusion by previous entities would not necessarily be tactically optimum. The algorithm must balance competing goals such as locating entities where they can observe the largest number of sighting locations against locating entities so as to include every point in the set of sighting locations within a militarily sufficient number of fields of fire. The combination of a greedy selection with a reduction of the sighting location weights after each defensive location selection is a heuristic intended to achieve good tactical effectiveness.

4. Evaluation experiment

This section reports the experiment conducted to compare the HOP and FZD algorithms.

4.1. Experiment overview

The HOP and FZD algorithms both perform the same task: given a CGF unit ordered to take up a defensive position relative to an engagement area, determine good deployment locations for each of the individual entities of that unit. The obvious experiment to compare the two algorithms is to have both perform that task under identical circumstances and evaluate their performance. That obvious approach is precisely what was done. A set of trials, each consisting of a terrain locale, a unit battle position, left and right TRPs, and an engagement area, were created. The HOP and FZD algorithms were run for each trial to produce a deployment for the entities of the unit. Each deployment was evaluated in terms of the cumulative observation metric and the results were compared.

4.2. Experiment design

Trial characteristics. The HOP and FZD algorithms were compared over a sequence of trials. Each trial consisted of a given terrain database, locale on that terrain database, battle position, engagement area, left and right unit TRPs, and military unit.

Two runs were conducted for each trial, one using the HOP algorithm and one using the FZD algorithm.

In order to provide comparable values and eliminate confounding variables, most of the characteristics of the experiment remained fixed over all trials and runs. In particular, these items did not change:

1. Terrain database (the Ft. Knox CTDB version 5)
2. HOP and FZD algorithm implementations
3. ModSAF cover and concealment algorithm
4. Cover and concealment spacing (5.0 meters)
5. Military unit (U.S. M1 tank company)

Certain items varied with the trial, but were fixed for the two runs (HOP and FZD) of a trial. Those were:

1. Terrain locale
2. Engagement area
3. Left and right unit TRPs
4. Unit battle position

Of course, the algorithm used (HOP or FZD) varied between the two runs of each trial.

Evaluation metric. As previously defined, *location observation* is the portion of the engagement area visible from a location and *cumulative observation* is the sum of the location observation values for all the deployment locations selected for the entities of a unit. Cumulative observation was used to compare the performance of the two algorithms. Values for cumulative observation were computed by summing location observation values for the selected deployment locations for each of the entities.

The location observation values were obtained from the ModSAF terrain analysis tool, a feature of the ModSAF user interface that includes an area intervisibility function that graphically displays the area visible from a given point within a given circle, based on the intervisibility values for a grid of points within the circle. The ModSAF terrain tool was slightly modified so as to sum the intervisibility values within a given polygonal area. The sum so computed gives the location observation value for a given point relative to the given polygonal area. When that given polygonal area is an engagement area the intervisibility value sums returned by the terrain tool for a set of locations can be used as the location observation values. Those values were accumulated into a value for cumulative observation, relative to the engagement area, and compared between two deployments for the same engagement area.

Note that the modification did not change the underlying intervisibility process used by the ModSAF terrain tool; rather it simply summed the intervisibility values found to be within a polygon. The point is that the calculation of the cumulative observation metric is based on the ModSAF terrain tool, not on the observation value used by the FZD algorithm, making it an unbiased metric for comparing the algorithms.

4.3. Experiment results

Overall results. Table 1 summarizes the results of the 5 trials. The HOP and FZD cumulative observation values cannot be compared from trial to trial because they depend on the size of the engagement area, the number of features in the engagement area, and the ModSAF map scale when the terrain tool was used. However, they can be compared for the two runs of a trial, and show that the FZD algorithm outperformed the HOP algorithm on every trial.

Table 1: Cumulative observation values for the experiment trials

Trial #	Trial name	Cumulative observation		
		HOP	FZD	Ratio, FZD/HOP
(1)	Tree lines	8,950.75	21,855.58	2.44
(2)	Avenue of approach	10,929.68	13,536.80	1.24
(3)	Canopies	5,640.73	11,600.36	2.06
(4)	Road junction	7,856.39	16,118.99	2.05
(5)	River crossing	5,613.41	15,136.29	2.70

The ratio of the cumulative observation values FZD/HOP for each trial is also given. It can be compared from trial to trial. It shows that the difference in performance varied by trial, ranging from a low of 1.24 to a high of 2.70. The average ratio is approximately 2.10, so in general terms the FZD algorithm performed about twice as well as the HOP algorithm over all of the trials.

No perceptible difference in execution time between the algorithms was observed, so the execution times were not measured. Both algorithms ran in a few seconds.

Trials. Five trials were used to compare the two algorithms. Each was given a name descriptive of the terrain area used for the trial: (1) Tree lines, (2) Avenue of approach, (3) Canopies, (4) Road junction, and (5) River crossing. Trials (1) and (2),

for which figures are given, are described in some detail; the descriptions for trials (3), (4), and (5) are abbreviated.

Figures 2 and 3 show the deployment locations chosen by the HOP and FZD algorithms for trial (1) Tree lines. The company was positioned within a loose cluster of tree lines on a company battle position that runs northeast to southwest and extends approximately 1200 meters. The engagement area is to the northeast of the battle position, is approximately 1500 × 1500 meters in extent, and is bordered on the right and divided across the center by more tree lines. The terrain was generally flat, though the company battle position was slightly elevated relative to the engagement area. This trial was intended to test the performance of the algorithms at avoiding the tree lines near the battle position and finding locations that saw between the tree lines in the engagement area. The HOP algorithm selected locations off the evenly spaced defaults for only 3 entities. The left subunit battle position generated by the FZD algorithm is relatively large because of generally poor observation over much of its length, but some cover locations were found within its battle area along the forward crest of the slight hill and the FZD algorithm chose those as deployment locations. Interestingly, in the small center subunit battle position the FZD algorithm used the default locations, due to the lack of cover and concealment and generally good observation of the engagement area in that battle area.

Figures 4 and 5 show the deployment locations chosen by the HOP and FZD algorithms for trial (2) Avenue of approach. A major road ran northeast to southwest between moderately dense clusters of tree lines and elevation rises, forming an obvious military avenue of approach. The company was positioned among tree lines on a hilltop at the southwest end of the avenue of approach, so as to block movement towards a major road junction just behind it. The company battle position ran northwest to southeast and was approximately 1000 meters long. The engagement area was to the northeast of the company battle position and is approximately 1000 × 2000 meters in extent. This trial was intended to determine if the algorithms would take advantage of the increased observation offered by certain locations on the hilltop near the battle position. The HOP algorithm generally used the default locations. The FZD algorithm fared slightly better, taking advantage of some covered locations on the hilltop behind the battle position with good observation of the engagement area.

In figures 3 and 5 it sometimes appears as if the tanks are touching or overlapping. That appearance is an illusion due to the fact that the tank icons are not shown at the same scale as the map. The algorithm includes a minimum distance between the selected deployment locations as a parameter; in these experiments it was 5 meters.

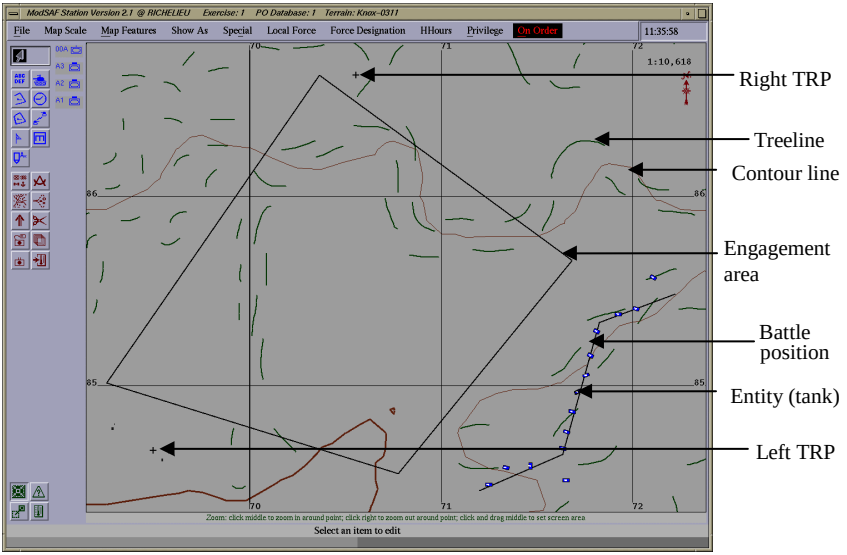

Figure 2. *Hasty Occupy Position deployment for trial (1).*

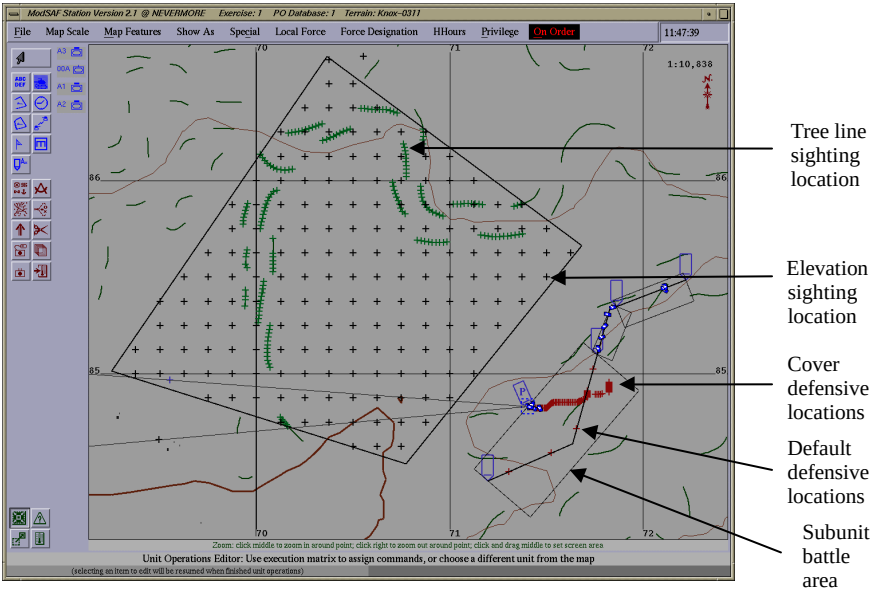

Figure 3. *Fire Zone Defense deployment for trial (1).*

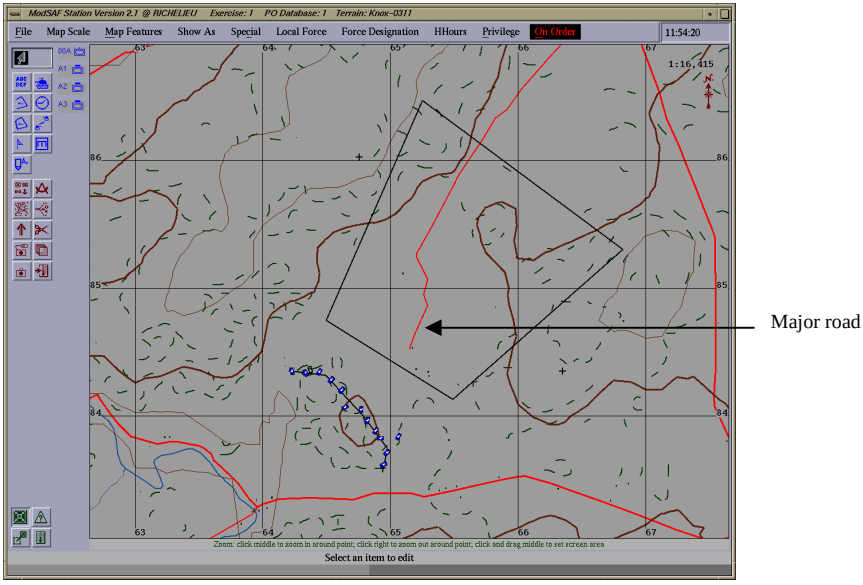

Figure 4. *Hasty Occupy Position deployment for trial (2).*

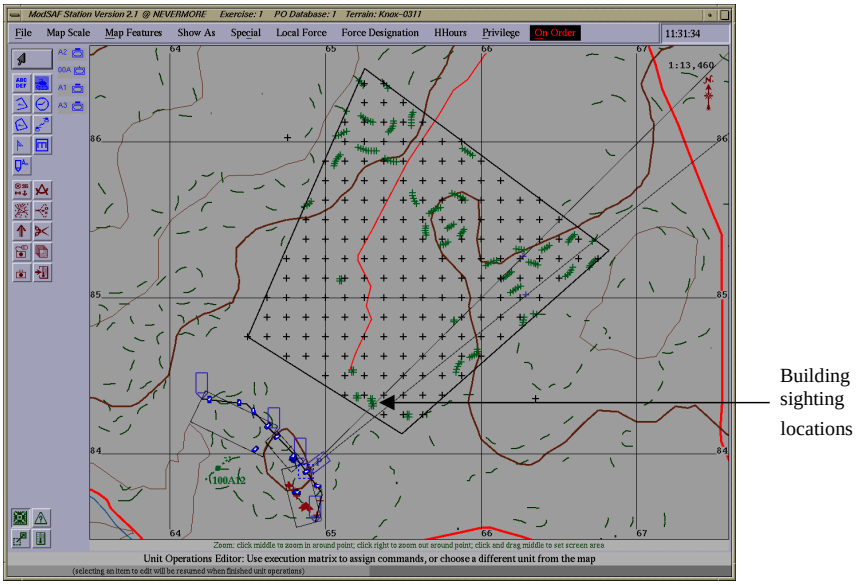

Figure 5. *Fire Zone Defense deployment for trial (2).*

Trial (3) Canopies had a relatively small engagement area located on flat, nearly featureless terrain between three large canopies. The FZD algorithm found better locations within the subunit battle areas behind the battle position. For trial (4) Road junction, the company was positioned adjacent to a major road along the northwest slope of a ridge that runs northeast-southwest, ending near the left end of the company battle position. The FZD algorithm deployed entities on covered locations on top of the ridge behind the battle position in the battle area of the right platoon. In trial (5) River crossing, the company is positioned on the southwest side of a river valley that ran northwest to southeast.

The HOP algorithm deployed entities where they could not observe the engagement area because of canopies and elevation slopes, whereas the FZD algorithm avoided the canopies and found good deployment locations along the forward crest in the left subunit battle area.

Overall comments. The FZD algorithm improves upon the HOP algorithm at two levels. At the company level, it partitions the unit battle position into subunit battle positions in a way that concentrates more entities in areas where better observation is available, while the HOP algorithm partitions the battle position based on length. This concentration is according to defensive doctrine: "The defender seeks to mass the effects of overwhelming combat power ..."[20] Second, at the platoon level, it avoids clearly suboptimal deployment of entities to locations that cannot observe the engagement area because of blocking terrain and instead finds good deployment locations from among the available defensive locations. Recall that both the HOP and the FZD algorithms are using the unmodified ModSAF cover and concealment finding algorithm to generate defensive locations. The difference is that FZD generates defensive locations by subunit, rather than by entity, and it chooses from among them based on observation, rather than by proximity to the default location.

Note that in two ways the FZD algorithm sacrifices optimum performance relative to the cumulative observation evaluation metric in order to gain military tactical effectiveness. First, entity sighting range is considered when computing observation values for subunit battle position partitioning and for selecting entity deployment locations, but it is not considered by the ModSAF terrain tool when computing the observation metric. In the latter case, the terrain analysis tool measures the visible portion of the engagement area from a location without regard to range. The FZD algorithm could have been implemented so as to likewise omit range from its observation value approximations and thereby achieve better values for the cumulative observation metric, but at the risk of deploying entities where they would get credit for observing portions of the engagement area they could not actually observe. We chose not to do so. Second, the reduction of the weights of observed sighting locations during the greedy optimization at the platoon level is intended to

encourage broader, and more militarily effective, coverage of the engagement area by reducing the importance of sighting locations each time they are observed. Of course, that tends to reduce performance relative to the cumulative observation metric because the metric measures the portion of the engagement area observed without regard to how many entities observe the same area. Eliminating the weight reduction step from the FZD algorithm would have boosted its cumulative observation values. Again, we chose not to do so.

Conclusions

The existing HOP algorithm and our FZD algorithm are intentionally identical in functionality; they both defensively deploy the entities of a company sized unit, they both consider cover and concealment, and they both take as input a battle position and an engagement area. However, the experiment results show that with respect to the cumulative observation metric the FZD algorithm is approximately twice as effective as the HOP algorithm. This is a significant improvement and meets our goal in this task.

FZD outperforms HOP because it considers observation when generating a deployment. The use of geometry in choosing the sighting and defensive locations allows the FZD algorithm to consider observation when generating a deployment without excessive computation. The FZD algorithm as implemented also provides a framework for more sophisticated geometric analysis methods.

Acknowledgements

This project was partially supported by the UCF Institute for Simulation and Training (Petty and Franceschini) and the UCF Office of the Vice President for Research and Graduate Studies (Mukherjee). Old Dominion University supported the preparation of this paper for submission. That support is gratefully acknowledged. Lee Napravnik provided help with the ModSAF integration. Sumeet Rajput and Hai-Lin Nee also provided assistance.

References

1. A tutorial on CGF systems can be found in Mikel D. Petty, "Computer generated forces in Distributed Interactive Simulation," *Distributed Interactive Simulation Systems for Simulation and Training in the Aerospace Environment*, SPIE Critical Review 58 (Orlando FL, April 19-20 1995), 251-280.

2. For additional information on ModSAF the reader may refer to A.J. Courtemanche and A. Ceranowicz, "ModSAF Development Status," *Proceedings of the Fifth Conference on Computer Generated Forces and Behavioral Representation* (Orlando FL, May 9-11, 1995), 3-13.
3. For additional information regarding the CTDB format see J.E. Smith, "Recent Developments in ModSAF Terrain Representation", *Proceedings of the Fifth Conference on Computer Generated Forces and Behavioral Representation* (Orlando, FL: Institute for Simulation and Training, May 9-11, 1995), 375-381.
4. D.E. Mullally, *Personal communication* (1997).
5. U.S. Army, *Operations, U. S. Army Field Manual No. 100-5* (U.S. Government Printing Office, 1976).
6. Ibid.
7. U. S. Army, *Operations, U.S. Army Field Manual No. 100-5* (U.S. Government Printing Office, 1993).
8. Mullally, *Personal communication.*
9. E.C. O'Byrne, "Dismounted Infantry: Indispensible to the Virtual Battlefield," *Proceedings of the 15th Interservice/Industry Training Systems and Education Conference* (Orlando, FL, November 19-December 2, 1993), 783-791.
10. U. S. Marine Corps, *Ground Combat Operations, United States Marine Corps Operational Handbook 6-1,* (U.S. Government Printing Office, 1988).
11. D.M. Keirsey, J. Krozel, and D.W. Payton, "Scale-Space Representations for Flexible Automated Terrain Reasoning," *Proceedings of the U.S. Army Symposium on Artificial Intelligence Research for Exploitation of the Battlefield Environment* (El Paso TX, November 15-16, 1988), 108-118.
12. T.M. Cronin, "Allocating Sensor Envelope Patterns to a Map Partitioned by Territorial Contours," *Proceedings of the U.S. Army Symposium on Artificial Intelligence Research for Exploitation of the Battlefield Environment* (El Paso TX, November 15-16 1988), 65-78.
13. M.R. Hieb, G. Tecuci, J.M. Pullen, A. Ceranowicz, and D. Hille, "A Methodology and Tool for Constructing Adaptive Command Agents for Computer Generated Forces," *Proceedings of the Fifth Conference on Computer Generated Forces and Behavioral Representation* (Orlando FL, May 9-11 1995), 135-146; D. Hille, M.R. Hieb, G. Tecuci, and J.M. Pullen, "Abstracting Terrain Data Through Semantic Terrain Transformations," *Proceedings of the Fifth Conference on Computer Generated Forces and Behavioral Representation* (Orlando FL, May 9-11 1995), 355-365.
14. S. Rajput and H. Tu, "Terrain Analysis for Determining Tactical Positions," *Unpublished manuscript* (Institute for Simulation and Training, 1997).
15. Courtemanche and Ceranowicz, "ModSAF Development Status."
16. M.J. Longtin, "Cover and Concealment in ModSAF," *Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation* (Orlando FL, May 4-6 1994), 239-247.
17. D.R. Van Brackle, M.D. Petty, C.D. Gouge, and R.D. Hull, "Terrain Reasoning for Reconnaissance Planning in Polygonal Terrain," *Proceedings of the Third Conference on Computer Generated Forces and Behavioral Representation* (Orlando FL, March 17-19 1993), 285-305.

18. U. S. Army, *Operations, U. S. Army Field Manual No. 100-5* (U.S. Government Printing Office, 1986).
19. Mikel D. Petty, "Computational Geometry Techniques for Terrain Reasoning and Data Distribution Problems in Distributed Battlefield Simulation", *Ph.D. Dissertation*, School of Computer Science (University of Central Florida, 1997).
20. *U.S. Army Field Manual No. 100-5* (1993).

MIKEL D. PETTY is Chief Scientist of the Virginia Modeling, Analysis and Simulation Center at Old Dominion University. He has been performing and leading modeling and simulation research since 1990. He received a Ph.D. from the University of Central Florida in 1997, an M.S. from UCF in 1988, and a B.S. from the California State University Sacramento in 1980, all in Computer Science. His research interests are in distributed simulation interoperability, computer generated forces, and applied computational geometry; he has published over 70 papers in those areas. Address: Virginia Modeling, Analysis & Simulation Center, Old Dominion University, 7000 College Drive, Suffolk VA 23435. tel.: (757) 686-6210. E-mail: mpetty@vmasc.odu.edu.

ROBERT W. FRANCESCHINI is a Senior Research Computer Scientist at the Institute for Simulation and Training and a Visiting Assistant Professor in the School of Electrical Engineering and Computer Science at the University of Central Florida. He has performed research in distributed simulation, computer generated forces, multi-resolution simulation, data compression, and graph theory. He has over 30 published papers in those areas. Dr. Franceschini received a B.S. in Computer Science from the University of Central Florida in 1992 and a Ph.D. in Computer Science from UCF in 1999. Address: Institute for Simulation and Training, University of Central Florida, 3280 Progress Drive, Orlando FL 32826-0544. tel.: (407) 658-5519. E-mail: rfrances@ist.ucf.edu.

AMAR MUKHERJEE is a Professor of Computer Science in the School of Electrical Engineering and Computer Science at the University of Central Florida. Professor Mukherjee received the D.Phil. (Sc.) degree from the University of Calcutta in 1962 at the Institute of Radiophysics and Electronics. Since then his research has been focused on VLSI design, algorithms, and architectures. He has previously held faculty positions at the University of Iowa, Montana State University, and Princeton University. Professor Mukherjee is a Fellow of the IEEE and served two terms as Editor of the journal *IEEE Transactions on Computers*. Address: School of Electrical Engineering and Computer Science, University of Central Florida, 4000 Central Florida Boulevard, Orlando FL 32816-2362. tel.: (407) 823-2763. E-mail: amar@cs.ucf.edu.