Enhanced Battlefield Visualization for Situation Awareness

Young J. Kim
Department of Computer Science
University of North Carolina at Chapel Hill
youngkim@cs.unc.edu

Christoph M. Hoffmann
Department of Computer Science
Purdue University
cmh@cs.purdue.edu

Abstract

We present tools to visualize density, clustering, and lethality assessment that help a military commander achieve situation awareness (SA) on the battlefield. For the density computation, we provide a geometry-based, image-based, and hybrid approach to tackle the problem. We choose proximity-based clustering as a suitable definition for our application. Based on this clustering definition, we show an efficient computational method based on the Delaunay triangulation. Finally, we present a probabilistic model for the lethality assessment and an efficient way to compute it using hardware support where available. We also explain how to deliver visual information to the human visual system effectively. The principles of human visual perception, such as preattentive and Gestalt perceptual processing, play an important role in this work.

Keywords: Battlefield Visualization, Situation Awareness, Density, Clustering, Lethality Assessment

1 Introduction

Suppose you are a top air defense commander who is in charge of making a strategic decision on the intrusion of enemy forces in a timely manner. You would be provided with tens of thousands of tactical data arriving from every corner of the world every minute. Your job is to quickly grasp and react to changes in the evolving battle; you must possess good situation awareness (SA). This problem is known as *command and control* in the military community. However, understanding the huge positional data and combining it into a single comprehensive view of the battlefield can be extremely difficult, error prone, and time consuming [Durbin et al. 1998a; Durbin et al. 1998b]. Thus we need to reduce the complexity of the information without losing the essential meaning.

We postulate that geometric queries on the positional data set and their appropriate visual presentation can reduce cognitive overhead. In this paper, we present various techniques from visualization and computational geometry to tackle the SA problem. The principles of human visual perception also play an important role in this work.

1.1 Main Results

The major results of our paper include the following:

- We present density (concentration), clustering (boundary detection), and lethality assessment (threatening level) as visual tools to help a user (military commander) perform exploratory tasks on large multi-dimensional datasets rapidly, accurately, and effectively.

- We provide various methods to efficiently compute the density, clustering, and lethality assessment. For the density computation, we provide a geometry-based, image-based, and hybrid approach to tackle the problem. We choose proximity-based clustering as a suitable definition for our application. Based on this clustering definition, we show an efficient computational method based on the Delaunay triangulation. Finally, we present a probabilistic model for the lethality assessment and an efficient way to compute it using hardware support where available.

- We discuss how to render the computed results in a computationally effective, as well as perceptually efficient, way to the human visual system. In order to achieve this objective, we take advantage of two methods from the domain of perceptual psychology: preattentive features and Gestalt perception in human perceptual processing.

1.2 Paper Organization

The rest of the paper is organized as follows. In Section 2, we explain the preliminary concepts used throughout the paper. We also review relevant prior work. In Section 3, we identify various visual tools (density, clustering, and lethality assessment) that help a military commander achieve situation awareness on the battlefield, and we give geometric techniques to compute them efficiently. In order to gain perceptual effectiveness using the tools, Gestalt perception and preattentive features in the human visual system are considered, thereby showing that the tools are appropriate. In Section 4, we consider how to render the results of the visual tools discussed in Section 3. An efficient hardware-accelerated technique is presented for each tool. We show the performance results of our techniques in Section 5, and we summarize the paper and suggest future work in Section 6.

2 Preliminaries

We start this section by explaining two important notions in our work, situation awareness and battlefield visualization. We also review related previous work.

2.1 Situation Awareness

Situation awareness is the concept of describing the performance of a domain expert during the operation of complex systems, such as aircrafts, vehicles, and chemical plants. The concept was first brought up in discussing the critical difference between ordinary fighter pilots and ace pilots [Sukthankar 1997]. Due to the relative importance of different aspects in situation awareness, there has been a broad range of definitions of situation awareness.

From the point of view of human factors, Endsley [Endsley 1988] describes situation awareness as follows:

> "An expert's perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future."

From the military point of view, Blanchard [Blanchard 1996] depicts battle space awareness as "knowing what is needed to win with the minimum number of casualties." He identifies situation awareness as one of seven core concepts in battle space awareness and refers to it as a situation of friendly and enemy forces.

2.2 Battlefield Visualization

Traditionally, paper maps and acetate overlays have been used predominantly in the military to visualize situations. Only very recently, virtual reality technology and high computing power have started to take the place of acetate sheets and have proven to work just as well in a real battle scenario. Durbin et al. claim that such a battle space awareness system must present a comprehensive and timely view of the battle environment [Durbin et al. 1998a]. The Army TRADOC pamphlet [U.S. Army Combined Arms Center 1995] further explains the term, battlefield visualization:

> "The process whereby the commander develops a clear understanding of the current state with relation to the enemy and environment, envisions a desired end state which represents mission accomplishment, and then subsequently visualizes the sequence of activity that moves the commander's force from its current state to the end state."

Thus, it can be concluded that helping a commander develop situation awareness plays a crucial role in battlefield visualization and thereby relates directly to mission success.

2.3 Previous Work

2.3.1 Situation Awareness Application

Many applications from avionics and the human factors community are deemed successful because of the effective use of situation awareness. The goals of such applications range from training [Kass et al. 1990] to tactical engagements [Rosenbloom et al. 1994]. They adopt multi-modalities to increase the situation awareness of a human subject, using head mounted displays (HMD), data gloves, data suits, chamber technology, and a cave environment. Most of them are also incorporated into an immersive display technology or virtual reality facility. Extensive references to situation awareness in the avionics applications can be found in [Resource Coordinator 2000].

TSAS (Tactile Situation Awareness System) is a system to utilize a human's touch in order to deliver situation awareness. It is designed to improve the performance of a human pilot in simulated rotorcraft under high-load working conditions. The unique feature of this system is that, unlike other packages, it does not rely on visual or aural information for an efficient delivery of situation awareness. Instead, a wearable suit equipped with a tactile device is provided as an intuitive human computer interface to three-dimensional space. The system helps a pilot avoid a three-dimensional disorientation during the maneuver of a rotorcraft [Human and Cognition 2000].

Sentinel is a software tool that provides an analysis capability of the battlefield to a military commander. It helps maintain situation awareness [Stytz et al. 1993]. The tool gives the user an indication of the importance of an action within a "watch space," a designated area by the user. The importance indication is displayed on a configurable status board, and fuzzy logic is used to judge importance. The goal of Sentinel is very similar to our application in the sense that it tries to make the complex battlefield environment intuitive by providing situation awareness. However, Sentinel lacks the consideration of a human perceptual process when displaying the importance factor on the status board. Therefore, there is a possible danger that it can lead to a perceptual overhead and delay as the delivered information becomes more complex and increases in volume.

Research on the tactical driving of an autonomous vehicle shows a novel application of situation awareness [Sukthankar 1997]. SAPIENT (Situation Awareness Planner Implementing Effective

Navigation in Traffic) is developed as part of the Automated Highway System (AHS) to allow an intelligent vehicle to operate without human control. The goal of SAPIENT is to simulate the situation awareness of a human cognitive process in traffic. That way, the system achieves the intelligent behavior of a vehicle.

2.3.2 Battlefield Visualization Package

The primary concern in battlefield visualization packages has been real time terrain visualization, simply because terrain is an important referent in the battlefield. Typically, terrain can be modeled either as a regular grid [Lindstrom et al. 1996; Duchaineau et al. 1997; Pajarola 1998] or as a triangulated irregular network (TIN) [Rossignac and Borrel 1992; Schroeder et al. 1992; Cohen et al. 1996; Hoppe 1996; Hoppe 1997]; however, in practice, its raw data size can be in the tens of millions of triangles. Therefore, many efforts have been made to reduce the substantial volume of triangle data while minimizing visual artifacts. Progress in the mesh reduction technique combined with Level of Detail (LOD) control has played a major role in that work.

The alternative to such object-based techniques is to use image-based rendering and volume-rendering techniques [Mueller et al. 1998]. Unfortunately none of these efforts have been made in the context of battlefield visualization.

The Naval Research Laboratory's Virtual Reality Responsive Workbench (VRRWB) and the Dragon software system are well known in battlefield visualization. With the extensive use of virtual reality technology, the dragon system is considered a critical breakthrough over the previous battlefield visualization system [Durbin et al. 1998b]. However the system does not employ any technology for reducing the cognitive overhead of a commander, and it merely provides a "3D metaphor" of a real battle.

Plan View Display (PVD) from MÄK technologies provides a *birds-eye-view* into a simulated battle by overlaying entities and information onto 2D views of tactical, strategic, and visual databases [MAK n. d.]. The overlayed information includes tracking individual entities and groups of entities and displaying intervisibility of entities and points. Moreover, since PVD is compatible with both Distributed Interactive Simulation (DIS) and High Level Architecture (HLA) simulation protocols,

it is highly interoperable with other products. However PVD lacks consideration of the human visual perception process, such as preattentiveness; thus the visual presentation in PVD could bring about an additional cognitive overhead.

The Army Research Laboratory's updated Virtual Geographic Information System (VGIS) has the same objective as our application: providing visual information to a commander to help build situation awareness [Walrath et al. 2000]. The main functionality of this new system is to display the concentration information of battlefield entities. The system uses a grid (raster) based approach to compute concentration and constructs isosurfaces of concentration by considering the concentration value as vertical data in three-dimensional space. Depending on the viewpoint, concentration is shown by height or by color intensity. Thereby, the vertical data serves as a redundant encoding of concentration. Another functionality in the system is that it temporally condenses a potentially lengthy battle into a MPEG movie. Later, the movie can be played back at a desired speed. By doing this, the commander can grasp the strategic or tactical implications of lengthy battles that might be missed otherwise.

3 Computational Techniques for Situation Awareness

In this section, we suggest three visual tools - *density*, *clustering*, and *lethality assessment* - to enhance situation awareness on the battlefield and investigate how to compute them efficiently. In the following section, we explain how to visualize them.

We start by investigating the theory of perceptual psychology to justify why we compute these visual tools and how we should display them.

3.1 Applying the Theory of Perception

Usually the battlefield is represented by large datasets with many attributes. These large datasets make it difficult for a user, in our case a military commander, to assess a situation on the battlefield in a timely manner. Moreover, it is more troublesome when it comes to dealing with multi-attribute or multi-dimensional datasets since the user must spend more time building up a single comprehen-

sive view of a battle.

We postulate that an appropriate visual interpretation of the battlefield helps a military commander build up a comprehensive view of the battlefield effectively and rapidly and, as a result, to make a strategic decision accurately. In order to accomplish this objective, we employ two important methods from the domain of perceptual psychology: preattentive features and Gestalt perception in human perceptual processing. Taking advantage of low level human visual systems, these perceptual techniques allow users to perform exploratory tasks on large multi-dimensional datasets rapidly, accurately, and effectively. Such tasks include identifying concentration (density), boundary detection (clustering), and threat level (lethality assessment) from a large positional dataset.

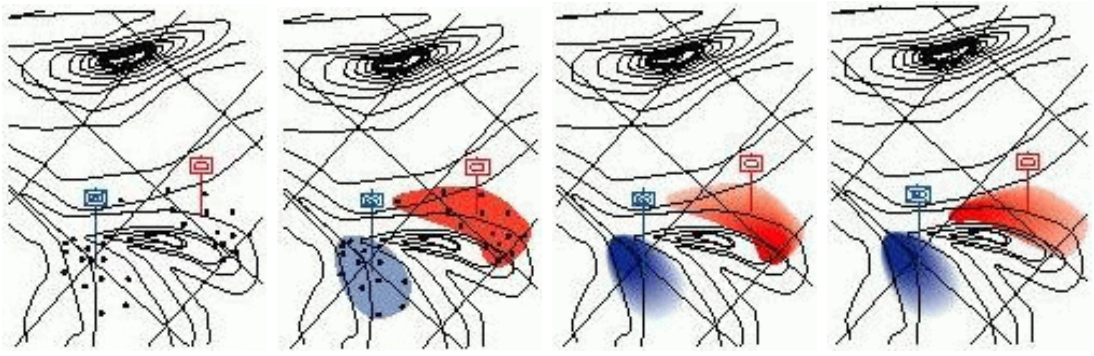
3.1.1 Preattentive Processing

Psychophysicists have identified a limited set of features that the human visual system detects very quickly without the need for searching. These features are called *preattentive*, and they include color (hue), intensity, texture, length, and width. A good survey of these features can be found in [Healey 1999]. It is known that by using these features one can perform various exploratory tasks independent of the total number of elements involved in the tasks and that these tasks can be performed in a single glance in less than 200 milliseconds. It has also been shown experimentally that the preattentive features can work not only in a static setting but also in a dynamic setting where datasets are constantly changing. Furthermore, some preattentive features have no interferences with each other, and they seem to be prioritized by the human visual system [Callaghan 1989; Callaghan 1990]. For instance, form, hue, and intensity are prioritized features.

In our application, such hierarchy of features is used for density and clustering visualization later in Section 4.3.2. We use the HSV color scheme to exploit the feature hierarchy. Since we have only a limited number of preattentive features that do not interfere with each other, we must limit the number of features coexisting in a single view. We resolve this problem by distributing some of the features into different views (windows) while minimizing the number of views.

3.1.2 Gestalt Perceptual Processing

Early in the 20th-century, Gestalt psychologists observed that when elements were gathered into a figure, the figure took on a perceptual salience that exceeded the sum of its parts. Then, two decades ago, it was demonstrated that people extract the global aspects of a scene before smaller (local) details are perceived [Navon 1977; Hoffman 1980]. Taking advantage of this pre-attentive, global processing we can make the density distribution of troops within a geographical area instantly apparent; see Figure 1.



(a) Point Map (b) Shaded Map (c) Density Map (d) Density Map

Figure 1: *Various Maps (from [Hoffmann et al. 1998]). In map (a) where an individual battle unit is displayed as a point, it takes some time to recognize the whole troop boundary and concentration. In a shaded map (b) the troop boundary becomes easy to recognize, but it is still hard to see the troop concentration. In density maps (c) and (d), the troop concentration becomes apparent. A correct understanding of the troop concentration depends on a correct display.*

The user can thus determine troop concentrations with less visual interrogation and cognitive effort because the differential saturation is immediately apparent. A user study conducted jointly by the authors and the Army Research Laboratory shows that displaying different levels of concentration to the users helps them achieve higher levels of situation awareness rapidly [Emmerman et al. 1998; Hoffmann et al. 1998]. In this experiment, the users are divided into two groups, and each group is provided with a set of points in the standard format (SF) and in the enhanced format (EF) respectively; the SF format includes data attributes such as individual unit identifications,

positions, and command affiliations, and the EF format is further augmented with concentration information. Then, the users are asked to answer various questions regarding who is where and what is happening on the battlefield. From this experiment, we conclude that density is a very important tool for enhancing situation awareness.

It was previously known, and we also found out experimentally, that some preattentive features, such as blinking, can be distracting and can interfere with the gestalt perception process. Therefore, one should be very careful about putting various preattentive features together and ensure that they do not interfere with each other while performing the preattentive processing.

3.2 Density Computation

By *density* of a point, we mean a measure of how many other points are close by. Both the number and distance of the nearby points are considered. We discuss three approaches to compute density, namely a geometry-based approach, an image-based approach, and a hybrid approach. These different approaches have their own strengths. For example, the geometry-based approach can be highly adaptable to a dynamic environment by utilizing its underlying geometric coherence, whereas the image-based approach can provide immediate rendering of its computational result.

3.2.1 Geometry Based Approach

We rely on the Delaunay triangulation to compute density. In the Delaunay triangulation, nearness is approximated by adjacency [de Berg et al. 1997]. Moreover, distance from a nearby point can be measured by the edge length [Keil and Gutwin 1989]. The Delaunay approach has two important properties: it is the dual of the Voronoi diagram of points, and it maximizes the minimum angle over all possible triangulations of points. The duality property allows us to know which point surrounds each point by following its incident edges. The maximal angle property is helpful for an approximate density rendering in which the triangulation is lifted to a 3D polygonal terrain using the assigned point height as the third coordinate. Another attractive property is that the Delaunay triangulation has been widely studied and can be implemented in a numerically stable way [Shewchuk 1996; Devillers 1992; Mehlhorn and Näher 1998].

ComputeDensityUsingDT(DT)

Input A Delaunay triangulation DT of n points in the plane.

Output Density assignment on each vertex in DT.

1. Set each vertex density to zero.
2. For each edge incident to vertex p, compute the edge length and classify into one of a small number of length ranges.
3. Increase the density of p by an integer derived from the length range.

ALGORITHM 3.1: ComputeDensityUsingDT

Once the Delaunay triangulation has been computed, density assignment can be done as in Algorithm 3.1. Density at each point is approximated from the triangulation by summing a quantized edge length for every incident edge and then scaling this value to a predetermined range that is based on an exponential quantization of the edge length sum. We feel that using a quantized edge length (vs. continuous lengths) and an exponential quantization of the edge length sum (vs. linear or quadratic functions) results in a less ambiguous rendering of density. After all vertices have been processed, the average density is obtained as the density value divided by the number of incident edges.

The density computation requires $O(n)$ steps for the edge quantization and averaging and $O(n \log n)$ steps for the Delaunay triangulation construction.

3.2.2 Image-Based Approach

In the image-based approach, we assume the existence of mapping from geographic coordinates to pixel positions displayed on a digitized map. Using the mapping, we define a local density function from each grid position or pixel to neighboring pixels. This local density function defines an area of influence around each pixel position containing at least one entity. For each entity, we sequentially compute the appropriate pixel position and then cumulatively apply the local density function to

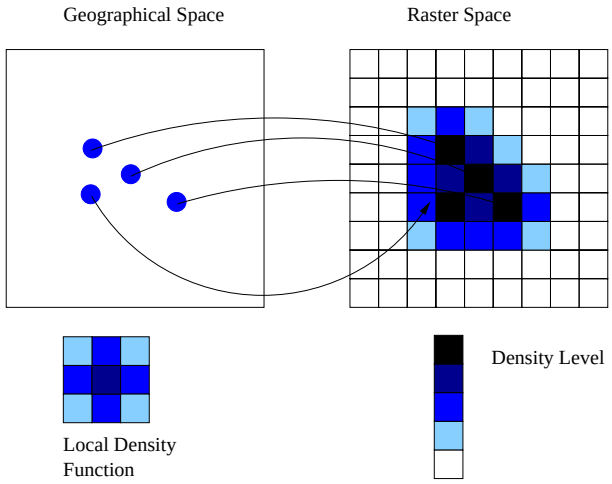


Figure 2: *Image-Based Approach to the Density Computation. Based on mapping from geographical space to raster space, the density is accumulated by superimposing the local density function.*

each neighboring pixel in the area of influence around that pixel. After all entities are processed, the result is a pixel array containing the global cumulative results of the repeated application of the local density functions [Hoffmann et al. 1998].

The image-based approach is attractive because it is very easy to implement. Another advantage is that a constant number of pixels are processed for each report. Thus spatial and temporal processing complexity can be made linear with respect to the number of entity position reports.

3.2.3 Hybrid Approach

A particular difficulty in the image-based approach involves constructing the local density function in a uniform and automatic fashion. One possible solution is to precompute a table of all the possible local density functions and retrieve them as needed. Clearly this requires extra storage and cannot be easily customized and extended by an end user.

When fast graphics hardware support is available, one can exploit the hardware to construct highly flexible local density functions. The technique is similar to Hoff et al.'s idea of computing a Voronoi diagram [Hoff et al. 1999] but is applied from a different perspective. We use the fast rendering pipeline and *alpha blending function* available in OpenGL as in Algorithm 3.2.

We can further accelerate the density computation by storing the result of step 4 into a template

ComputeDensityUsingAlpha(P)

Input A set P of n points in the plane.

Output A buffer B containing density assignment of P.

1. Define the local density function either in an explicit or implicit form.
2. *Polygonalize* it.
3. Set the source and destination alpha to one in the alpha blending function[1]; i.e. $DestinationColor = SourceColor \cdot 1.0 + TargetColor \cdot 1.0$.
4. For each entity p in P, render the polygonalized density function using the above alpha blending formula.
5. The resulting density distribution is stored in a frame buffer.

ALGORITHM 3.2: ComputeDensityUsingAlpha

buffer and reusing it later instead of re-rendering it. This technique is applied also to lethality assessment; see Section 3.4.2.

3.3 Clustering

A military commander will want information about the formation of adversary forces in order to react appropriately to a possible attack from the enemy or to counter-attack the enemy. However, such formation information about the opposite side is not readily available. In this situation, the commander should infer the enemy formation from the available data, usually positional data. For instance, a database file with useful attributes, such as enemy position, date, and weapon type, can be furnished via a surveillance satellite or aircraft in a track file form. One possible way to infer the enemy formation from the track file is to use the proximity of enemy entities to each other since entities in the same unit tend to move together. This is a clustering problem.

3.3.1 Clustering Techniques

Depending on the application, the clustering problem has different objective functions. In general, the clustering problem is known to be NP-hard regardless of the objective function. Moreover, for Euclidean space, it is NP-hard to approximate to within factors close to two in higher than one dimension. Therefore, most clustering algorithms work only on a fixed number, k, of clusters.

In *k-center clustering* or *pairwise clustering*, the objective function is to minimize the radius or diameter of each partitioned cluster. The *doubling algorithm* is a typical example of such a k-center clustering algorithm. Here, the objective function is to minimize the maximum cluster diameter [Charikar et al. 1997].

In *variance-based clustering*, the objective function is to minimize the sum of squared errors in each cluster. The *Voronoi diagram-based approach* is one such algorithm [Inaba et al. 1994]. The main idea is that an optimum clustering that minimizes such an objective function is a Voronoi partition, i.e., the ordinary Euclidean Voronoi diagram for some k points.

3.3.2 Delaunay-Based Approach

One can also think of a different definition of clustering. Suppose points belonging to the same cluster should move maintaining at most a given maximum distance from some of their neighbors. In this case, the outline of clustering can be any arbitrary shape, for example, a *sickle* shape. This is a particularly important case for the military unit formation: each entity in a military unit is moving while maintaining some distance from others, as shown in Figure 3. We can define this type of clustering formally as follows:

DEFINITION 3.1 *Given n points in R^d and distance threshold r, find clusters $S_1, S_2, ..., S_k$ which satisfies the following*

1. $\forall x_i \in S_l, \exists x_j \in S_l$, *such that* $|x_i - x_j| \leq r$.

2. $\forall x_i \in S_l, \forall x_j \in S_m$ *where* $l \neq m$, *we have* $|x_i - x_j| > r$.

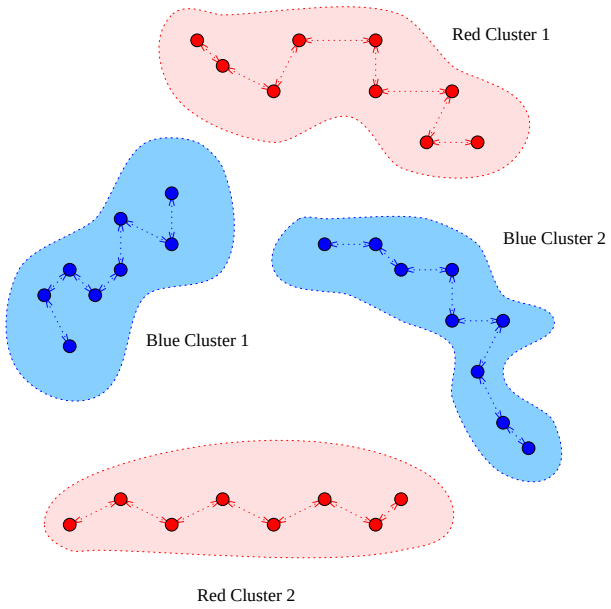


Figure 3: *A Typical Example of Clustering in the Application. Each cluster has its own threshold value to be its member, which is a maximum distance between members. The dotted arrow between any two points denotes such a maximum distance within a cluster.*

Fortunately, this computation is not NP-hard and can be solved easily, especially in 2D based on lemma 3.1.

LEMMA 3.1 *[Dickerson and Drysdale 1990] Let S be a set of distinct points on a plane, δ a distance, and D the Delaunay triangulation of S. If $|p,q| \leq \delta$ for $p,q \in S$, then either $< p,q >$ is an edge in D or there exist distinct points $o_1, o_2, ..., o_m$ such that:*

1. $< p, o_1 >, < o_m, q >$ *and* $< o_i, o_{i+1} >$ *are edges in* D *for* $1 \leq i < m$,
2. $|p, o_1| \leq \delta, |o_m, q| \leq \delta$, *and* $|o_i, o_{i+1}| \leq \delta$ *for* $1 \leq i < m$, *and*
3. $|p, o_i| \leq \delta, |o_i, q| \leq \delta$ *for* $1 \leq i < m$.

We can compute clusters as in Algorithm 3.3. Once the Delaunay triangulation has been computed, the computation requires $O(n)$ running time for edge cutting plus extracting connected components by a Depth First Search (DFS) on the triangulation.

ComputeClustering(P, r)

Input A set P of n points in the plane, and minimum distance to neighbors in a same cluster, r.

Output Return clustered sets $S_1, S_2, ..., S_k$ of P.

1. Compute a Delaunay triangulation of the given points.
2. Cut edges whose length is more than the given threshold, r.
3. Compute connected components and output each component as a different cluster S_i.

ALGORITHM 3.3: ComputeClustering

3.4 Lethality Assessment

Consider tank warfare. Every battle unit, a tank, has an associated threat region and also a preferred direction of threat, the direction in which the on-board cannon of the tank is positioned. Typically that is the direction in which the tank moves. The commander must assess overall lethality distribution of all battle entities involved in warfare in order to appropriately avoid a "danger zone" or even safely retreat friendly forces from the combat zone if necessary.

In light of the computation, a lethality assessment is similar to the density computation discussed in Section 3.2 in that the concept of a threat region merely replaces the local density function in the density computation.

3.4.1 Probabilistic Model

We model the threat region with a probability density function that has exponential decay in two dimensions; see Figure 4.

Here the probability measures the likelihood of being killed at a certain position in the plane. Note that the probability outside the preferred direction range is zero. We model the killing probability by the penetration ratio of steel armor hit by a projectile. Assuming ideal conditions, the penetration is fully governed by the kinetic energy E of the projectile [Okun 1998].

Consider the trajectory of a projectile fired by a cannon where air resistance is the only force

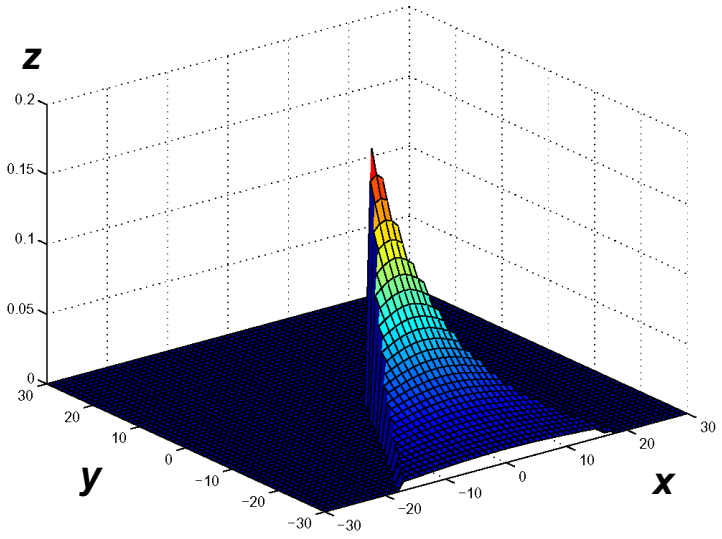


Figure 4: *Threat Region Model. The x and y axes define a planar region where lethality is assessed, and the z axis denotes the probability density function (i.e., lethality) for a given location in the planar region.*

exerted on the projectile after shooting. Other parameters, such as gravity, firing angle, and wind, are ignored. Since the projectile moves fast, the air resistance increases proportionally to the square of velocity (Newtonian drag) $F = -kv^2$, where k is the coefficient of air resistance [Wilson 2000]. Thus we get the following equation of the motion of the projectile,

$$m\frac{d^2x}{dt^2} = -k(\frac{dx}{dt})^2 \tag{1}$$

where m denotes mass of the projectile and k denotes the air resistance coefficient.

By solving the nonlinear differential equation (1), we get the following equations for the velocity v and the displacement D of the projectile,

$$D = x = \frac{m}{k}\log(\frac{v_0kt + m}{m}) \tag{2}$$

$$v = \frac{dx}{dt} = \frac{mv_0}{v_0kt + m} \tag{3}$$

where v_0 denotes the nozzle velocity of the projectile.

From equations (2) and (3), we get the following relationship between the kinetic energy E and

the displacement D,

$$\begin{aligned} E &= \frac{1}{2}mv^2 \\ &= \frac{1}{2}m\left\{v_0 \exp\left(-\frac{k}{m}D\right)\right\}^2 \\ &= \frac{1}{2}mv_0^2 \exp\left(-\frac{2k}{m}D\right) \\ &= E_0 \exp\left(-\frac{2k}{m}D\right) \end{aligned} \tag{4}$$

We see the exponential decay of the penetration (kinetic energy) as a function of the displacement.

Once all threat regions have been defined, we compute the overall lethality using the additive rule for two independent random variables; i.e., $p_{composite} = p_1 + p_2 - p_1 p_2$.

3.4.2 Image Based Approach

The computational method used for the lethality assessment is similar to the hybrid approach in section 3.2.3, except that it uses a different blending formula. The main idea is that we interpret the destination color (D_c) as probability p_1 with alpha value (A_2) $1 - p_2$ and the source color (S_c) as probability p_2 with value alpha (A_1) 1, since

$$\begin{aligned} D_c &= S_c A_1 + D_c A_2 \\ &= p_2 \cdot 1 + p_1(1 - p_2) \\ &= p_1 + p_2 - p_1 p_2 \end{aligned} \tag{5}$$

Initially, p_1 is zero, and it iteratively accumulates the composite probability by blending the probability p_2 of each threat region. So the whole process is given in Algorithm 3.4.

Unlike the density computation, we cannot reuse the result of step 4 since the threat region is constantly changing.

4 Visualization Issues

In this section, we address the problem of how to render the concepts that we discussed in the previous section. In order to assure situation awareness, visualization should be delivered to a commander in a way that is both perceptually meaningful and computationally efficient.

AssessLethalityUsingAlpha(P)

Input A set P of n points in the plane.

Output A buffer B containing lethality assessment of P.

1. Define the threat region either in an explicit or implicit form.
2. *Polygonalize* it.
3. Set the alpha blending formula as in equation 5.
4. For each threat region, render its polygonalized function using the above alpha blending formula. See the illustration in Figure 5.
5. The resulting lethality assessment is stored in a frame buffer.

ALGORITHM 3.4: AssessLethalityUsingAlpha

4.1 Density Visualization

Depending on the way we compute the density, we can render the density distribution using different approaches. If the density has been approximated by the Delaunay approach, the underlying Delaunay triangulation provides a quick rendering method utilizing hardware-supported acceleration. If more intuitive visual presentation is needed, Blobby shading meets the demand. Since both the image-based approach and the hybrid approach maintain a raster, the raster can be rendered immediately by redirecting it into the "frame buffer." These methods are to be looked at in more detail in the following sections.

4.1.1 Direct Shading

Once density has been assigned to each point, we have reduced the problem to height interpolation and terrain visualization. A simple way to do this is to use the piecewise linear interpolant induced by the Delaunay triangulation. Considering the density value of each vertex as intensity, we render the *polygonal terrain* using Gouroud shading. This is especially easy with OpenGL. Note that

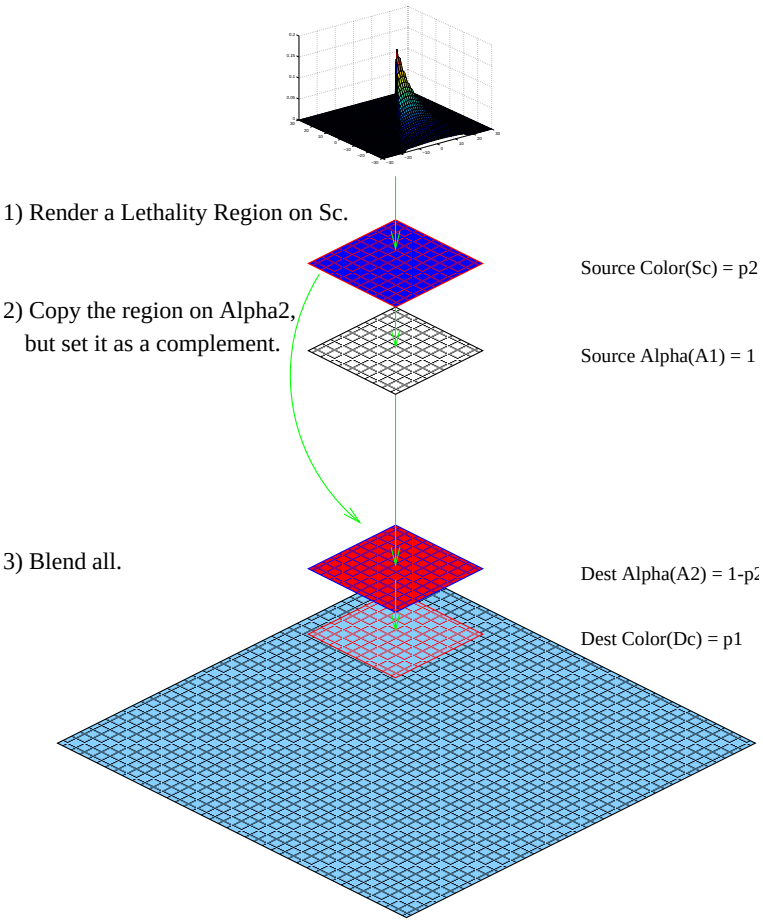
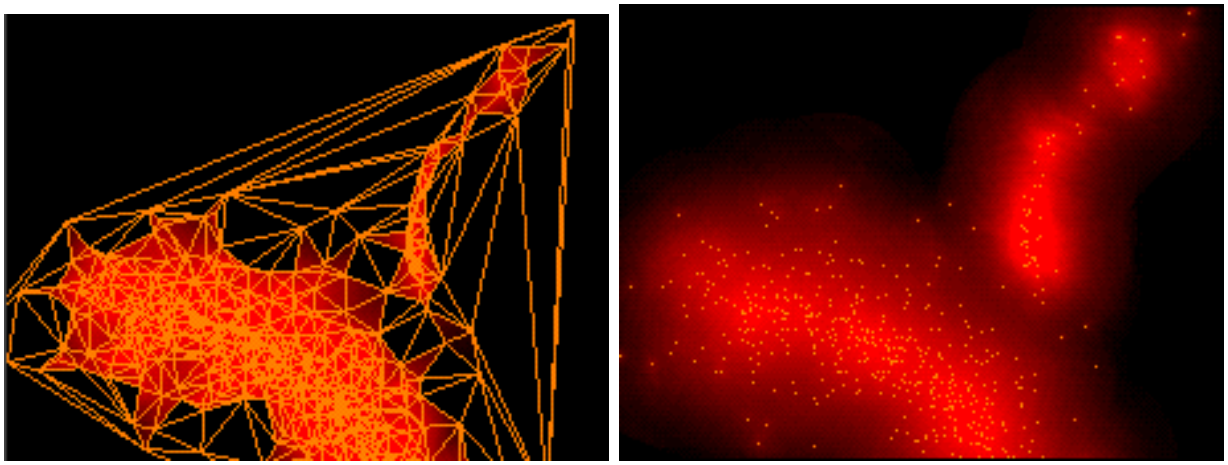


Figure 5: *One Step in Lethality Computation Using Image-Based Approach. We interpret the destination color as probability p_1 with a alpha value $1 - p_2$ and the source color as probability p_2 with a alpha value 1. Then the OpenGL blending function computes $p_1 + p_2 - p_1p_2$.*

good OpenGL implementations exploit available hardware for this, and the approach is suitable for real-time. This method, however, cannot render density outside the triangulation. Moreover, the rendered image looks polygonal, especially when there are only a few points.

4.1.2 Blobby Shading

Another alternative to the visualization of density is to render by *blobby shading*. Blobby shading was motivated by molecule visualization, where the electrons in an atom are represented as a density function [Blinn 1982]. Simplifying, the density distribution is obtained by summing the contribution from each atom separately:



(a) Direct shading (with the Delaunay triangulation)

(b) Blobby Shading

Figure 6: *Comparison of Density Visualization. The left figure shows the direct shading method using the underlying Delaunay triangulation as a polygonal terrain. Here the density value determines the height of the terrain. The right figure shows the blobby shading method by considering the density value as a radius of an atom in Equation 6.*

$$D(x, y, z) = \sum_i b_i \exp(-a_i r_i) \tag{6}$$

where a_i is the radius parameter of atom i, b_i is a "blobbiness" parameter for atom i, and r_i is the distance from the point (x, y, z) to the center of atom i.

We apply blobby shading by considering the density value of each point as a radius parameter of each atom. Using the Equation (6) for every pixel in a screen window, we compute its blobby sum from all points based on the distance between the pixel and each point.

The brute-force implementation of this approach is much more expensive than the triangle-based direct shading, but there are many possibilities to increase efficiency. One possibility is a hierarchical decomposition used in N-body algorithms [Blelloch and Narlikar 1997]. Moreover, the computation is inherently parallel. Improvements are also possible by applying interpolation techniques at the raster level. Finally, when the density is computed by the image-based approach or hybrid approach

using the Equation (6) as a local density function, its visualization is immediately available. All that is needed is to dump the resulting frame buffer onto the screen. Thus when the primary goal of density computation is only its visualization, the hybrid approach is the most flexible and efficient approach. The computation and visualization time of the hybrid approach is significantly faster than the brute-force implementation of blobby shading.

Blobby shading provides a highly intuitive presentation of the density distribution. Moreover, it shows pleasing boundaries and does not degrade in quality for a small number of points.

When density has been computed either by the image-based approach or by the hybrid approach, the computation result is stored in a raster buffer. By construction, the raster directly corresponds to the approximating result of Blobby shading. Therefore the rendering process simply dumps the raster buffer to the frame buffer of the graphics display. However, when the raster size is not sufficiently big compared to the frame buffer size, the result of the computation is inaccurate, and there could be visual artifacts similar to aliasing. In this case, one might be able to avoid the problem by zooming in the *region of interest* and reducing the required raster size.

4.2 Lethality Assessment Visualization

Since the result of lethality assessment computation is readily available at the frame buffer, we can directly dump it onto the screen to visualize it; see Figure 7-(c). Generally in a graphics work station, the dumping process is very fast because it is facilitated by Direct Memory Access (DMA).

4.3 Clustering Visualization

4.3.1 Boundary Detection in the Human Visual System

Researchers from both the cognitive psychology and scientific visualization domains have found various preattentive features that can be used to assist in boundary detection in the human visual system, and such features include hue, form (shape), and intensity [Healey 1999]. Callaghan reported that the visual system seems to prioritize features in order of importance. In his feature hierarchy, intensity is more important than hue to the low-level visual system during boundary iden-

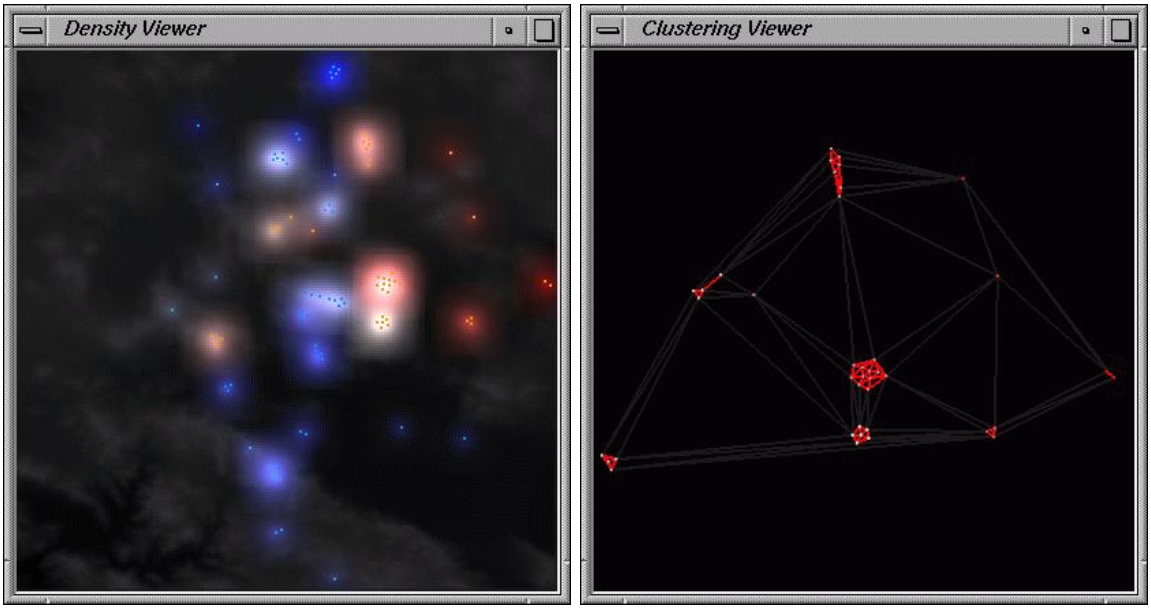
tification, and hue is more important than form [Callaghan 1984; Callaghan 1989; Healey 1999]. Moreover, it turns out that the feature hierarchy still applies to a dynamic environment where a real time sequence of frames is displayed at ten frames per second. We exploit this fact for clustering visualization.

4.3.2 HSV Scheme

The clustering visualization is overlayed on top of density visualization. Using the Hue, Saturation, Value (HSV) color scheme, we prioritize the clustering visualization as follows:

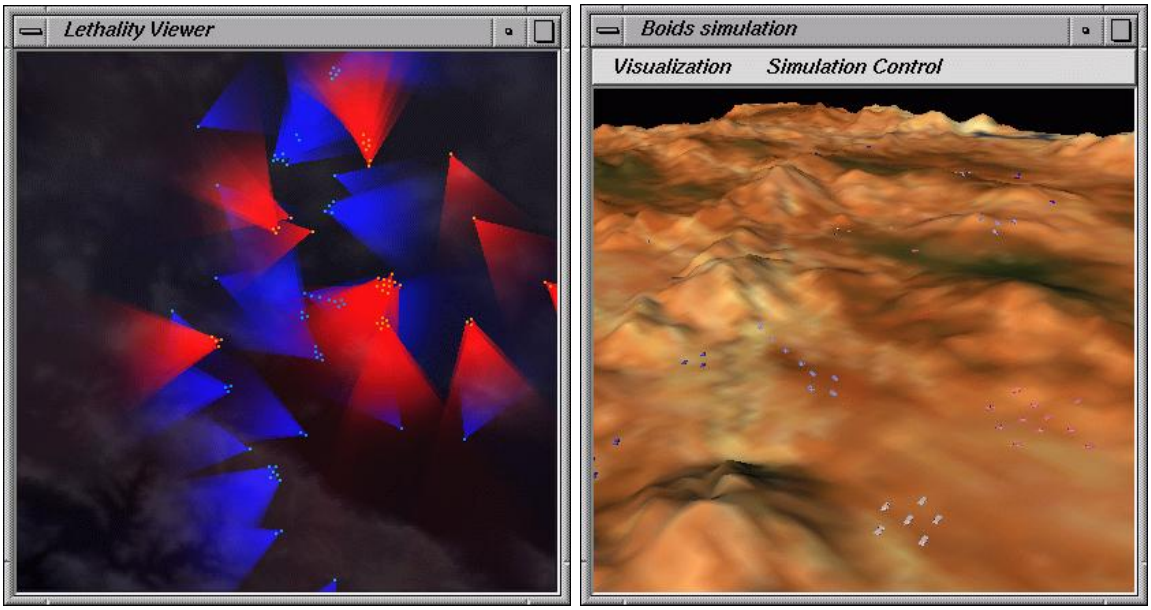
- Assign different Hues (H), $H = \frac{2}{3}$ (blue) for friendly platforms and $H = 0$ (red) for hostile platforms.
- Assign different Saturations (S) to different clusters.
- Value (V, pixel intensity) at position p is determined by its density level.

Thus, by different V we can identify the boundary of platforms, and by different H we can differentiate between the opposing platforms, and finally by different S we can further differentiate the boundary of opposing platforms; see Figure 7-(a).



(a) Density and Clustering Visualization

(b) Underlying Delaunay Triangulation



(c) Lethality Visualization

(d) Overall Battlefield View

Figure 7: *Battlefield Visualization. (a) shows the results of the density and clustering visualization. The blue platforms represent friendly forces, whereas the red platforms represent enemy forces. (b) shows the visualization of the "edge cutting" technique discussed in Section 3.3.2. The dark edge represents a disconnected edge in the Delaunay triangulation. Here the clustering computation is performed only for red forces. (c) shows the result of the lethality visualization. (d) shows the overall 3D view of the underlying battlefield where the computations (a), (b), and (c) are performed.*

5 Performance Results

5.1 Implementation Workbench

The experiment was carried out on 32 SGI 250 MHZ IP27 processors with 16G memory. An Infinite Reality 2E graphics board was used for fast polygon rendering for visualization and geometry computation. No parallel computation capability was used during these computations. Most components of the program were coded in C++ and compiled by MIPSpro C++ compiler (CC). In order to provide a convenient 3D interaction (GUI) to an end user and to easily manipulate 3D geometry objects in the Battlefield simulation, we used the Open Inventor graphics library. LEDA (Library of Efficient Data structures and Algorithms) version 3.8 was extensively used for various basic geometric data structures. In particular, the POINT SET data structure was extensively used to implement the Delaunay triangulation.

5.2 Various Computational Results

The experimental setting for our battlefield simulation is as follows:

- Consider a square, 10 miles wide in each direction. Points are uniformly distributed within the square. We vary the number of points from 100 to 400. Each point is assigned a randomly chosen speed from two groups of uniform distributions.

- Each point keeps moving by following the Boids animation rules [Reynolds 1987], and its maximum speed is 45 mph[2]

The performance results of a typical simulation are given in Table 1. In this simulation, we achieve interactive rates for density, cluster, and lethality computations. Each Boid character, in this simulation a tank, is composed of 360 polygons, and the terrain data is composed of 80 K polygons as shown in Fig. 5.2. For density computation, we use the hybrid approach[3] explained

[2]We took the example of M1 Abrams tank whose maximum speed is 45 mph.

[3]For the performance results of density computation using the geometry-based approach, we refer the reader to see [Hoffmann et al. 1998].

Table 1: Simulation Performance Result

NUM	POLY	FULLREND	BOXREND	DEN	LETHAL	CLUSTER
100	106	12.4	31.0	0.0500	0.0264	0.0060
200	132	7.4	15.2	0.0860	0.0847	0.0123
300	158	5.1	10.0	0.1218	0.1436	0.0190
400	184	4.0	7.3	0.1576	0.2014	0.0256

Table 2: *Performance Results. NUM denotes the number of Boid platforms involved in the simulation. POLY denotes the number ($\times 10^3$) of polygons to be rendered in each frame. FULLREND denotes the time (frames/sec) to render a full scene including Boids and terrain in the simulation. BOXREND denotes the time (frames/sec) to render a scene in a bounding box mode. DEN denotes the time (sec) to compute density distribution. LETHAL denotes the time (sec) to assess lethality. CLUSTER denotes the time (sec) to compute clustering.*

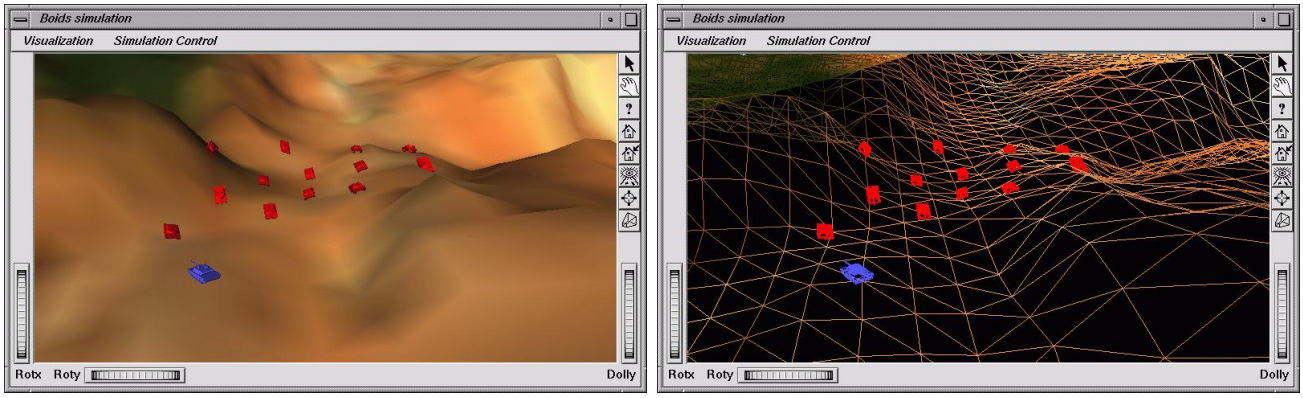
in Sec. 3.2.3, and the raster window size used in density computation and lethality assessment was 400×400. Particularly in the case of density computation, since each local influence region is identical, after one of the local influence regions has been rendered once onto an offscreen buffer, its raster copy is used for the remaining regions instead of re-rendering them each time. However, this cannot be applied to lethality assessment because the threatening regions are not identical.

6 Conclusion

6.1 Summary

We have presented three visual tools (density, clustering, and lethality assessment) to enhance the situation awareness of a military commander on the battlefield. A unique feature of this work is to extend the functionality of battlefield visualization packages so that they provide the visual abstractions of a battle. Traditionally, they have been devoted to terrain visualization.

By utilizing the perceptual psychology aspects of the low level human visual system, we effectively



(a) Fully Rendered Image (b) Wire Frame Image

Figure 8: *A snapshot of battlefield simulation using the Boids rules. In these figures, friendly forces (Blue tanks) are under attack from enemy forces (Red tanks)*

incorporated visual tools into the battlefield visualization package. Gestalt perception and the preattentive features of the human visual system were the key aspects considered in the development of the tools. The theory of Gestalt perception helps explain the global situation of a battlefield. The preattentive features, such as intensity, hue, and saturation, further assist in fast searching and quick boundary detection used by the low level human visual system.

Once the visual tools have been computed, they should be visualized; this is the main goal of battlefield visualization. Exploiting the aforementioned perceptual psychology aspects, we rendered the visual tools in a computationally efficient and perceptually meaningful way. Gestalt perception played an important role in our density rendering, and the preattentive features and their hierarchy in the human visual system were exploited both in clustering and in density rendering.

6.2 Future Work

Most of the techniques and visual tools in this paper have been developed in two dimensions. The two-dimensional view ultimately assists in understanding a three-dimensional combat view in the application. In some situations, such a two-dimensional approach is sufficient or even more suitable than the three dimension counterpart. Nevertheless, the extension to three dimensions is inevitable,

simply because the human visual system perceives the world in three dimensions. This should offer more opportunities to exploit preattentive rendering and density critical attributes.

From the perceptual psychology point of view, a three-dimensional extension to our application means that we need to consider three- dimensional preattentive features. Some recent works on oceanographic and atmospheric visualization have successfully incorporated such three- dimensional preattentive features as height and texture, in what they call "pexel (perceptual texture element)," in their three-dimensional visualization application [Healey 1999]. However, it is not known whether or not, and to what degree, three -dimensional features might interfere with existing two- dimensional features and thereby limit comprehension of all features present.

References

Blanchard, H. 1996. Dominant battlespace awareness. Posted on the C4I-Pro Archive, February.

Blelloch, G., and Narlikar, G. 1997. A practical comparison of n-body algorithms. In *Parallel Algorithms*, Series in Discrete Mathematics and Theoretical Computer Science. American Mathematical Society.

Blinn, J. 1982. A generalization of algebraic surface drawing. *ACM Transactions on Graphics 1*, 3 (July), 235–256.

Callaghan, T. 1984. Dimensional interaction of hue and brightness in preattentive field segmentation. *Perception and Psychophysics 36*, 1, 25–34.

Callaghan, T. 1989. Interference and domination in texture segregation: Hue, geometric form, and line orientation. *Perception and Psychophysics 46*, 4, 299–311.

Callaghan, T. 1990. Interference and dominance in texture segregation. *Visual Search*, 81–87.

Charikar, M., Chekuri, C., Feder, T., and Motwani, R. 1997. Incremental clustering and dynamic information retrieval. In *Proceedings of 29th Annual ACM Symposium on Theory of Computing*, 626–635.

Cohen, J., Varshney, A., Manocha, D., Turk, G., Weber, H., Agarwal, P., Brooks, Jr., F. P., and Wright, W. V. 1996. Simplification envelopes. In *Proceedings of SIGGRAPH '96 (New Orleans, Louisiana, August 4–9, 1996)*, ACM Press, Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, 119–128.

de Berg, M., van Kreveld, M., Overmars, M., and Schwarzkopf, O. 1997. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, Berlin.

Devillers, O. 1992. Robust and efficient implementation of the Delaunay tree. Rapport de recherche 1619, INRIA.

Dickerson, M., and Drysdale, R. 1990. Fixed-radius near neighbors search algorithms for points and segments. *Information Processing Letters 35* (August), 269–273.

Duchaineau, M. A., Wolinsky, M., Sigeti, D. E., Miller, M. C., Aldrich, C., and Mineev-Weinstein, M. B. 1997. ROAMing terrain: real-time optimally adapting meshes. In *IEEE Visualization*, 81–88.

Durbin, J., II, J. S., Colbert, B., Crowe, J., King, R., King, T., Scannell, C., Wartell, Z., and Welsh, T. 1998. Battlefield visualization on the responsive workbench. *Proceedings IEEE Visualization '98* (October), 463–466.

Durbin, J., Julier, S., Colbert, B., Crowe, J., Doyle, B., King, R., King, T., Scannell, C., Wartell, Z., and Welsh, T. 1998. Making information overload work: The Dragon software system on a virtual reality responsive workbench. *Proceedings of the 1998 SPIE AeroSense Conference 3393* (April), 96–107.

Emmerman, P., Walrath, J., and Winkler, R. 1998. Making complex, multidimensional battlefield information intuitive. In *Proceedings of the 21st Army Science Conference*.

Endsley, M. 1988. Design and evaluation for situation awareness enhancement. In *Proceedings of Human Factors Society 32nd Annual Meeting*, vol. 1.

Healey, C. 1999. Perceptual techniques for scientific visualization. SIGGRAPH '99 Course #6, Aug.

Hoff, K., Culver, T., Keyser, J., Lin, M., and Manocha, D. 1999. Fast computation of generalized Voronoi diagrams using graphics hardware. In *Proceedings of SIGGRAPH '99*.

Hoffman, J. E. 1980. Interaction between global and local levels of a form. *Journal of Experimental Psychology: Human Perception and Performance 6*, 222–234.

Hoffmann, C., Kim, Y., Winkler, R., Walrath, J., and Emmerman, P. 1998. Visualization for situation awareness. In *Workshop on New Paradigms on Information Visualization and Manipulation*.

Hoppe, H. 1996. Progressive meshes. In *Proceedings of SIGGRAPH '96*, 99–108.

Hoppe, H. 1997. View-dependent refinement of progressive meshes. In *Proceedings of SIGGRAPH '97*, ACM Press, Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, 189–198.

Human, I. F., and Cognition, M., 2000. Tactile situation awareness system. http://www.coginst.uwf.edu/tsas/main.html, Feb.

Inaba, M., Katoh, N., and Imai, H. 1994. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of 10th Annual ACM Symposium on Computational Geometry*, 332–339.

Kass, S., Hershler, D., and Companion, M. 1990. Are they shooting at me: An approach to training situation awareness. In *Proceedings of Human Factors Society 34th Annual Meeting*.

Keil, J., and Gutwin, C. 1989. The Delaunay triangulation closely approximates the complete Euclidean graph. In *Proceedings of 1st Workshop Algorithms Data Structures*, Springer-Verlag, vol. 382 of *Lecture Notes Computer Science*, 47–56.

Lindstrom, P., Koller, D., Ribarsky, W., Hodges, L., Faust, N., and Turner, G. 1996. Real-time continuous level of detail rendering of height fields. *Proceedings of SIGGRAPH'96*, 109–118.

MAK Technologies. *MÄK Plan View Display User's Guide*, 1.2 ed. 185 Alewife Brook Parkway Cambridge, Massachusetts USA.

Mehlhorn, K., and Näher, S. 1998. *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, New York.

Mueller, K., Müller, T., II, J. S., Crawfis, R., Shareef, N., and Yagel, R. 1998. Splatting errors and antialiasing. *IEEE Transactions on Visualization and Computer Graphics 4*, 2 (April-June).

Navon, D. 1977. Forest before trees: The precedence of global features in visual perception. *Cognitive Psychology 9*, 353–383.

Okun, N., 1998. Major historical naval armor penetration formulae. http://www.combinedfleet.com/formula.htm, June.

Pajarola, R. B. 1998. Large scale terrain visualization using the restricted quadtree triangulation. In *IEEE Visualization '98*, D. Ebert, H. Hagen, and H. Rushmeier, Eds., 19–26.

Resource Coordinator, O., 2000. References and readings related to situation awareness. http://www.ott.navy.mil/, April. OTTAdmin@navair.navy.mil.

Reynolds, C. 1987. Flock, herds, and schools: A distributed behavioral model. In *ACM SIGGRAPH '87 Conference Proceedings*, vol. 21, 25–34.

Rosenbloom, P., Johnson, W., Jones, R., Koss, F., Laird, J., Lehman, J., Rubinoff, R., Schwamb, K., and Tambe, M. 1994. Intelligent automated agents for tactical air simulation: A progress report. In *Proceedings of the Fourth Conference on Computer Generated Forces and Behavioral Representation*.

Rossignac, J. R., and Borrel, P. 1992. Multi-resolution 3D approximations for rendering complex scenes. Technical Report RC 17697 (#77951), IBM Research Division, T. J. Watson Research Center, Yorktown Heights, New York.

Schroeder, W. J., Zarge, J. A., and Lorensen, W. E. 1992. Decimation of triangle meshes. *Computer Graphics 26*, 2, 65–70. Proceedings SIGGRAPH '92.

Shewchuk, J. R. 1996. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *Applied Computational Geometry: Towards Geometric Engineering*, M. C. Lin and D. Manocha, Eds., vol. 1148 of *Lecture Notes in Computer Science*. Springer-Verlag, May, 203–222. From the First ACM Workshop on Applied Computational Geometry.

Stytz, M., Block, E., and Soltz, B. 1993. Providing situation awareness assistance to users of large-scale, dynamic, complex environments. *Presence 2*, 4.

Sukthankar, R., 1997. Situation awareness for tactical driving. Ph.D. thesis, January.

U.S. Army Combined Arms Center, A.-S., 1995. Battlefield visualization concept. Department of the Army TRADOC Pamphlet 525-70, October.

Walrath, J., Winkler, R., Emmerman, P., Hoffmann, C., and Kim, Y. 2000. Visualization technique for improved situation awareness. In *Society for Imaging Science and Technology (IS&T)International Society for Optical Engineering (SPIE) conference on Human Vision and Electronic Imaging V*.

Wilson, J. 2000. *College Physics*. Prentice Hall.