3-10-2010

Modeling Computer Communication Networks in a Realistic 3D Environment

Charles R. Rowell Jr.

Modeling Computer Communication Networks in a Realistic 3D Environment

THESIS

Charles R. Rowell, Jr., Captain, USAF

AFIT/GCE/ENG/10-05

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GCE/ENG/10-05

Modeling Computer Communication Networks in a Realistic 3D Environment

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Engineering

Charles R. Rowell, Jr., B.S.E.E.

Captain, USAF

March 2010

AFIT/GCE/ENG/10-05

Modeling Computer Communication Networks in a Realistic 3D Environment

Charles R. Rowell, Jr., B.S.E.E.
Captain, USAF

Approved:

/signed/	12 Mar 2009
Lt Col Stuart Kurkowski, PhD (Chairman)	date
/signed/	12 Mar 2009
Dr. Ken Hopkinson (Member)	date
/signed/	12 Mar 2009
Maj Ryan Thomas, PhD (Member)	date

AFIT/GCE/ENG/10-05

Abstract

Communication network simulations have typically been visualized in the past through 2D representations, but this is insufficient for battlefield network scenarios. Visual representations of battlefield networks greatly benefit from 3D visualization due to its ability to retain asset location. This research investigates the feasibility of modeling a typical battlefield communication network in a realistic 3D manner and discusses the effects of doing so. The result is an open source, 3D network visualization tool that can create highly intuitive connected battlefield scenes, enabling the user to quickly comprehend network state. It highlights mobile assets, packet movement, and node connectivity while allowing the viewer to interact with the scene.

Acknowledgements

Firstly, I would like to thank my thesis advisor Lt Col Stuart Kurkowski for his encouragement and support throughout the past year. I am forever indebted to Chris Sheffield from University of Dayton for his computer programming abilities, without which I would have been lost. Many thanks to Maj Mark Silvius for some last-minute corrections and guidance to keep me on track for graduation. Lastly, I would like to thank my wife, who motivated me throughout the school year and kept me sane during the thesis writing process.

Charles R. Rowell, Jr.

Table of Contents

Page

Page

List of Figures

List of Tables

List of Abbreviations

Abbreviation		Page
2D	two-dimensional	1
3D	three-dimensional	1
Nam	Network Animator	10
NS-2	Network Simulator-2	10
iNSpect	interactive NS-2 protocol and environment confirmation tool	12
OPNET	Optimized Network Evaluation Tool	13
NetViz	Network Visualization	14
HUD	Heads-Up Display	17
OSG	OpenSceneGraph	24
VTK	The Visualization Toolkit	25

Modeling Computer Communication Networks in a Realistic 3D Environment

I. Introduction

The visual representation of battlefield networks is increasingly important as the battlefield becomes more connected. Nearly every battlefield asset from soldiers to satellites contains a radio transceiver used to communicate throughout the operational environment. Network simulation is a cost effective method to determine problematic issues prior to deployment, reducing the need for building a test bed of computers, routers, and switches to evaluate network performance. These network simulations produce a large amount of data which contain all network events that occurred in the simulation. It is often desirable to display this data visually, in order to capitalize on the unique capabilities of the human visual system which "excels at processing images and recognizing patterns" [36]. In the past, these networks were simple enough to be visualized in a two-dimensional (2D) manner, but due to increased network connectivity in the battlefield, a three-dimensional (3D) visualization is more appropriate.

In a typical battlefield network scenario, node location is an important aspect of the network topology. Battlefield scenarios greatly depend on a 3D representation, since variation in elevation and the existence of obstacles such as mountains often have a significant effect on successful communication. Node location determines line of sight to the intended destination, and the existence of obstacles between them may impede communication. Additionally, 2D visualization methods cannot truly capture the movement of a variety of assets through an ever-changing terrain. One major thing people need from visualization today is the existence of custom solutions to meet their real-world needs [10], which in this case is a realistic 3D network visualization tool

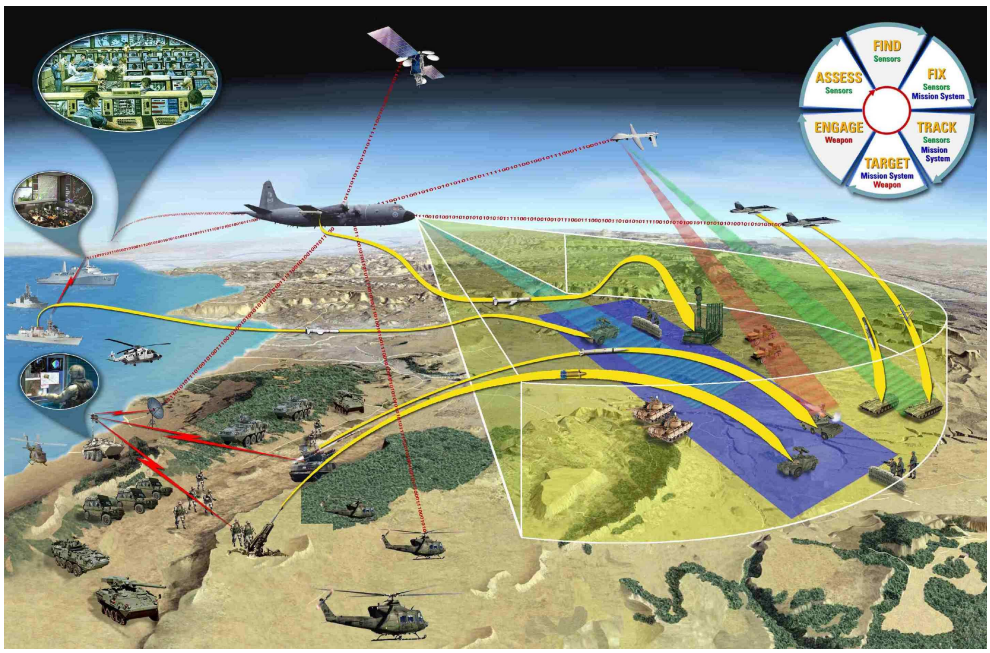


Figure 1: Artist's rendition of the connected battlefield [20]

which would allow the accurate portrayal of battlefield communication in an intuitive manner.

This research investigates the feasibility of creating a 3D network visualization that can accurately portray scenarios typically encountered on the battlefield. Current network visualizations typically display homogeneous networks well in 2D, but are lacking when displaying a realistic battlefield scenario such as the one shown in Figure 1. This artist's rendition of the battlefield is representative of the interoperability between multiple types of communication assets, across a variety of locations and elevations. 2D network visualizations cannot preserve the locations of a true battlefield scene, which drives the need for a 3D visualization solution.

One problem with creating a new visualization is quantifying its effectiveness. Information visualization is a very subjective research area, and no universal evaluation methods exist. To address this issue, Chapter II explores common themes in visualization research, including the use of affective computing, how to capitalize on human perceptual abilities, and the use of animation to trigger cognitive response in

users. This provides the basis for establishing criteria to create and evaluate the new 3D visualization tool as objectively as possible.

Building a new 3D visualization tool requires an understanding of current solutions in the network visualization field. Numerous network visualization tools are explored in Chapter II, separated into 2D and 3D categories. These include the Network Animator (NAM), the Optimized Network Evaluation Tool (OPNET), the interactive Network Simulator-2 (NS-2) protocol and environment confirmation tool (iNSpect), the Network Visualization (NetViz) toolkit, and Huginn. Each of these tools has distinguishing visualization characteristics, which can be leveraged for the creation of a new tool. Understanding the strengths and drawbacks of each tool results in a better final product.

The new 3D visualization tool created for this research is built using the OpenSceneGraph (OSG) 3D graphics toolkit. Of the investigated 3D toolkits—OSG, the Visualization Toolkit (VTK), and Autodesk 3ds Max—OSG proved to be the most suitable to the task. Using available tutorials as a starting point, a 3D network visualization tool was built that is both flexible and effective at displaying network simulation data.

This research provides three major contributions to the network visualization field. Firstly, the created 3D visualization tool is an excellent communication aid, which enables efficient and effective communication. Secondly, different packet animation schemes are investigated by this research, which attempts to find an appropriate balance between visual efficiency and simulation accuracy. Lastly, this research provides an open source, 3D network visualization tool that can be tailored to meet a variety of situations.

II. Information Visualization and Computer Networks

Successfully modeling a computer communication network in 3D requires an understanding of visualization techniques as well as a knowledge of network visualization currently in use. This chapter explores a number of topics in these two areas, starting with general visualization concepts.

The human visual system provides an unparalleled capability to processing and understanding data [35]. Effective visualization techniques exploit the human visual system by pushing as much data as possible to the eye. Creating a useful visualization requires an understanding of how to capitalize on the visual system through the use of animation as well as the fourth dimension, time. Finally, the concept of visualization applied directly to computer networks is discussed, and problems with transitioning from 2D to 3D are briefly addressed.

Creating a successful network visualization requires an understanding of existing tools. The major visualization solutions in use today are discussed, for both 2D and 3D, along with some previous research in the realm of 3D simulation and visualization. By examining the strengths and weaknesses of each visualization, lessons can be learned that aid in creating an effective new visualization.

2.1 Visualization Concepts

2.1.1 Affective Computing. In 1995, Rosalind Picard first coined the phrase *affective computing* with regard to human-computer interaction (HCI), which seeks to determine the emotional impacts of HCI while accurately conveying information [27]. Her research investigates how computers affect users' emotions and seeks to identify ways to reduce user frustration. Many people are confused and frustrated by technology, and in many cases turn hostile towards the computer itself. These violent outbreaks achieve nothing productive other than the release of user tension, so it is important to examine the source of this frustration and alleviate it if possible.

In many ways, this type of research is directly applicable to computer network visualization. In order to reduce user frustration when dealing with network visual-

izations, it is important to determine which aspects of computer interaction are not ideal, and hopefully devise a way to alleviate them. By creating a visualization interface that is intuitive, user satisfaction can be increased and frustration decreased. Due to the complex nature of the data portrayed in a network visualization, this is an important factor to consider when designing a visualization for computer communication networks. In a battlefield scenario in particular, there can be an extraordinarily large amount of communication occurring between numerous types of assets, and this can be overwhelming to users that are not accustomed to seeing this type of data. It is therefore critical to display the information in a way that is easily understood by the average user, hopefully making the analysis of the computer network both more intuitive and reducing the time needed to perform the analysis.

2.1.2 Human Perceptual Abilities. There has been a good deal of research performed in the field of affective computing to determine which types of visual stimuli have the greatest effect. Graphic representations in particular, such as pie charts or illustrated diagrams, allow people to easily process information visually in a quicker and more intuitive fashion than by meticulously examining the original data [19]. 3D interactive animation is very useful in shifting cognitive data load to the more capable human perceptual system [29]. The cognitive analysis that occurs during the visual perception process occurs very rapidly without conscious effort [36]. Visual representations of data allow people to quickly draw more accurate conclusions about the overall significance of the data.

It is worth noting that overuse of visual representations or the use of misleading visualizations can result in misinterpretation of the data by the viewing public [19]. It is therefore critical to understand how to effectively use visual representations to help people process information without adding further complication. By carefully constructing a visualization based on what has been shown to be effective, a network visualization can be created that is highly intuitive without leading to confusion.

2.1.3 Animated Visualization. As discussed in [19], [29], and [36], animation has a unique cognitive effect when combined with information visualization. Animated visualization in this case means the addition of time-dependent motion to the information visualization, either in an autonomous or user-controlled fashion. In particular, adding time-dependent motion to an existing 3D visualization provides an additional dimension, effectively creating a 4D graph. This enables increased realism and provides the opportunity for greater user understanding. Furthermore, these motions, when added to previously static visualizations, produce significant cognitive effects in users which allow them to more accurately process data [19].

Animated visualization has been shown to be useful in three primary ways [19], each of which can be directly applied to the successful portrayal of computer communication networks. The first of these is when a user needs to identify the data points that change in a significant manner. During network simulations, it is often desirable to identify which nodes experience the most traffic and which are largely idle. Animated visualization helps the user quickly and accurately identify nodes of interest. The second primary use for animated visualization is when the user needs to determine at what point in time a significant event takes place. In a computer network scenario, it is often necessary to determine at which point in time a network event occurred, such as a node failure or significant packet loss. Animating the network simulation in a visual manner allows the user to quickly identify key points in time that warrant further inspection. For example, examining packet flow in the network before a network failure can be crucial to identifying the possible cause. The third and most fundamental advantage that animated visualization provides is the feeling of immersion, which allows for more intuitive data analysis. Displaying a computer network in an animated fashion allows the user to picture the network in a realistic manner and see how data flows throughout the network, which cannot be experienced with a static display. Additionally, allowing the user to control the viewing angle, zoom in and out, and control the flow of time results in a more immersive visualization, leading to a more thorough grasp of the network.

Animated visualization exploits the unique capabilities of the human perceptual system, which stimulates the recognition of patterns in data and structure in information [29]. 3D visualization, especially when animated, allows the user to process more information, doing so more quickly, with more comprehension [36]. This animation aspect is particularly important, as it provides a greater degree of realism to the visualization. One problem that arises when visualizing a network in 3D is that of *node occlusion*, in which one object blocks the view of another. Objects in a 3D scene can occupy the background or foreground, meaning objects can sometimes overlap or block the view of other objects. However, by animating the representation to slowly rotate, or better yet, allowing the user to rotate, pan and zoom, this problem is solved, while also providing greater clarity of the interspatial relationships between scene components. Viewing a 3D object or scene from multiple views is critical to understanding what is seen [13]. The user should be free to rotate, pan and zoom throughout the 3D scene and investigate the distance between the different objects and determine the relative positions of each 3D scene component.

2.1.4 Visualization and Computer Networks. Computer communication networks consist of many communicating nodes, each transmitting and receiving signals to and from other nodes in the network. These networks are typically represented in a 2D manner, with lines drawn between communicating nodes to indicate network connectivity, which is essentially a graph. In a network graph, communicating nodes are represented by vertices and communication links are represented by vertices. This allows the user to see how data can flow from one device to another by determining available routes throughout the network. The user can also ensure that all nodes are reachable via communication paths by visually verifying network connectivity.

Since network maps are essentially graphs, metrics for graph aesthetics can often be applied to them. For example, minimizing the number of edge crossings and maximizing symmetry are two ways to enhance readability [17] [6], as demonstrated in Figure 2. In this figure, it can be seen that the graph is clarified from the random

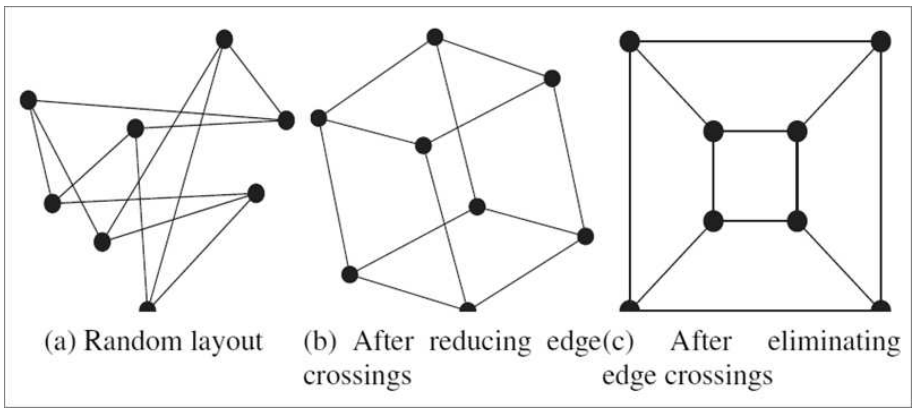


Figure 2: Readability is increased by reducing edge crossings and maximizing symmetry—the random layout of (a) is improved by reducing edge crossings (b) and is most readable after fully removing edge crossings and maximizing symmetry (c) [6]

layout of (a) by reducing the number of edge crossings, resulting in the layout seen in (b). The graph is further clarified in (c) by completely removing edge crossings and maximizing symmetry in the graph. By arranging the nodes in this manner, it becomes very easy to see how network traffic flows throughout the network.

Rearranging nodes to reduce edge crossings and maximize symmetry works well for existing 2D network visualizations, because node location is typically not important. For example, when visualizing the average corporate network, it is not especially important how the nodes are represented, because they are all hard-wired. Since nodes are connected by a wired connection, they each perform equally well regardless of node placement. However, this cannot be done in a 3D battlefield scenario where node location is vital to the clarity of the scene. If a tank asset is communicating with a nearby plane asset, the two cannot be simply relocated for the purpose of visual clarity, because their interspatial relationship carries vital information.

To alleviate this problem of overlapping communication links, which can quickly create a dense web of network paths, the user must have control over the scene to clarify links. By allowing the user to pan, rotate, and zoom around the scene, network links can be clarified by viewing the scene from a different angle, while maintaining node locations. In this manner, the geographic positions of battlefield assets can be

retained while still allowing the user to clarify communication links between two or more nodes.

In a military battlefield network scenario, there are many disparate types of communicating nodes, such as airplanes, helicopters, tanks, aircraft carriers, satellites and ground stations, among others. Each of these has a specific manner in which they can move and communicate with other assets. For example, if a helicopter needs to communicate with a ground station that is currently blocked by a hill or ridge, the helicopter can change its elevation to rise above the obstacle. On the other hand, a land-based asset such as a tank does not have this same capability. It is important to maintain this realism when creating the 3D scene, to help establish *visual credibility* of the visualization. A high degree of visual credibility includes using realistic assets, realistic terrain, and animation paths that are typical of those assets. By ensuring that assets move in a similar manner as their real-life counterparts, the result is a scene that is more intuitive because assets behave in a manner that the user expects.

Adding an animation aspect allows the user to visually examine such factors as system throughput, packet loss, and network congestion as a function of time. This not only gives a better understanding of the network, but it also allows for the rapid identification of critical nodes and backbone network paths.

2.2 Current Network Simulation and Visualization

There are numerous network simulators and visualizations in use today, and they primarily fall into two distinct categories, two-dimensional (2D) and three-dimensional (3D). To successfully create an accurate simulation based 3D network visualization tool, it is imperative to examine the various solutions that currently exist. The following section delves into each category and exposes the major works in each respective field.

2.2.1 Two-Dimensional Network Simulation and Visualization. Of current network simulation visualization, the bulk of it falls in the realm of 2D visualization.

This stems from the fact that 2D is much simpler to configure than 3D and requires far fewer system resources. 2D is especially useful when node location is not a critical characteristic of the communication network. This allows nodes to be rearranged as necessary to enhance readability without affecting the performance of the network. This section discusses the major 2D visualizations in use today and lists the distinguishing characteristics of each. These include the Network Animator (Nam), the interactive Network Simulator-2 (NS-2) protocol and environment confirmation tool (iNSpect), the Optimized Network Evaluation Tool (OPNET), and the Network Visualization (NetViz) toolkit. Understanding the strengths and weaknesses of each of these is instrumental to creating an effective visualization. By capitalizing on the strengths and avoiding the weaknesses, a more robust visualization can be realized.

2.2.1.1 Nam. The Nam visualization produces a visual representation of network events that occurred during an NS-2 simulation [9]. NS-2 produces a trace file as it executes a specified network scenario, which can then be later used by Nam to aid the user in understanding network events. The user specifies the desired scenario and network events in NS-2 and then can use Nam to animate these events in a visual manner. Nam allows for real-time packet animation, current queue size at each node, and graphs of various network parameters, adding significant capability to the NS-2 simulator. The user is free to adjust a time slider to any point in time in the simulation to get a better look at specific network events. A sample Nam network visualization is shown in Figure 3, which shows a simple five node network. In this scenario, packets are being sent from *node 0* to *node 3* (color coded in blue) and from *node 3* to *node 1* (color coded in red), with *node 2* forwarding packets as necessary to support both of these network flows.

Nam was originally created to support wired network visualization, but has been extended to also display wireless network activity. In a wired scenario, Nam uses direct lines from node to node to indicate communication and displays rectangles that travel from source to destination to indicate data flow. For wireless connections,

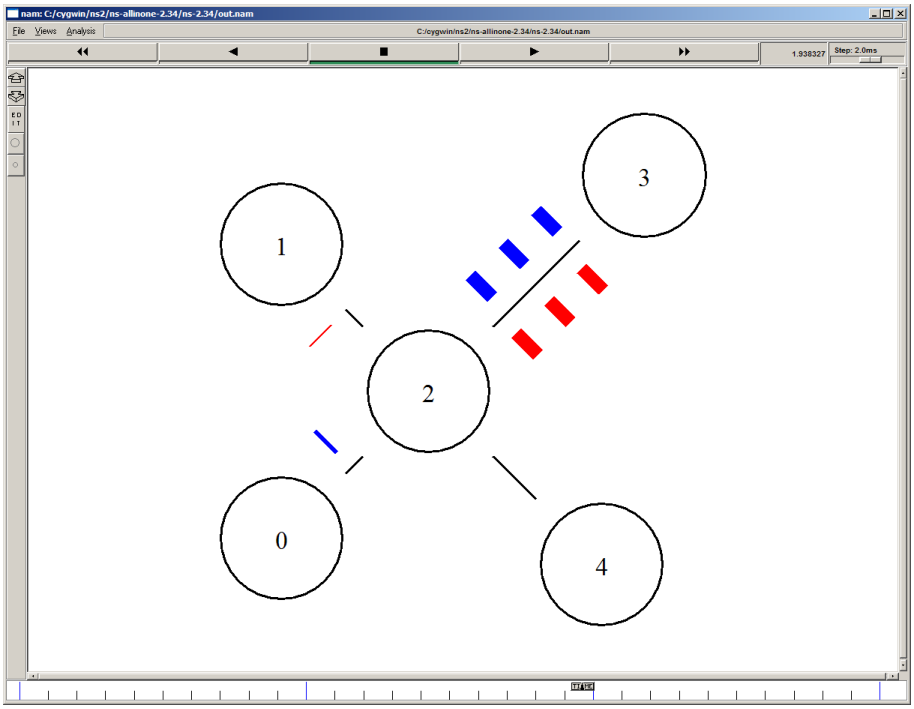


Figure 3: Sample wired network visualization in Nam showing transfer of data packets from *node 0* to *node 3* and from *node 3* to *node 1*, using *node 2* as an intermediary node

however, Nam uses outward expanding broadcast rings, which reminds the user that wireless transmission typically occur in all directions, more or less equally. An added capability of representing wireless communication in this manner is that it allows the user to visually inspect collision domains between two adjacent nodes, since each node is competing for the same physical medium. However, it can sometimes be difficult to determine which nodes are communicating when there are many overlapping rings displayed at once.

The open source nature of both NS-2 and Nam has made each a favorite in the research community. Without the need for expensive or restrictive software licenses, researchers are free to explore the capabilities of each without restriction. This contrasts from proprietary networks simulation visualization software, in which altering core functionality is not possible due to the closed source of the program code.

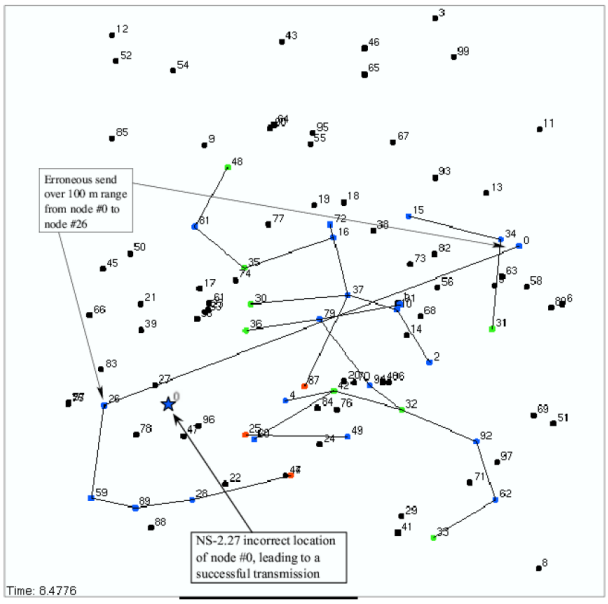


Figure 4: iNSpect visualization featuring color coded nodes: source (blue), successful destination (green), unsuccessful destination (red), and intermediate (yellow) [16]

2.2.1.2 iNSpect. The iNSpect visualization tool was created to display network trace data from NS-2 and address shortfalls of visualizing wireless networks in Nam [16]. iNSpect uses direct lines to indicate communication between nodes, similar to wired communication in Nam. Instead of displaying data flow as moving objects, iNSpect instead uses arrows between nodes to indicate in which direction communication is occurring. Additionally, nodes are color-coded so the user can quickly distinguish what role a node is currently performing. In the default color scheme, blue indicates source nodes, red or green indicates destination nodes (depending on whether the transmission is successful), and yellow indicates nodes on the path that are forwarding the message towards its destination. A sample iNSpect network visualization is shown in Figure 4, which shows network traffic in a mobile ad hoc network.

The iNSpect visualization provides benefits as well as drawbacks, as is the case with any solution. The arrows from source to destination make it much easier to follow a data packet along its path compared with Nam's broadcast rings, and the red and green destination nodes give the user a quick idea of whether most communi-

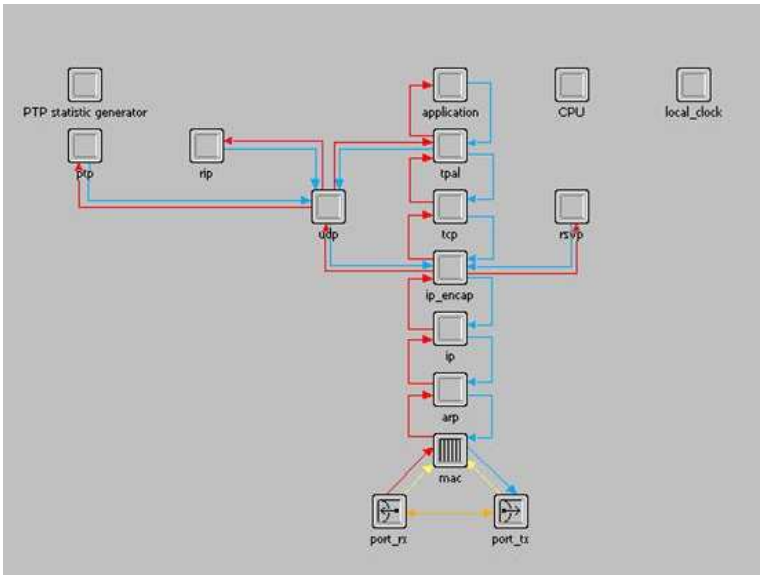


Figure 5: Sample network visualization in OPNET, in which network nodes are represented by generic boxes

cation attempts are successful. However, the program requires a more extensive NS-2 simulation to function properly, which almost doubles the trace file size. Another drawback is that the arrows are not as intuitively obvious that data is currently flowing from source to destination. A user unfamiliar with the program may interpret the arrows as potential communication links, which could be solved by animating packets between nodes as in wired communication.

2.2.1.3 OPNET. The OPNET network simulation suite provides a network event simulator as well as an associated visualization to display network events. OPNET represents network events in much the same way as Nam, with links representing wired connections and broadcast rings representing wireless connections. A sample OPNET visualization can be seen in Figure 5, which shows communication between a number of network nodes. These network nodes are represented by generic boxes, which makes distinguishing between them difficult.

OPNET is a powerful tool that allows the user to create and execute network scenarios, though its proprietary nature presents difficult challenges. Successfully modifying OPNET's visualization behavior requires extensive knowledge of the inter-

nal workings of the simulator, and it is easy to corrupt existing functionality when attempting to add alternate visualization methods. The proprietary nature of OPNET severely limits its use as a network visualization research platform since modifying its visualization features is very difficult due to the closed source nature of the program. OPNET is a very useful tool for network simulation research, but for exploring alternate network visualization techniques, other open source tools such as such as Nam or iNSpect are more accommodating.

2.2.1.4 NetViz. The NetViz toolkit was developed to address shortfalls of Nam, iNSpect, and OPNET, building a network visualization framework for both wired and wireless communication [2]. NetViz supports traditional network visualization features, such as packet animation, queue levels at each node, and dropped data packets in the network. Unlike other 2D visualization, NetViz features realistic assets instead of generic boxes. This can be seen in Figure 6, which depicts communication between aircraft, an unmanned aerial vehicle (UAV), a satellite, and other assets. This figure shows individual packets and supports display of transmission queues, as seen on the satellite node. A distinguishing characteristic of NetViz is the ability to examine the current status of any node as the visualization executes, allowing the user to see in real-time such statistics as packets sent, received or dropped, node position, and velocity. NetViz also explores the concept of force-directed layouts, in which the visualized network layout changes according to certain network characteristics [3]. For example, one such method is based on end-to-end delay, in which nodes that experience shorter end-to-end delays are pulled closer together and nodes with higher end-to-end delays are pushed further apart. This allows the user to gain insight into the delay of the network without fully analyzing each link individually.

Originally designed to support trace files from a simulation executed in NS-2, the toolkit was expanded to also support OPNET trace files as well, which are generated through the use of a third party plugin [2]. This was achieved through the development of a robust file parser that processes the trace files, extracting the

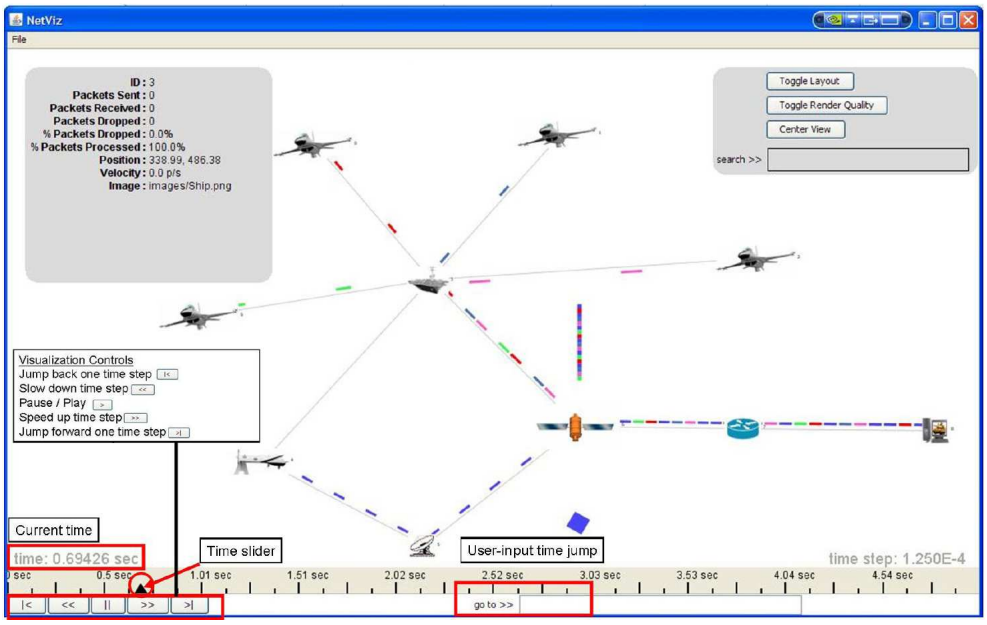


Figure 6: A NetViz visualization featuring aircraft, a UAV, a satellite, and an aircraft carrier [3]

critical information needed to run the visualization. The result is a visualization that is simulator-independent which can be used to analyze scenarios executed in either simulator.

2.2.2 Three-Dimensional Network Simulation and Visualization. There has been significant work done in the realm of 3D network simulation and visualization, though not quite to the extent of 2D visualization. Two primary research works stand out in particular: one investigates ad hoc network simulation using realistic 3D terrains [7], while the other renders a 2D network map in 3D, and uses the third dimension to display additional network information [30]. Additionally, the OPNET suite includes the ability to visualize network events in 3D, which shows that the commercial world is interested in a 3D networking solution. Each of these works is important to 3D network visualization, and lessons can be learned by examining their strengths and weaknesses.

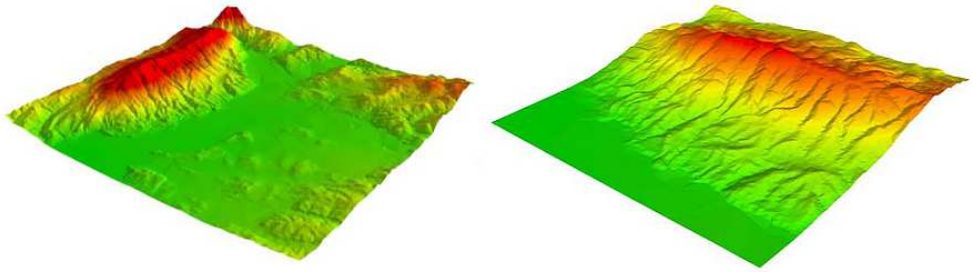


Figure 7: Realistic 3D terrains used in [7]: valley (left) and mountain range (right)

2.2.2.1 Network simulation with 3D terrains. Most wireless network simulators use an extremely idealized model known as the "flat earth" model, in which all radio signals propagate equally in all directions, with no regard for interfering objects [28]. In some situations, this model performs quite well at accurately reflecting real-world performance. However, in other situations such as battlefield communication or the deployment of an ad hoc network, use of the flat earth model is questionable due to real world terrain and features such as rocks and trees. These obstacles introduce interference and the natural terrain of the area creates hills and valleys that affect the ability of wireless signals to travel to their intended destination.

This terrain blocking dilemma has been explored in previous research, which shows the results of a typical flat earth network simulation compared with one executed with a realistic 3D terrain [7]. Developed as an extension for the NS-2 simulator, the model uses line-of-sight and a custom propagation algorithm to determine more accurate signal propagation ranges for a given terrain. Figure 7 shows two of the terrains used in the experiments: the left terrain is a valley created by nearby mountains and the right terrain is a mountain range. Using this model, the authors have shown that terrain can have a significant effect on throughput when compared with the idealistic flat earth model. By taking into account terrain features, it is possible to get a more accurate picture of the ad hoc communication network before actual deployment.

2.2.2.2 Huginn visualization. The Huginn visualization takes a 2D trace file from NS-2, models the network in 3D, and then uses the third dimension to display additional network characteristics [30]. This visualization uses the third dimension more for visual reasons than for real-world accuracy purposes. Created before NS-2 supported 3D coordinates, this network visualization uses the third dimension to display statistics such as bar charts, floating text, and data transmission paths, as shown in Figure 8. In this figure, the smaller radius rings indicate a communicating node's transmission range, while the larger radius ring indicates which nodes could potentially cause a collision of wireless signals. Small rings above a node indicate that it is currently transmitting, and the cones projecting outward from transmitting nodes indicate which nodes are receiving that transmission. In addition to other displayed data, the visualization also provides a Heads-Up Display (HUD) which provides the user with other scene controls, such as the ability to mark a particular node with a colored flag. The user can see connectivity between nodes, data packet flow, and other network characteristics typically experienced in 2D visualizations, but the addition of a third dimension allows for more information to be displayed at one time.

Huginn supports user control of both the overall view of the network as well as control over simulation time. Huginn allows the user to pan, rotate and zoom throughout the network to gain a better understanding of the interaction between communicating nodes. For example, the user can examine a small subset of nodes, if those nodes are of particular interest. This allows the user a greater understanding of the ongoing data communication and can be useful for evaluating performance of various routing protocols. Additionally, the user can move the simulation forward or backward in time to examine interesting data traffic at a given time point. Giving the user full control of all four dimensions of the visualization provides optimum immersion, and allows for a greater understanding of the network.

2.2.2.3 OPNET. The OPNET Modeler has an optional add-on which enables 3D animation of network events [24]. The OPNET 3D Network Visualization

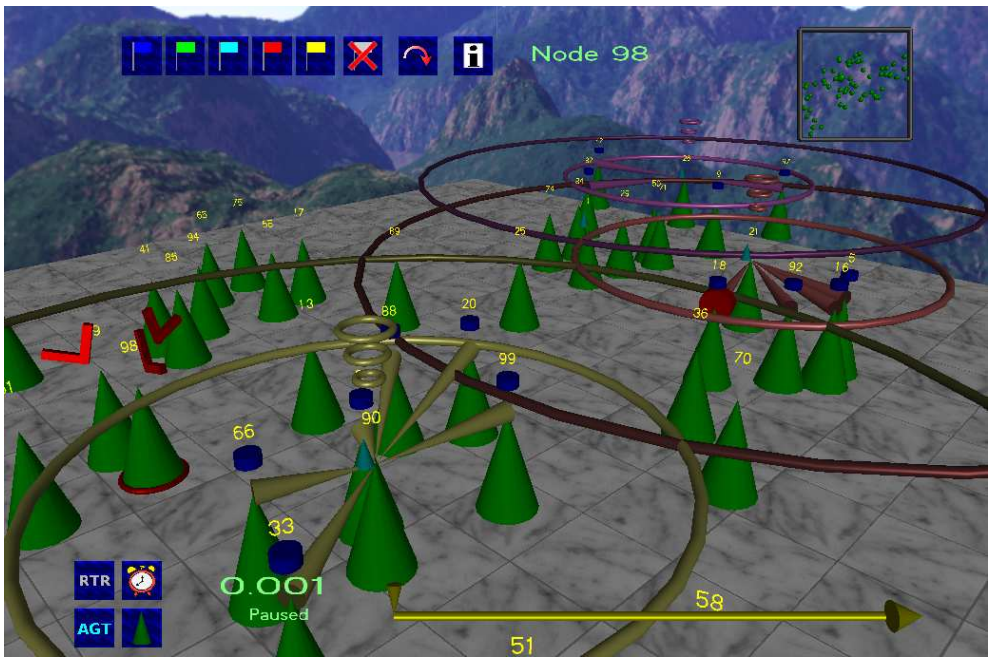


Figure 8: Huginn uses the 3^{rd} dimension to display additional information while visualization a 2D simulation [30]

(3DNV) tool adds the capability to create 3D scene graphs which can be used to depict communication between network assets. The suite provides the capability to use realistic terrain to gain an understanding of terrain masking, propagation loss, and feature interference. The suite provides significant capability for visualizing military networks, and provides a library of 3D models and realistic terrains, which can be further expanded by the user. The visualization allows the user full control of the camera and provides the ability to record animation sequences for future playback. Figure 9 shows a mapping from a 2D OPNET scenario to an equivalent 3D scenario. In each of the two composite images, yellow lines are drawn from the 2D scene to the 3D scene, indicating the mapping of 2D nodes to 3D nodes. Display of network node data such as latitude, longitude and compass bearing in the corners of the screen help the user better comprehend the 3D scene.

Like the rest of the OPNET suite, the 3DNV tool is a proprietary solution which is both expensive to license and difficult to modify. The 3D visualization is even further locked down compared to the core OPNET package, preventing the user from

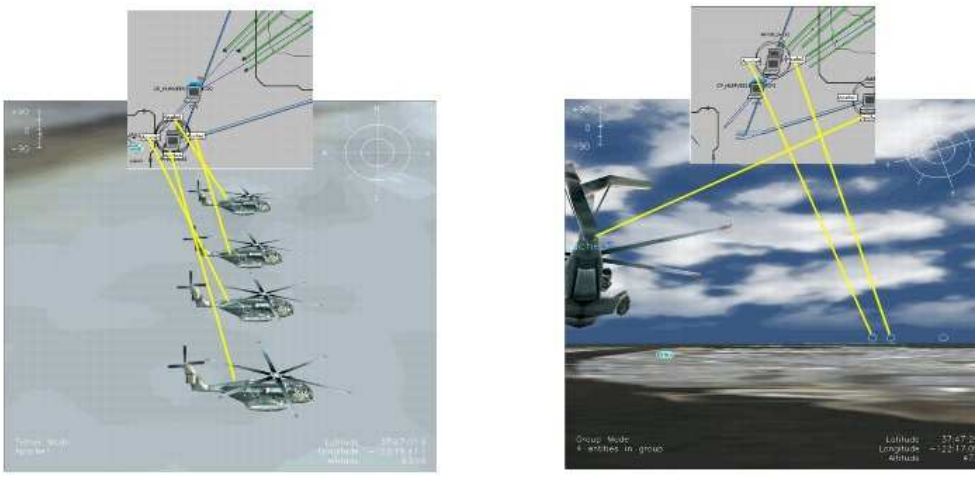


Figure 9: Mapping from a 2D scenario to a 3D scenario in OPNET–the yellow lines indicate which nodes correspond in the two scenarios [26]

modifying the visualization to explore new areas. For example, exploring alternate packet animation schemes is extraordinarily difficult without access to the program's source code. The OPNET 3DNV tool uses colored links to represent current signal strength, but real-time packet animation is not shown in the 3D scene graph [25]. For this reason, the OPNET 3DNV capabilities are not sufficient to examine strengths and weaknesses of various packet animation schemes. An open source solution with equivalent capability would alleviate this problem, allowing the research community to examine alternate network traffic animation algorithms.

2.2.3 Discussion. Based on existing 2D and 3D network visualization, it can be seen that 3D brings new capabilities to the table, though it also brings its share of difficulties. One major strong point of 3D visualization is the ability to use realistic terrain data to explore terrain blockage of wireless communication. Simulations can be executed in 2D using topological data, but to truly grasp how terrain affects signal propagation, a 3D scene is ideal. Another strong suit of 3D compared with 2D is the ability to maintain the interspatial relationships between network nodes. This allows the user to recreate battlefield scenarios in a very realistic manner, accurately reflecting how units will be deployed. This also brings about a downside compared

with 2D, as nodes can no longer be rearranged to increase clarity and address the node occlusion problem. This problem is greatly reduced by allowing the user to reposition the scene camera, though this can require constant interaction from the user in the case of a densely populated scene.

The OPNET 3DNV tool provides the ability to view network simulations in 3D, though its lack of extensibility unfortunately makes it a poor choice for investigating new visualization methods. In addition, OPNET 3DNV requires an expensive license to use, which can be prohibitive to further development. A freely available, open source solution would be ideal for investigating the benefits and drawbacks of different animation schemes, user interaction, and other visualization techniques.

2.3 Evaluating Visualization

As information visualization is subjective in nature, it is difficult to establish a de facto standard for evaluating visualizations. However, there are numerous goals established by prior research that can be used to provide some level of evaluation for the desired network simulation visualization. These evaluation criteria primarily fall into two categories: information visualization and communication network visualization. A third method of evaluating visualization is through human testing, though this method introduces new complications.

2.3.1 Evaluating Information Visualization. There are many suggested methods for evaluating whether a given visualization is effective at conveying information while minimizing user frustration. The following list enumerates the evaluation concepts, and each will be discussed in further detail. The created visualization should:

1. Allow users to have control of time and space [19]
2. Facilitate narration of events [10]
3. Stand on its own [15]

4. Be based on non-visual data [15]
5. Create a result that is readable and recognizable [15]
6. Adapt to serve multiple needs [34]
7. Make effective use of spatial layout and grouping [6]

2.3.1.1 User Control of Time and Space. By allowing the user full control of time and space, many problems are averted when transitioning from a 2D visualization to a 3D one. Viewpoint control helps the user overcome problems of node occlusion and edge intersections, while providing a feeling of immersion in the 3D scene. Control over the time domain allows the user to move forward and backward in time as desired and examine points of interest. By giving the user control over time and space, the user feels less like a spectator and more like an active participant [19].

2.3.1.2 Narration of Events. It has been noted that something people need from information visualization are "tools that make it easy to tell stories" [10]. As a people, telling stories is a fundamental aspect of life. From casual conversations to keynote speeches, effectively telling stories is common in everyday life. A mark of a successful visualization is its ability to facilitate the telling of a story by shifting the narration of events from the presenter to the visualization itself.

2.3.1.3 Independence. An effective visualization must be able to stand on its own, without the need to reference the original data [15]. Visualizations largely exist to reduce the need for manual data processing, allowing trends and patterns to emerge visually without extensive examination of the source data. It is critical that a visualization create an independent picture that needs no cross-referencing to comprehend.

2.3.1.4 Based on Non-visual Data. One of the key points made regarding visualization is that visualization must be based on non-visual data [15]. If the input data is an image and the output includes the same image, no visualization

is actually being performed [15]. Turning raw text or numerical data into something visual is a good step towards creating a useful visualization.

2.3.1.5 Readable and Recognizable. Data is able to be transformed into an image in numerous ways, though not all of these produce an image that allows the user to understand the data. A visualization must produce an image that is understandable to the user, even if it requires training for the user [15]. Images created by the visualization should be immediately recognizable for what they are and not resemble other objects.

2.3.1.6 Adaptability. A visualization should be adaptable to serve multiple needs [34]. Creating a visualization for a sole data sample may be useful for that one sample, but a flexible visualization that can interpret other similar data is ideal. The visualization should be created with a broad function in mind, not only one specific data set.

2.3.1.7 Spatial Layout and Grouping. Spatial layout in a graph influences how a user interprets a given visualization, including determining the relative importance of objects as well as identifying groups [6]. Users often interpret object proximity to indicate relationships, so related objects should be closely grouped while unrelated objects should be placed further apart [4]. Objects should be placed in a meaningful manner in the visualization and not placed at random.

2.3.2 Effective Communication Network Visualization. Since network visualization is a subset of information visualization, there are fewer evaluation methods for determining its effectiveness. The most straightforward method is to compare a newly created visualization to existing visualizations, though there are other factors to consider. A visualization should:

1. Accurately display network simulation events [34]
2. Be efficient with regard to data display [34]

3. Provide equivalent capability as existing visualizations

2.3.2.1 Accurate Network Visualization. A visualization should provide reliable results based on the source data, allowing the user to make accurate quantitative evaluation. For network visualization, this means that every network event in the simulation trace data should have a corresponding event in the visualization. Additionally, timing of network events should be preserved in the visualization. For example, propagation delays should remain consistent between simulation and visualization.

2.3.2.2 Efficient Data Display. A visualization should make effective use of the display space. Ideally, visualizations should be free of needless clutter and maximize the "data-ink ratio" when possible [34]. This means maximizing data while minimizing on-screen clutter and removing unnecessary or redundant images from the visualization. For network visualizations, this means displaying network activity without overwhelming the user with too much data. It can be difficult to find a good balance of data and ink, so it is often necessary to examine multiple visualization methods to find the most appropriate mix.

2.3.2.3 Comparison to Existing Visualizations. A good metric for quantifying the performance of a visualization is to compare it to existing visualizations. If the created visualization can provide equivalent capabilities as current visualizations, while also providing new or improved capabilities, it supports the claim that the new visualization is a step forward. A visualization that provides a feature previously missing from current similar visualizations provides a valid contribution to the community.

2.3.3 Human Testing. Since evaluating a given information visualization is a largely subjective exercise, human testing is often used to quantify results. Test subjects are presented with multiple visualizations based on identical data sets and

each is ranked according to factors such as readability, aesthetics, intuitiveness, and gained insight. This allows researchers to get an understanding of which visualization methods are effective and which are not. However, this requires very careful planning of the tests, and it is often difficult to validate the results due to the underlying nature of human testing. For example, a user who primarily uses a particular visualization is often inherently biased against alternative visualizations because the other visualizations are unfamiliar and therefore more difficult to use. Additionally, it is difficult to quantify the amount of insight gained from one visualization over another [21]. In many cases, the visualization featuring more "eye candy" is seen as superior, despite technical shortcomings [10]. Sometimes there are many solutions that are equally good, but vary in execution [12]. These difficulties with human testing can be alleviated through fine-tuning of the user tests, though this can be a lengthy process.

2.4 3D Graphics Toolkits

Successfully creating a 3D visualization tool first requires choosing a 3D development environment. There are numerous toolkits available for creating 3D visualizations, three of which are investigated in the sections that follow. OpenSceneGraph (OSG) and the Visualization Toolkit (VTK) are both open-source solutions, while Autodesk 3ds Max is a proprietary solution. Distinguishing characteristics are discussed for each.

2.4.1 OpenSceneGraph. OSG provides robust scene management capabilities and optimization of rendered graphics by using a collection of open source libraries. Scenes are built using standard C++, which in turn uses OpenGL low-level graphics APIs to actually draw the scene. Begun as a flight simulator, OSG is currently used in numerous fields, such as virtual reality and scientific visualization. The source code for OSG is freely available and the result is a design environment that is portable to nearly all common operating systems, such as Windows, Mac OS X, and most UNIX and Linux distributions [18].

OSG was designed to create a development environment that is strong in the areas of performance, scalability, flexibility, and portability. OSG supports customization of the scene drawing process, allowing for extensive fine-tuning of system performance. OSG's use of underlying OpenGL libraries means that the programmer can focus on creating content without the need for doing low level programming. OSG comes with its own libraries to facilitate the easy creation of visually attractive scene graphs. These libraries enable easy creation of things such as particle effects, shadows, terrain rendering, animation, and 3D interactive controls. OSG was designed with the idea of minimal dependency of operating platform, and only requires standard C++ and OpenGL to function properly. The result is a scene graph environment that is highly portable and can run on Linux, Windows, Mac OSX, and even the Playstation 2 [23]. OSG was also designed for scalability and can run on anything from the most basic computer systems to multi-core, multi-GPU systems. The result of these design decisions is a highly flexible program that can be used to address a very wide variety of needs.

Like most open source software, OSG's open source nature results in improved quality and stability, as it is reviewed and testing by a large community. By allowing free access to the source code, developers are free to review and debug the code, implement system optimizations, and overall produce a more robust software programming environment. Furthermore, the open source nature of OSG means that the code is available without cost and does not contain any proprietary or patented software. A strong user community has developed around OSG, resulting in many custom-built library solutions for a host of user-specific needs. These libraries extend the core capabilities of OSG in many ways and are freely available for use by anyone.

2.4.2 The Visualization Toolkit. VTK is an open source graphics toolkit used in many universities throughout the world. Originally created as a software companion for a 3D graphics textbook, it has since developed a significant community around it. VTK consists of a custom C++ class library that has been used for

applications such as 3D computer graphics, image processing, and visualization. VTK scenes are typically constructed using interpreted programming languages Python, Java, or Tcl, which is then converted to C++ code before compilation. For advanced users, this C++ code can be tweaked before compilation, or the entire program can be written in C++ from the start if desired. VTK was designed in this way to provide the performance of a compiled language while also providing the flexibility of an interpreted language [31].

The VTK source code is available for anyone to compile and use, which allows users and developers to improve the code as well as tailor it to meet their real-world needs. VTK is a highly flexible toolkit that is appropriate for use in scientific and information visualization applications due to its strong image manipulation capabilities. VTK can be run on any computer independent of windowing system, supporting Windows, Linux, and Macintosh operating systems. Though the source code is provided at no charge, Many questions can be answered by the community built around the toolkit, though the company that created it, Kitware, does offer paid professional support to those that need it [14].

2.4.3 Autodesk 3ds Max. Autodesk 3ds Max, previously known as 3D Studio MAX, is a proprietary toolkit used for modeling, animation, and rendering of 3D graphics. It is used by numerous video game developers, television and movie studios for its flexible plugin architecture and robust 3D modeling capabilities. The feature list for 3ds Max is very lengthy, including multiple shading and texturing tools, polygon modeling tools, and motion capture capability. This extensive functionality comes at a price, however; a single license for 3ds Max retails at $3,495, though student and educational institution licenses are available at a discounted rate [1]. Additionally, 3ds Max only runs on Microsoft Windows, which precludes portability to UNIX-based systems such as Linux or Apple OSX.

3ds Max allows for very extensive scene tuning by giving the user control over individual polygons. This allows the user to fine-tune the 3D object or scene to

achieve optimization. 3ds Max uses its own scripting language, MAXscript, which allows the user to automate tasks, create workflows, define custom user interfaces, and much more. It features a useful scene explorer tool that presents the scene in a hierarchical view, which allows the user to sort, search, or filter the scene by object type. 3ds Max has animation features which allow an object to be animated along a defined curve, with options for alignment, velocity, and looping, among others. This allows the developer to quickly define an animation path for an object. The extensive 3D capabilities of 3ds Max make it a viable candidate for creating a 3D battlefield scene.

2.5 Summary and Way Forward

This chapter has shown that there is a need for an open source 3D network visualization solution. The created 3D visualization should be intuitive for ease of use, animated to capitalize on the visual system, and realistic for quick interpretation. Additionally the visualization must be able to be fully controlled by the user to provide an immersive experience as well as minimize the node occlusion dilemma. Guidelines are established in Chapter III based on the research performed, which should result in a visualization that is both effective at conveying information as well as intuitive to the user.

III. Creating a Realistic 3D Communication Network Visualization Tool

One of the major things that people need most from information visualization today is custom solutions built to specifically address real-world needs [10]. In this case, the real-world need is the ability to visualize complex, military communication systems as deployed in the battlefield, in a realistic three-dimensional (3D) manner. This research aims to directly address that need by creating a flexible, 3D scene graph that is able to accurately depict battlefield communications as well as preserve the interspatial relationships between battlefield assets as deployed in the field. This is a capability that is absent from network visualization today and would be greatly beneficial to the military communication community.

Additionally, this research aims to explore the effectiveness of different data packet animation schemes. The OPNET 3DNV tool is useful for modeling computer networks in 3D, but is limited in its ability to show network traffic, only displaying node connectivity, as seen in Figure 10. In this figure, the aircraft is communicating with nearby nodes, but data packet transmission is not visualized. Due to this limitation, it is desirable to investigate potential ways to animate network traffic in a way that is both accurate according to the simulation and efficient regarding data display.

This chapter discusses a variety of issues used to create a 3D network visualization based on an actual network trace. The first section outlines the development goals used to construct the 3D scene based on concepts discussed in Chapter II. The second section discusses the development environment used to render the scene, including why it was chosen and its benefits. The third section discusses user input, which will allow the user to populate the 3D scene as desired. The fourth section discusses interaction with trace files generated in the NS-2 network simulator and synchronization between NS-2 and the created visualization. Finally, the last section discusses proposed methods of evaluation, to determine the effectiveness of the created visualization.

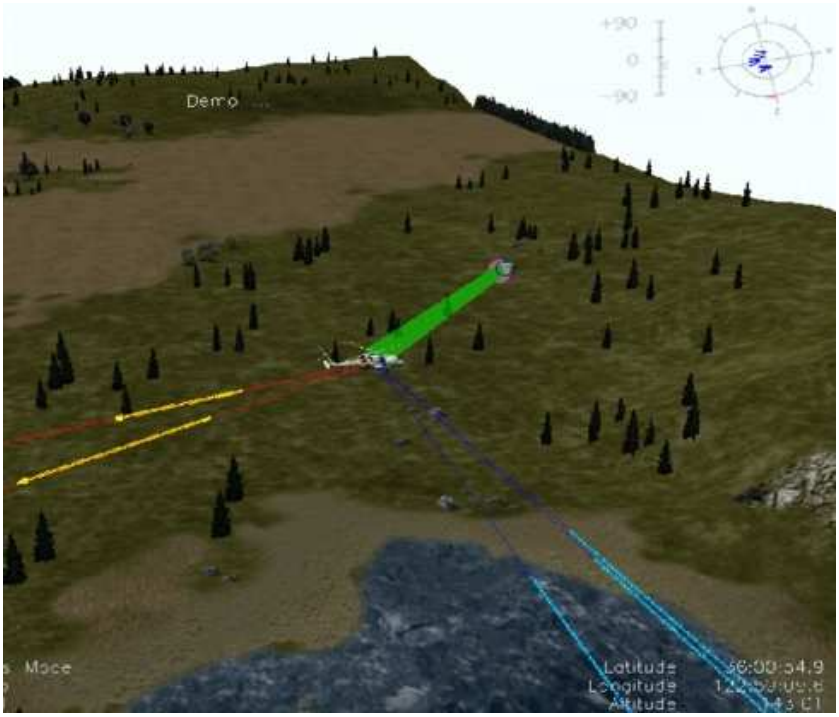


Figure 10: OPNET 3DNV displays asset connectivity, but network traffic is not visualized [26]

3.1 Development Goals

The focus of this research is to model a computer communication network in a realistic 3D environment, with an emphasis on scenes typically encountered in the battlefield. This is done by building a 3D scene graph modeled after a typical battlefield scenario, populated with realistic terrain and communication assets. Figure 11 shows the complexity of 21^{st} century battlefield communications. This image portrays how air assets must communicate with other air assets as well as land, sea, and space assets, in a continuously changing environment. Networks such as the one depicted can be modeled in 2D with limited success, but to effectively convey the geographic location and interspatial relationships between communication nodes, a 3D scene graph is more effective.

The aim of this research is to build a 3D scene representing a theoretic battlefield scenario, guided by the list of goals below. These goals provide a framework for effective 3D network visualization, based on the concepts outlined previously in

Figure 11: Battlefield networks are increasingly connected, requiring communication across a wide variety of asset types [5]

Chapter II. These goals were chosen to enhance the user experience when viewing a network in 3D, while still maintaining equivalent capability as existing 2D visualizations. The goals are described in brief below and each will be discussed in further detail in the sections that follow. The created scene should:

1. Use realistic assets, rather than generic boxes or spheres
2. Use realistic terrain and position the assets at terrain based elevations
3. Allow assets to be mobile according to user input parameters, i.e. flight patterns
4. Show network connectivity as assets travel throughout the scene
5. Show network traffic via animated sequences
6. Allow the user to pan, rotate, and zoom around the scene

3.1.1 Realistic Assets. One major failing of existing visualizations is the lack of realistic assets when portraying networks. Currently used visualizations, such as Nam or OPNET use generic boxes to represent network components. Though this is adequate for representing computer networks featuring stationary assets, such as corporate networks or wireless sensor networks, these types of visualizations fall short

when portraying battlefield components. When battlefield communication assets, such as tanks and planes, are represented by a non-descript box, user immersion is greatly reduced. Either the user has to reference an external network listing that describes which box represents which asset, or the boxes are labeled with functional, yet not ideal, labels beneath the objects. NetViz explored this problem to some extent, by replacing the communicating nodes with images of their real-world counterparts (see Figure 6.

Ideally, the user should be able to tell assets apart immediately without having to cross-reference other documents or text labels. In short, a tank should look like a tank, and a plane should look like a plane. This affords the user greater immersion in the network scene, and allows the user to spend a greater deal of time analyzing the scene, rather than consulting a network key to distinguish network components. In this vein, the 3D network visualization created herein shall use realistic assets as encountered in an actual battlespace environment. By using real-world assets to visualize the network simulation, the foundation is laid for visual credibility by the resulting scene. Visual credibility is vital to user comprehension, since it means that the user's experiences coincide with his/her expectations.

3.1.2 Realistic Terrain and Positioning. The idea of using realistic terrain when visualizing the network simulation again falls under the umbrella of visual credibility. When a user sees a scene that appears to be natural, the user is then able to spend more time analyzing the network, rather than trying to comprehend the scene. Realistic terrain establishes visual credibility by displaying terrain typically encountered on the battlefield. This technique is not currently utilized by existing network visualizations, even those that dabble in the third dimension. The Huginn visualization discussed in Section 2.2.2.2 represents the network in 3D, but positions the nodes along a flat surface. This lack of terrain is not indicative of the real world and therefore visual credibility is weakened. Additionally, the research using realistic terrains, discussed in Chapter II, Section 2.2.2.1, is lacking in its implementation.

The authors use realistic terrains when executing their simulation, but the resulting visualization is the standard Nam output. A user viewing the output in Nam would see the packets being dropped due to terrain interference, but would have no way of knowing precisely which communication links are blocked.

Relatedly, realistic positioning allows for greater visual credibility as well. Existing network simulators position nodes along a 2D plane, or in the case of the research explored in Section 2.2.2.1, along realistic terrain, though always at the surface. However, in any battlefield scenario, there are assets positioned above the ground surface. Most battle operations employ a variety of assets not only on the ground, but also above or below the surface. For example, many ground operations have air or sea support at some level, and no operation is executed without the use of satellite communication. In a 2D network visualization, these air and space assets can be positioned in such a manner as to separate them from the ground assets, but this requires repositioning assets in an unrealistic fashion that degrades the user experience. By representing the network in 3D, however, assets can be correctly positioned in real space, creating an image that is indicative of the real world and adding further to the idea of visual credibility.

3.1.3 Mobile Assets. The third design goal for the created visualization, again tied to the idea of visual credibility, is to allow nodes to be mobile according to user input parameters. Aside from assets such as satellite ground relays or geostationary satellites, communicating nodes in a battlefield environment are rarely stationary. The visualization should support mobile assets, either defined explicitly by the user or inherited implicitly from the simulation trace file. This includes driving paths for ground-based assets, flight paths for air-based assets and submerge animations for assets such as submarines. For example, ground units should be mobile based on a starting location and subsequent waypoints for the asset to travel in a sequential manner. Similarly, air assets should be allowed to loiter in a particular location, subsequently travel to a new location, and loiter there if necessary. If the assets in

the visualization behave in a predictable and familiar manner, the user experience is enhanced.

Allowing communicating assets to be mobile adds a greater degree of realism to the visualization by more accurately depicting what is encountered on the battlefield. Existing 2D visualizations have reasonable capabilities in this regard, especially with the increased research in mobile ad hoc networks. Both the NS-2 and OPNET network simulators allow mobile assets in their simulations, and 2D visualizations such as Nam, OPNET, and iNSpect support accurate visualization of the created simulation data, including node mobility. It is therefore desirable to extend this simulated behavior to the 3D environment, allowing for a more realistic user experience.

3.1.4 Network Connectivity. Displaying network connectivity is a fundamental aspect of successful network visualization. In order to accurately depict data packets traveling from node to node, it must be represented to the user that those two communicating nodes share a communication link between them. To achieve this, the created network visualization will represent this connectivity by drawing lines between communication nodes, much like the manner in which Nam indicates connectivity. This informs the user which nodes communicate with each other in the network simulation. The visualization will have the option to display all links, indicating all available communication paths, or only display those that will be utilized. Displaying all possible communication paths informs the user what communication routes are available, while displaying only those that are utilized reduces scene clutter.

Showing network connectivity in this fashion is the commonly used method in existing wired network visualizations. Typically wireless communication is represented by rings that propagate outward from the transmitting node, which illustrates that the signal often propagates in all directions. This is not the case for all wireless transmissions, however, since satellites, their associated ground stations, and some other battlefield assets utilize directional antennas that transmit roughly in a straight line. On the other hand, some assets have non-directional antennas that communi-

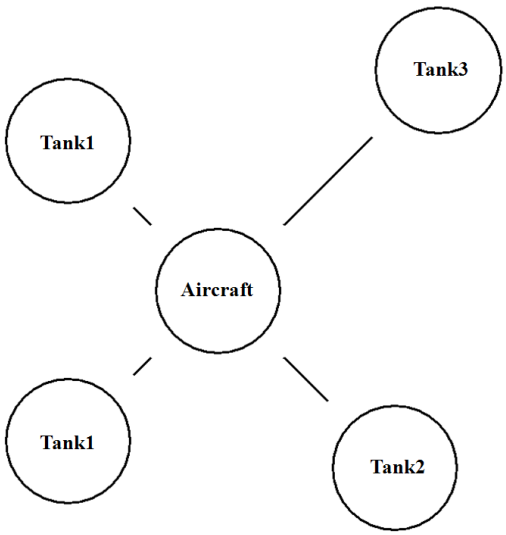


Figure 12: In this sample battlefield scenario, the aircraft at center is providing network connectivity to the tanks around it

cate outward from the transmitting node in all directions. For the sake of consistency, the created visualization will depict all network communication links as lines between two communicating assets. These communication links can either be populated at the start of the scenario and drawn at all times or only drawn when they are being used, based on the desires of the user.

An example scenario is that of a loitering aircraft providing network capability between a group of tanks, as depicted in Figure 12. In this simple example, the aircraft is connected with a line to each of the tanks to illustrate that connectivity is being provided. The tanks, however, are not be directly connected to each other, because communication between them are routed through the loitering aircraft. In the visualization created by this research, the user will be able to choose between displaying all available links or only displaying those links with data being sent across them. Displaying all links allows the user to investigate alternate data paths, while displaying only the communication links that will be utilized decreases the amount of clutter in the visualization and provides more clarity of the network scene.

3.1.5 Network Traffic Animation. Accurately displaying network traffic via packet animation is another fundamental concept of successful network visualization. For the user to be able to determine which nodes are currently communicating there must be a visual indication that this communication is occurring. In the visualization created by this research, network packets will be represented by a small geometric shape that travels along the communication links. This small shape will travel from the transmitting node to the receiving node, much like is seen in a Nam visualization (see Figure 3). In this manner, the user can visually confirm that nodes are successfully communicating and determine which links are the most utilized.

A problem arises when animating packets in this manner, while also allowing assets to be mobile. In a computer communication network, there can be hundreds or even thousands of network events occurring every second. This means that for every foot a tank travels, it is possible that hundreds of packets are transmitted in the network in the same time frame. If the tank is moving in real time, the packets will be moving so quickly that the user will be unable to comprehend what is going on. On the other hand, if time is slowed to allow for closer inspection of network traffic, the tank will appear as if it is not moving at all. Thus the problem is that the quantity of events with regard to packet animation is significantly larger than that of node animation in the same period of time. Four potential solutions are explored for this problem, as described in the following paragraphs.

3.1.5.1 Packet Aggregation. In this solution, network packets are not animated until a transmitting node has sent a threshold number of packets. Different quantities of thresholds will be tested to see if this solution is viable. This will greatly reduce the amount of ongoing packet animation in the scene by only transmitting when the set number of packets is reached. For example, a possibility is to only animated a packet for every 100 simulation packets. This can be easily achieved by implementing a running counter that ignores the first 99 packets, animates the 100^{th} packet, then resets the counter to zero. The packet aggregation scheme can be fine-

tuned according to the scenario to strike a balance between packet animation and node animation.

3.1.5.2 Enlarged Packets. This solution waits a certain duration of time before displaying the packet animation and varies the visualized packet size based on the number of packets it represents. Different amounts of time will be tested to determine optimal time accumulation– one possibility is to only display packet transmission every 2 seconds. Since some nodes will send a greater or fewer number of packets in this timeframe, the size of the visualized packet will increase or decrease accordingly. This will be achieved by varying the packet size based on the number of packets it represents. For example, a visualized packet representing 1000 simulation packets will be twice as large as a visualized packet representing 500 simulation packets. This will be done by implementing counters for each network communication link that increment each time a simulated packet is read from the simulation trace file. Then once every user defined duration, such as every 2 seconds, these counters will used to determine the size of the visualized packets and animate them along each corresponding data path. The counters are then reset and the process repeats. This allows the user to get a quick visual reference of which nodes are sending the most data while helping to find a balance between packet flow and node animation.

3.1.5.3 Triggered Links. This solution eliminates the traveling packet from node to node and instead varies the presence, size, and opacity of the transmission link to indicate packet flow. When two nodes are not communicating at all, no link will be shown between them. When transmission begins, the cylindrical link will be drawn between the communicating nodes and will initially be translucent with a small inside diameter. As network traffic increases, the inside diameter of the cylinder will increase and the link will be made increasingly opaque as data flow increases. The user will be able to define the threshold at which the link becomes completely solid, and this value will be used to determine intermediate transparency values. For example, the user may set a threshold value of 100 packets per second, which means

that a link experiencing 25 packets per second will be 25% opaque and links with a flow of 100 packets per second or greater will be completely opaque. This solution allows the user to quickly grasp which nodes are currently communicating and get a rough esimate of the volume of traffic that is flowing between them.

3.1.5.4 Inspection Button. The final solution implements an inspection button that greatly reduces the time step increment, to allow for further network traffic analysis. In this scenario, packet animation is not shown at all during normal scenario execution, but allows the user to inspect the network traffic at any time by pushing an inspection button. After the button is pressed, time is greatly slowed so that packet animation can be seen, and mobile assets such as tanks and planes appear stationary, though they are actually moving at a very slow rate. The user can then resume the scenario once the visual analysis is complete. A small banner will be overlaid at the top of the screen indicating the scene is currently in the inspection state. This lets the user know that the scene is currently slowed, in the event that the inspection button is pressed during a period of no network traffic. Without the banner, the scene would appear to be frozen, perhaps causing the user to think the application has stopped responding. This solution allows for full analysis of packet animation, while also providing a way to simply view the battlefield scene, all based on the user's desires.

3.1.6 Scene Control. Based on research discussed in Chapter II, the created visualization will allow the user full control of the camera viewing the scene. This means that the user will be able to zoom in and out, pan around the scene, and fully rotate as desired for clarification or inspection of an area. Allowing the user to have full control over the viewing angle of the scene should provide more immersion than fixing the camera in a particular view, and also solves the problem of node occlusion. Transitioning from 2D to 3D means that an asset in the foreground can block other assets communication links located in the background of the scene. To alleviate this problem, the user will be able to rotate around the scene as necessary to remove these

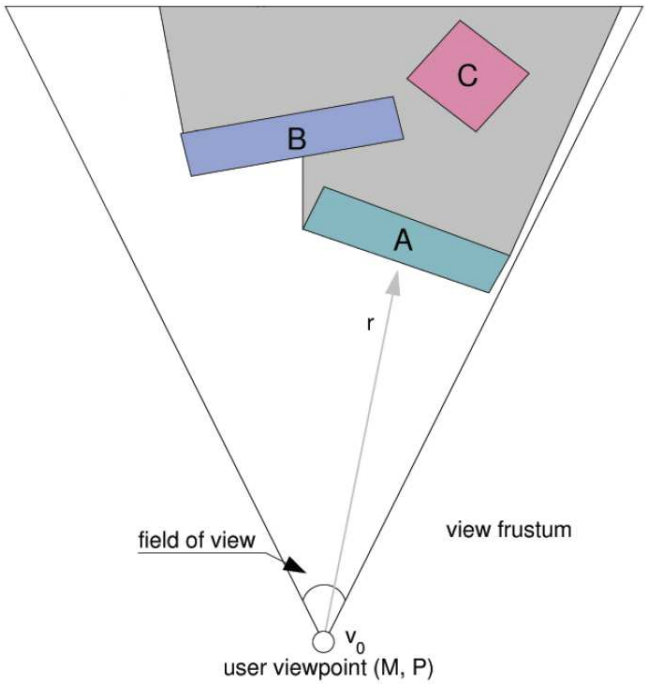


Figure 13: The node occlusion problem—objects A and B block view of object C from the viewpoint of the user [8]

obstructions. Figure 13 demonstrates this problem visually. In this figure the user, located at point v_0, cannot see object C because it is hidden behind objects A and B.

Most importantly, allowing the user full control of the camera creates a feeling that the user is actually "in" the battlefield scene, moving among the various assets. Rather than only viewing the battlefield scene from a distant location, the user is free to travel in and around the scene, move between assets, and truly grasp the interspatial relationships between the communicating nodes.

3.2 Development Environment

Of the considered 3D visualization toolkits discussed in Section 2.4, OpenSceneGraph (OSG) was chosen to be the development environment for the 3D scene. For the sake of flexibility and portability, it was desirable to use a platform independent, open source development environment, which eliminated 3ds Max. In the end, OSG was chosen over the Visualization Toolkit (VTK) due to the availability of extensive

Figure 14: Sample models available for use with OSG [11] [22] [32] [33]

tutorials. Created by a computer science instructor at the Naval Postgraduate School, these tutorials provide a foundation for scene creation, including asset positioning and terrain placement [32]. These tutorials proved to be very useful in designing 3D scenes and helped get creation of the visualization tool off the ground.

OSG supports numerous 3D model file types, of which there are many available freely on the Internet. Figure 14 shows a few of the 3D models that are freely available for use with OSG. Clockwise from the top left, these models are: an A-10 aircraft, a tank, a Harrier jet, and a Humvee. Further realism can be achieved by using the exact models that will be deployed in a given military operation.

3.3 Simulator/Visualization Synchronization

The executed network simulation must be properly synchronized with the desired battlefield scene to produce meaningful results. There must be a one-to-one correspondence of nodes in the network simulator to those in the visualization. Animation paths must be consistent between the simulation and visualization. Care

must be taken to ensure that nodes are mapped accurately from the simulation to the visualization, so that network communication is represented correctly.

To ensure that the visualization and simulation aspects are properly synchronized, the visualization must be created in a three step process. First, the user must design the 3D scene using the created visualization tool, including the placement of network nodes, utilized communication links, and definition of desired animation paths. Creating the scene in 3D first allows the user to determine problem areas before proceeding to the simulation step. Drawing desired communication links ahead of time helps to identify signal blockage due to terrain or other obstacles. Defining animation paths prior to simulation identifies potential object collision due to overlapping animation paths. Once this step is complete, the scene will then need to be recreated in an NS-2 simulation, maintaining node location and animation as accurately as possible. The distance between nodes should remain consistent to preserve signal propagation data and the 2D nodes should have animation paths that match their visualization counterparts. The simulation can then be executed, which will produce a network trace file that will be representative of the 3D network. Finally, the last step is to import the NS-2 trace file into the 3D visualization tool, allowing the user to see the simulation results in a realistic 3D environment.

This three step process ensures that each visualized node has a corresponding simulation node, while also alleviating problems with transitioning from 2D to 3D. The user can first ensure that the 3D scene is acceptable before proceeding to the simulation step. Otherwise, the user may need to change the 2D simulation multiple times due to signal blockage not present in a 2D scene. Each of these steps is explored in further detail in the sections that follow.

3.3.1 Scene Creation and Validation. The rendered battlefield scene must be defined by the user before it can be effectively displayed. The created network visualization will allow the user to define the number, type, and position of assets as well as animation paths for mobile assets. The user can define which nodes are

connected via a communication link to determine if there is any obstacle that would prevent successful communication. This allows the scene to be fully customizable to the user's needs.

3.3.1.1 Asset Population. Many node types will be available for use in the scene, including planes, helicopters, tanks, and submarines, among others. Other node types can be utilized if they are in a format that is recognized by OSG. Each of these assets will require an *X, Y, Z* coordinate for placement in the scene

To populate the scene, the user will be able to specify the number, type, and location of desired assets. Each created asset will have an associated group of Cartesian coordinates (*X, Y, Z*) that specifies where in the scene it will be located when the visualization is started. Ground assets such as tanks or satellite ground stations will be positioned using only the X and Y coordinates and the Z coordinate will be automatically adjusted so the asset is located on the ground. Since the visualization uses realistic terrain, the ground Z component will vary throughout the scene, meaning the Z component of a ground asset must be updated based on the elevation to prevent driving under the terrain or floating above ground. A Z coordinate of zero will be reserved for ground assets, which tells the visualization to keep the asset in contact with the terrain as the scene executes. Air assets such as planes and satellites and sea assets such as aircraft carriers and submarines will be positioned using all three coordinates. Sea assets such as submarines may require a negative Z coordinate, to allow them to be partially submerged under a water terrain feature.

3.3.1.2 Animation Paths. To further add to the realism of the battlefield communication scenario, the user will be able to define paths for mobile nodes such as tanks and planes. For all node types, the user will be able to define waypoints for the asset to travel, using Cartesian coordinates, a time value, and a delay value between waypoints. The asset will then travel to the waypoints in a sequential manner, ultimately waiting at the final waypoint for the remaining duration of the visualization. The time value will specify the time the asset should take to travel

between waypoints, which will effectively define the velocity of the mobile node. The delay value allows the mobile asset to wait at a particular waypoint for a given length of time before proceeding to the next waypoint. For land-based nodes, the Z coordinate value will be automatically updated to correspond to the current terrain. For air-based nodes, the asset will travel in a direct line-of-sight manner to the next waypoint. For air assets only, the user will be able to specify a second type of animation path. The user will be able to specify a loiter animation path, which allows an airplane to fly in a circular manner around a target area. This can be useful if an air asset is going to be used to provide connectivity to other nodes in a given area. Defining and visualizing these animation paths in the first step allows the user to discover and fix problems such as node collisions prior to simulation.

3.3.1.3 Communication Links. It is desirable to investigate communication links prior to network simulation, in the event that communication will be blocked by terrain features or other obstacles. The user will be able to display all links between nodes to determine if signal blockage will occur. Based on this information, the user can then reposition assets if necessary. For example, if a mountain range is blocking communication between a tank and a helicopter, it may be possible to restore communication if the helicopter is positioned at a higher altitude. Displaying communication links prior to simulation improves the accuracy of the simulation results.

3.3.2 Simulation. Once the scene has been defined by the user, the 3D scene must be manually translated into an equivalent 2D version in NS-2. The 2D network nodes should be positioned in a way that reflects the 3D positioning, and animation routines must be recreated in NS-2. The user must then define network flows for each node in the network, including data rates, propagation delays, queueing delays, and so on, as they normally would for NS-2 simulations today. The simulation can then be executed, which will then create a trace file to be used with the visualization.

```
h -t 1.0016 -s 5 -d 1 -p cbr -e 1000 -c 3 -i 7 -a 3 -x {5.0 3.0 -1 ------- null}
+ -t 1.01 -s 0 -d 1 -p cbr -e 1000 -c 0 -i 8 -a 0 -x {0.0 9.0 2 ------- null}
- -t 1.01 -s 0 -d 1 -p cbr -e 1000 -c 0 -i 8 -a 0 -x {0.0 9.0 2 ------- null}
h -t 1.01 -s 0 -d 1 -p cbr -e 1000 -c 0 -i 8 -a 0 -x {0.0 9.0 -1 ------- null}
+ -t 1.01 -s 0 -d 1 -p cbr -e 1000 -c 0 -i 9 -a 0 -x {0.0 9.0 3 ------- null}
+ -t 1.01 -s 8 -d 7 -p cbr -e 1000 -c 1 -i 10 -a 1 -x {8.0 7.0 2 ------- null}
- -t 1.01 -s 8 -d 7 -p cbr -e 1000 -c 1 -i 10 -a 1 -x {8.0 7.0 2 ------- null}
h -t 1.01 -s 8 -d 7 -p cbr -e 1000 -c 1 -i 10 -a 1 -x {8.0 7.0 -1 ------- null}
+ -t 1.01 -s 8 -d 7 -p cbr -e 1000 -c 1 -i 11 -a 1 -x {8.0 7.0 3 ------- null}
+ -t 1.01 -s 2 -d 1 -p cbr -e 1000 -c 2 -i 12 -a 2 -x {2.0 4.0 2 ------- null}
- -t 1.01 -s 2 -d 1 -p cbr -e 1000 -c 2 -i 12 -a 2 -x {2.0 4.0 2 ------- null}
h -t 1.01 -s 2 -d 1 -p cbr -e 1000 -c 2 -i 12 -a 2 -x {2.0 4.0 -1 ------- null}
+ -t 1.01 -s 2 -d 1 -p cbr -e 1000 -c 2 -i 13 -a 2 -x {2.0 4.0 3 ------- null}
+ -t 1.01 -s 5 -d 1 -p cbr -e 1000 -c 3 -i 14 -a 3 -x {5.0 3.0 2 ------- null}
- -t 1.01 -s 5 -d 1 -p cbr -e 1000 -c 3 -i 14 -a 3 -x {5.0 3.0 2 ------- null}
h -t 1.01 -s 5 -d 1 -p cbr -e 1000 -c 3 -i 14 -a 3 -x {5.0 3.0 -1 ------- null}
```

Figure 15: Sample NS-2 trace file

The 3D network visualization will require an output file from a network simulator to provide realistic results. The file format for this research is an NS-2 trace file, typically used by the Nam visualization to display simulation results. The structure of this file format makes visualization of the simulation results fairly straightforward.

Figure 15 shows a sample NS-2 output file in a format typically used with the Nam visualization. It is important to understand what each symbol means in the trace file to correctly implement the visualization. Lines starting with an *h* indicate that a packet is transmitted to the next hop. Lines starting with an *r* indicate that the packet has been received by the destination node. Lines that begin with + and - indicate enqueue and dequeue, respectively. The next parameter, -*t*, on each line is the corresponding time stamp, which indicates at what point in the simulation an event occurs. The next two parameters, -*s* and -*d*, indicate the source and destination nodes, respectively. The parameter -*i* establishes a packet identification (ID) field that can be used to track packets as they flow through the network. The remaining parameters will be unused.

3.3.3 Network Visualization. Once the trace file is generated by the simulation, it can then be imported into the 3D scene to display simulation results. The created visualization will utilize the NS-2 simulation file output to display the network activity in the simulation, much as Nam is used today. The packet animation subroutine will be triggered by each *h* line to indicate that a packet is being trans-

mitted from the source node to the destination node. By using the ID field and the time stamps from the h line to the corresponding r line, the visualization can determine how quickly each packet will travel from source to destination. This allows the visualization to accurately portray network events according to the simulation.

3.4 Measures of Success

Due to the subjective nature of information visualization, it can be difficult to establish effective measures of success for a visualization. However, as discussed in Chapter II, Section 2.3, prior research outlines a number of goals which can be used to provide some level of evaluation for the created visualization. These evaluation methods primarily fall into two categories: effective information visualization and effective communication network visualization. A third method of evaluating visualization is through human testing, though this method is not without its share of difficulties.

3.4.1 Effective Information Visualization. There are many suggested methods for evaluating whether a given visualization is effective at conveying information while minimizing user frustration. The following list enumerates the goals to be evaluated against; further detail follows in subsequent sections. The created visualization should:

1. Allow users to have control of time and space [19]
2. Facilitate narration of events [10]
3. Stand on its own [15]
4. Be based on non-visual data [15]
5. Create a result that is readable and recognizable [15]
6. Adapt to serve multiple needs [34]
7. Make effective use of spatial layout and grouping [6]

3.4.1.1 User Control of Time and Space. Allowing the user full control of time and space averts many problems when transitioning from a 2D communication network to a 3D one. This goal will be achieved in the created visualization by allowing the user to have complete control over the scene camera, including panning, zooming, and rotating the scene as desired. The user will also be able to move forward or backward in time to examine events of interest as the scenario executes.

3.4.1.2 Narration of Events. The created visualization will support the narration of events by shifting the narration of events from the presenter to the visualization itself. Accurately depicting battlefield events, including network traffic, directly aids the presenter by painting a clear picture of the operation environment. This will create an intuitive representation that allows the presenter to focus more on network event details and spend less time explaining the scene itself.

3.4.1.3 Independence. The created visualization will portray the battlefield scene and ongoing communication in such a way that it will not be necessary for the user to reference the original trace data to understand what is happening. By mimicking reality as must as possible, the created visualization will provide a representation of a battlefield scenario that requires minimal explanation. The user will be able to comprehend the scenario without needing to examine the NS-2 trace data to determine context.

3.4.1.4 Based on Non-visual Data. The visualization will be executed based on an NS-2 trace file, which is purely textual in nature. To the layperson, this file appears as a string of random numbers and letters. For someone that understands the NS-2 file convention, network events can be read manually from the file, though this is incredibly tedious and mistakes are easily made. The created visualization will take this data and present it in a more understandable manner.

3.4.1.5 Readable and Recognizable. Creating a 3D scene that accurately depicts a real-world battlefield scenario through the use of realistic asset models and terrain will create a scene that is easily readable and recognizable to the user. For example, a tank will be immediately recognized as a tank, a plane will be immediately recognized as a plane, and so on. Positioning these realistic assets in a realistic terrain environment creates a coherent, meaningful framework that can be rapidly understood without conscious thought [36].

3.4.1.6 Adaptability. The visualization is designed to support any number of battlefield scenarios, including the use of numerous asset types and any number of nodes. The visualization will prove useful for modeling any type of battlefield scenario, from small-scale operations including a handful of assets, to large-scale operations utilizing dozens of assets. The created scene is designed in particular to depict battlefield scenarios, but could easily be extended to other applications by replacing the 3D models with those of other significance to the user.

3.4.1.7 Spatial Layout and Grouping. The spatial layout of scene objects must be carefully considered when creating a visualization. By accurately maintaining the interspatial relationships of battlefield nodes, the user gains a greater understanding of the scenario. Assets will be positioned in exactly the same manner as their real-world counterparts, resulting in a high degree of visual credibility. Users will be able to rely on the location of the visualization nodes and draw appropriate conclusions.

3.4.2 Effective Communication Network Visualization. Since network visualization is a subset of information visualization, there are fewer evaluation methods for determining its effectiveness. The primary method is to compare the newly created visualization to existing visualizations, though there are two other considerations. The created visualization should:

1. Accurately display network simulation events [34]

2. Be efficient with regard to data display [34]
3. Provide equivalent capability as existing visualizations

3.4.2.1 Accurate and Efficient Network Visualization. The visualization as a whole, as well as each individual network traffic animation scheme explored in Section 3.1.5, will be evaluated based on accuracy of network events and efficiency of data display. Accuracy in this case means that every simulation event has a corresponding visualization event and packets are transmitted at the same rate as the simulation. For network visualization, it is primarily packet animation that can quickly fill the screen with an overwhelming amount of data, which creates a need for efficient data display. In some cases, the user cannot effectively process everything that is occurring on the screen, which necessitates an animation scheme that alleviates this problem. Especially when combined with node mobility, some animation schemes may be too cluttered while some may be too sparse in detail. Each packet animation scheme has associated benefits and drawbacks, and will be compared with each other to determine the viability of each. Some potential benefits and drawbacks are discussed in the sections that follow, for each proposed packet animation scheme.

Standard Animation Scheme. This is the simplest of the packet animation schemes, in which every simulated network event has a corresponding event in the resulting visualization. Since there is a one-to-one correspondence from simulation to visualization, this animation scheme is expected to be very accurate. However, when a simulation containing a high volume of traffic is combined with a scene with numerous mobile assets, this animation scheme is likely to overwhelm the user with too much information. Thus the standard packet animation scheme is expected to have a high degree of accuracy, but a low degree of display efficiency.

Packet Aggregation. The first explored packet animation scheme to address the shortfalls of the standard scheme involves reducing the volume of traffic by a set factor. For example, network traffic can be reduced by a factor of 100 by

visualizing every 100^{th} simulation event at each node. In this manner, the display efficiency is increased, though it comes at a cost of accuracy. The user still has a rough idea of the volume of traffic leaving each node, but it is not as precise as the default scheme. Thus this scheme is expected to have a medium degree of accuracy and a medium degree of efficiency. There is an additional shortfall, in that some nodes may appear to be inactive if they rarely transmit. Since each source generates traffic at a different rate, less active nodes rarely send visualized packets, while more active nodes send multiple visualized packets over the same time period. A less active node may appear to be dormant for long periods of time, though this may not be the case. Additionally, in many network simulations, a successfully transmitted message is often followed by an acknowledgment back to the source but in this animation scheme, this exchange cannot be seen.

Enlarged Packets. The second explored packet animation scheme attempts to address one problem of the packet aggregation scheme, in which less active nodes may appear dormant despite infrequent transmissions. This problem is addressed by visualizing packets in a fixed time interval and varying packet size based on accumulated simulation packets. For example, in a one-second time block, a more active node may have generated 1000 packets, whereas a less active node may have only generated 100 packets. In this scenario, the visualized packet sent from the more active node will be 10 times larger than the visualized packet sent from the less active node. However, like the packet aggregation scheme, this animation scheme also is unable to convey the message/acknowledgment handshake between nodes. This scheme also makes the scene appear as though all communication is synchronous, which may be misleading to the user. Like the packet aggregation scheme, this animation scheme is expected to have a medium degree of accuracy and a medium degree of efficiency.

Triggered Links. Unlike the previous animation schemes, this scheme does not display packets, instead focusing on connectivity. The result is

a scene that is highly efficient at displaying data, though performs very poorly at accurately displaying network traffic. The size and transparency of the link drawn between two assets gives the user some indication of the volume of traffic exchange, but distinguishing between the opacity of two links can be difficult. Additionally, determining which endpoint is the source and which is the destination is not possible, so traffic direction is not known. However, if the user is only interested in a high-level overview of the connectivity of the network, this scheme works extremely well. Since links are only drawn if they are currently in use, it becomes easy to see which nodes are currently communicating and which are not. This scheme is expected to have a low degree of accuracy, but a high degree of display efficiency.

Inspection Button. Implementing an inspection button attempts to strike a balance between the time difference of mobile assets and network communication. This packet animation scheme is a hybrid of the standard scheme and the triggered links scheme. When the inspection button is inactive, assets move freely about the scene, but packet animation is not shown; the links are drawn in the manner of the triggered links scheme, but the packets are not visualized. When the inspection button is active, packet animation is shown as in the standard scheme, and mobile assets move extremely slowly, appearing to be nearly stationary. The ability to toggle the current state of the visualization creates two complementary situations. In the state in which the inspection button is inactive, the scene is efficient with regard to data display, though inaccurate with regard to network traffic. In the state in which the inspection button is active, the opposite is true—the display space is cluttered and inefficient, but highly accurate with regard to network communication. The result is a visualization tool that can be altered on the fly to display the current desired amount of detail.

Summary. A summary of the different animation schemes can be seen in Table 1, which shows that there is no universal solution among the animation schemes explored. Accuracy and efficiency are two sides of the same coin, and it is

Table 1: Comparison of proposed packet animation schemes

Animation Scheme	Accuracy	Efficiency
Standard	High	Low
Packet Aggregation	Low	Medium
Enlarged Packets	Low	Medium
Triggered Links	Low	High
Inspection Button–Inactive	Low	High
Inspection Button–Active	High	Low

difficult to increase one without decreasing the other. However, by offering many options, the user is free to explore which scheme is most useful for a given scenario.

3.4.2.2 Comparison to Existing Visualizations. Of the many network visualizations discussed in Chapter II, Section 2.2, the visualization created by this research will be compared with one 2D visualization and one 3D visualization. The created visualization will be compared with the NS-2 visualization, Nam, as well as OPNET 3DNV to determine distinguishing characteristics of each. From a network state perspective, packet animation, queue size, and node movement will be examined for each visualization. There will be a discussion of the features present in the newly created visualization that are not present in either Nam or OPNET 3DNV.

3.4.3 Human Testing. As discussed in Chapter II, human testing can be useful for quantifying visualization, though it brings about numerous difficulties. Successfully designing a test that eliminates user bias is a daunting task that can take a significant amount of time. Users are often won over by the most attractive visualization, regardless of capability. Data collection itself can take a significant amount of time, depending on test subject availability. Finally, once these results are collected, it can be difficult to validate them in a meaningful way due to the lack of standards in the visualization community. Due to these difficulties with human testing and the imposed time constraints, human testing will not be performed for the created visualization, though it will hopefully be explored at a later date.

3.5 Summary and Way Forward

This chapter has outlined the steps necessary to successfully create the 3D network visualization tool. The chosen development environment should prove to be a suitable platform to create such a tool, and utilizing NS-2 trace data avoids the complications often encountered with proprietary software. By aiming to achieve the development goals described and keeping the measures of success in mind, a 3D network visualization can be created that is capable of accurately and efficiently displaying network events and packet animation.

IV. Evaluating the Created 3D Network Visualization Tool

This research produced a 3D communication network visualization tool that is capable of rendering a realistic battlefield scene, along with ongoing communication between network nodes. The new tool allows users to create 3D battlefield scenarios, and parses NS-2 trace files to display network communication. Each topic discussed in Chapter III is revisited and the realized tool is compared with the proposed criteria. Visualization is very subjective in nature, and therefore difficult to quantify. The tool is evaluated through visual inspection, with every attempt made to be as objective as possible. The core functionality of the tool has been realized, though further refinements would be very beneficial.

4.1 Development Goals Revisited

In Section 3.1, six development goals were outlined, serving as a roadmap for creating a 3D network visualization. In this section, each development goal is revisited to assess whether each is satisfied. As with all software development, tradeoffs must be made and some features placed on the back burner, possibly to be implemented in a future version. The six goals are reprinted below for ease of reference.

1. Use realistic assets, rather than generic boxes or spheres
2. Use realistic terrain and position the assets at various elevations
3. Allow assets to be mobile according to user input parameters, i.e. flight patterns
4. Show network connectivity as assets travel throughout the scene
5. Show network traffic via animated sequences
6. Allow the user to pan, rotate, and zoom around the scene

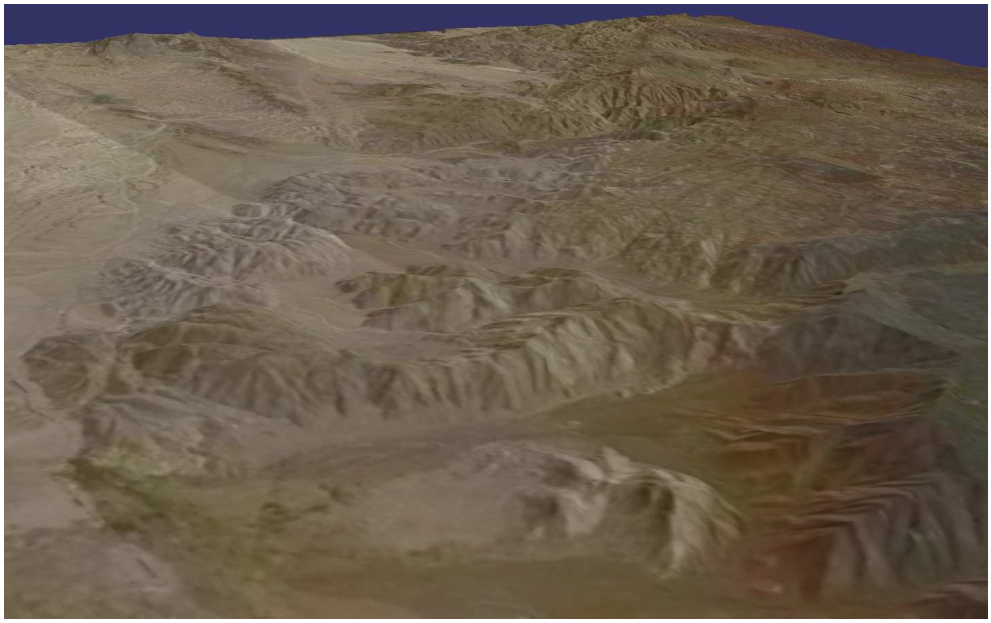
4.1.1 Realistic Assets. The 3D scene is constructed from freely available 3D models of various military assets, including tanks, Humvees, helicopters, and planes. The process of placing models onto the terrain is fairly straight-forward, as assets can be positioned using a simple *X, Y, Z* coordinate system to specify their location. An

Figure 16: Realistic assets positioned on a realistic terrain

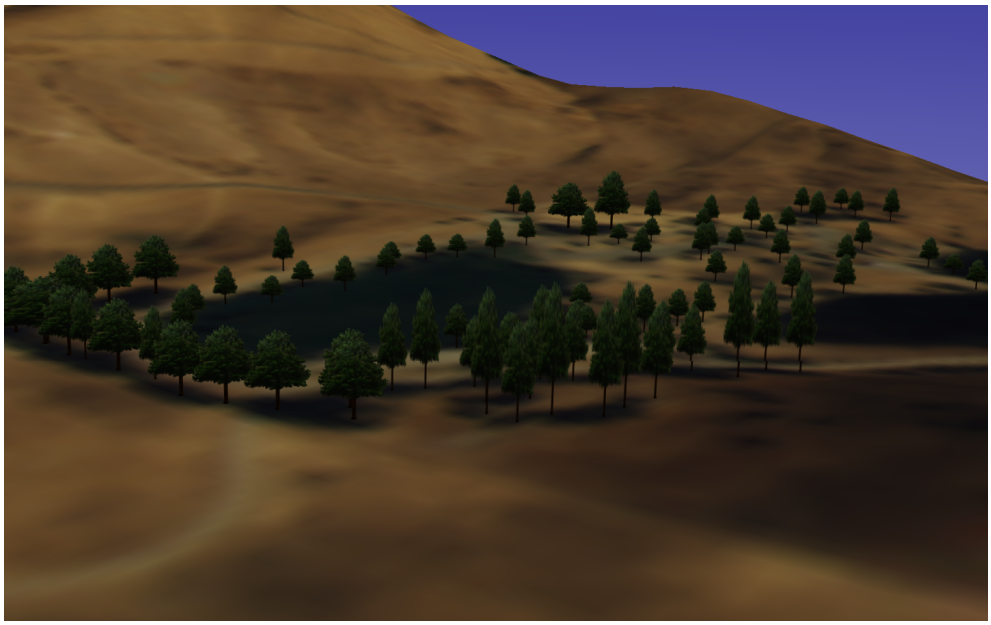
example of the realistic assets can be seen in Figure 16. The assets shown here are just a sample of the 3D models available, which can easily be swapped out to suit the desired operating environment.

4.1.2 Realistic Terrain and Positioning. Similar to the realistic assets, there are also openly available terrain models that mimic real-world environments fairly well. Figure 16 shows a sample terrain with assets positioned throughout the scene. This sample terrain illustrates how network communication could potentially be blocked by terrain features such as hills or mountains. The shown terrain is not based on real-world data, though more realistic terrains could be created from topological data or purchased from a third-party vendor. Terrains based on real-world data would provide increased realism to accurately represent an operational environment, including precise placement of buildings and foliage. Figure 17 depicts two other sample terrains, including mountains, trees, and water.

4.1.3 Mobile Assets. The created visualization supports circular mobility paths for both ground and air nodes. The user must define the circular path by



(a) Terrain featuring a mountain range



(b) Terrain featuring trees and lakes

Figure 17: Two sample terrains that can be used to create 3D scenarios

specifying an X, Y, Z coordinate that indicates the center of the circular path, a desired radius, and the amount of time each revolution should take. This allows assets to travel throughout the scene as defined by the user. The series of images in Figure 18 shows an A-10 aircraft flying in a circular path near other assets. The A-10 maintains connectivity to the connected nodes at all times.

It was originally envisioned that the visualization would also support linear animation paths, but this was not achievable in the given time frame. This would enable mobile assets to be defined by a series of sequential waypoints, allowing a node to travel through the scene in a very specific manner. This addition would greatly increase the mobility capabilities of the created network visualization tool.

4.1.4 Network Connectivity. The created visualization in Figure 19 depicts two communicating assets as connected via a thin green cylinder. The default behavior only connects those nodes that communicate, which allows the user to quickly get an idea of which nodes communicate during the scenario. The visualization can be modified to display all available links or to only draw links during communication, which must be set by the user before visualization runtime. An additional aspect of this development goal is to maintain connectivity in the event that assets are mobile. This has been successfully achieved and can be seen in action in Figure 18. The tanks and Humvee ground assets maintain connectivity to the A-10 as it travels throughout the scene.

4.1.5 Network Traffic Animation. The created visualization shows data packet flow between communicating nodes, implemented as spheres traveling along the communication link cylinders. Each network event in the NS-2 simulation generates a visualized packet in the visualization, which travels from source to destination. A sphere, representing one data packet, is sent from one node to the next, traveling along the cylinder connecting them. This is demonstrated in Figure 20, though the animation aspect is lost in the figure. Instead, the packet has been made increasingly opaque to emulate animation from the tank to the helicopter.

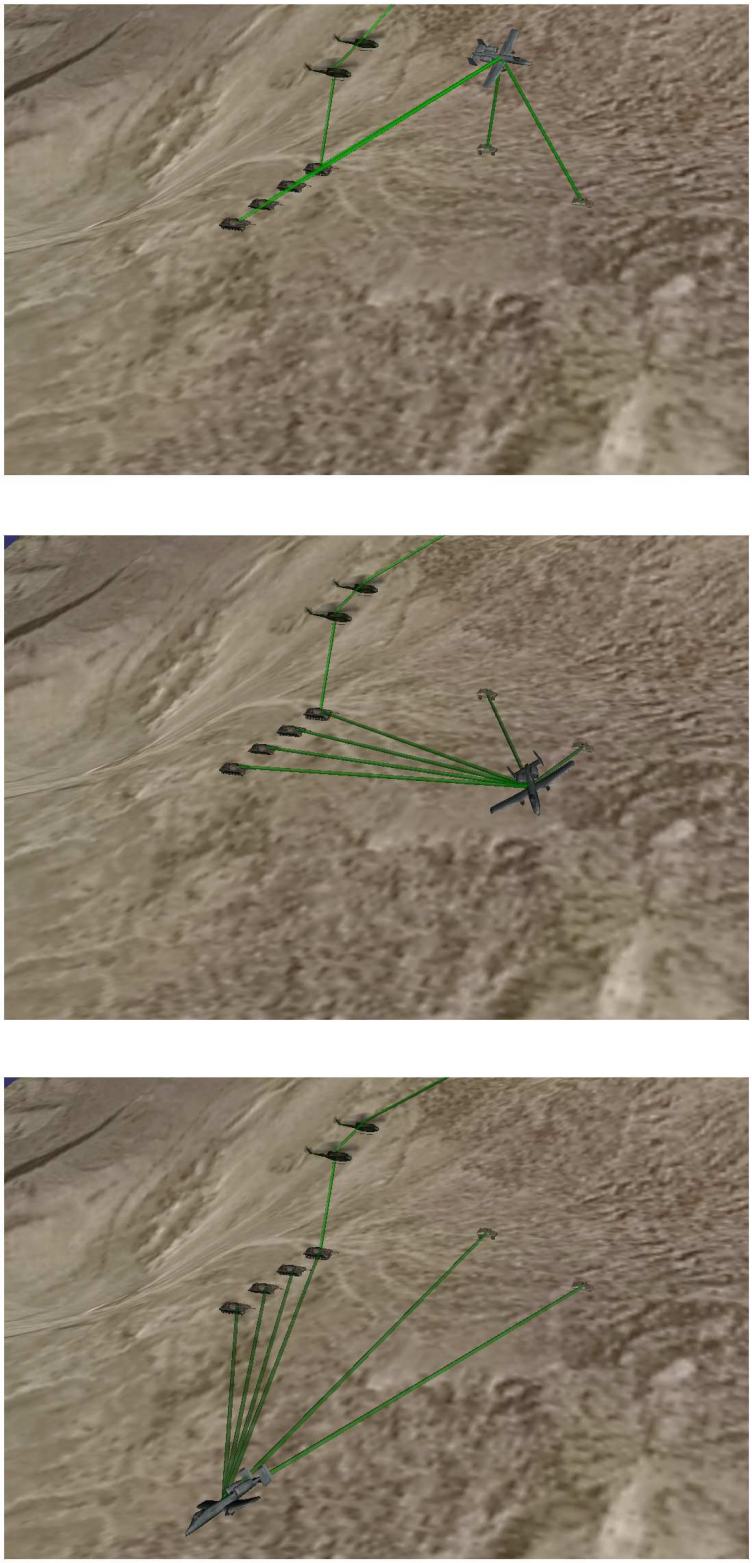


Figure 18: The A-10 maintains connectivity while traveling around the scene

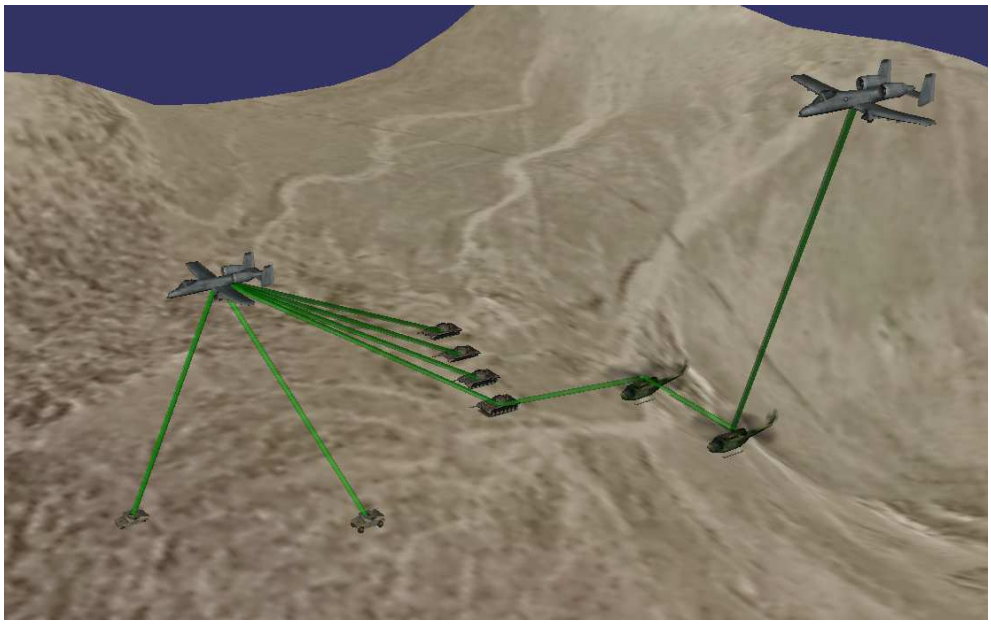


Figure 19: Network assets are connected with green cylinders, which indicate potential communication paths

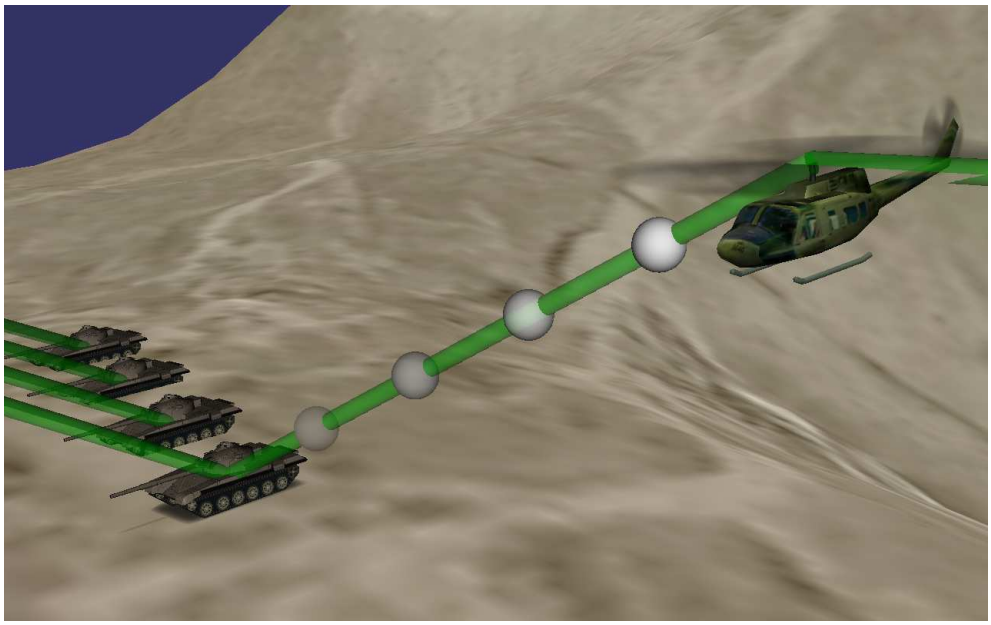


Figure 20: Simulated packet flow between network assets—the packet is flowing from the tank to the helicopter

There were numerous packet animation schemes outlined in Chapter 3, though only the standard animation scheme was realized due to time limitations. In the standard animation scheme, each simulation network event has a corresponding visualization network event. This is the most basic of the proposed animation schemes as well as the most accurate, so priority was given to implementing it correctly. The visualization tool could be further improved by the implementation of the other proposed schemes, but this was not achievable in the available time for this research.

4.1.6 Scene Control. The ability for the user to pan, zoom, and rotate the scene as needed is trivially satisfied by core functionality of OSG itself. OSG provides the ability for the user to freely position the scene camera, allowing the viewer to investigate the scene as desired. This allows the user to zoom in to examine a particular node, as in Figure 21. This figure shows a closeup view of an A-10, which is connected to tank assets on the ground. User control over the scene camera also solves the problem of node occlusion, in which a foreground node blocks a background node. Figure 22 shows how the user can reposition the camera to avoid this problem, uncovering the Humvee from behind the foreground helicopter.

4.1.7 Discussion. All of the six development goals were completely realized, which is a significant achievement. The resulting scene is a very intuitive representation of a realistic battlefield scene, including communication links and packet animation. The implemented asset mobility capabilities allow air, land, and sea assets to travel throughout the scene in a realistic manner. Finally, allowing the user full control of the scene not only provides for a more immersive experience, but also alleviates some of the problems of transitioning from 2D to 3D.

4.2 Development Environment

OSG proved to be a suitable development environment for creating a 3D network visualization, and its core functionality proved quite useful. Its camera controls in particular were extremely useful in eliminating the node occlusion problem without

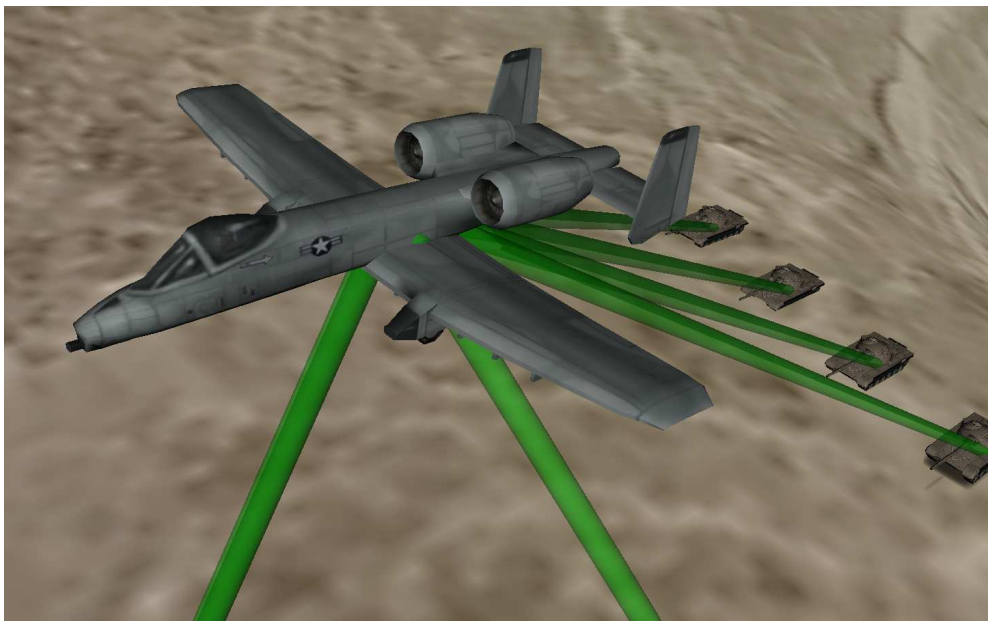
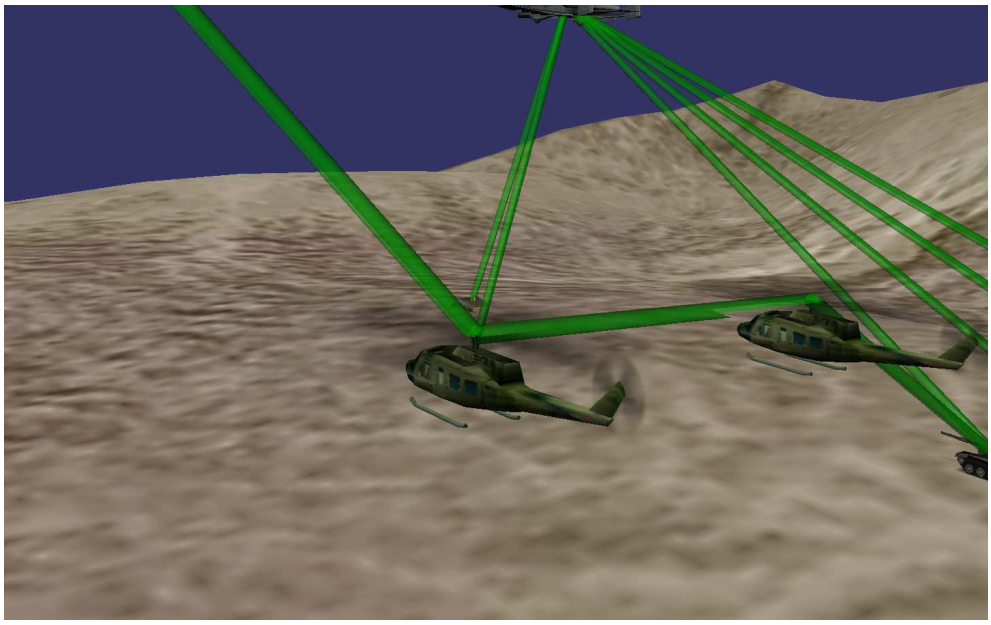


Figure 21: The user is free to zoom in on any asset, in this case an A-10 connected to assets on the ground

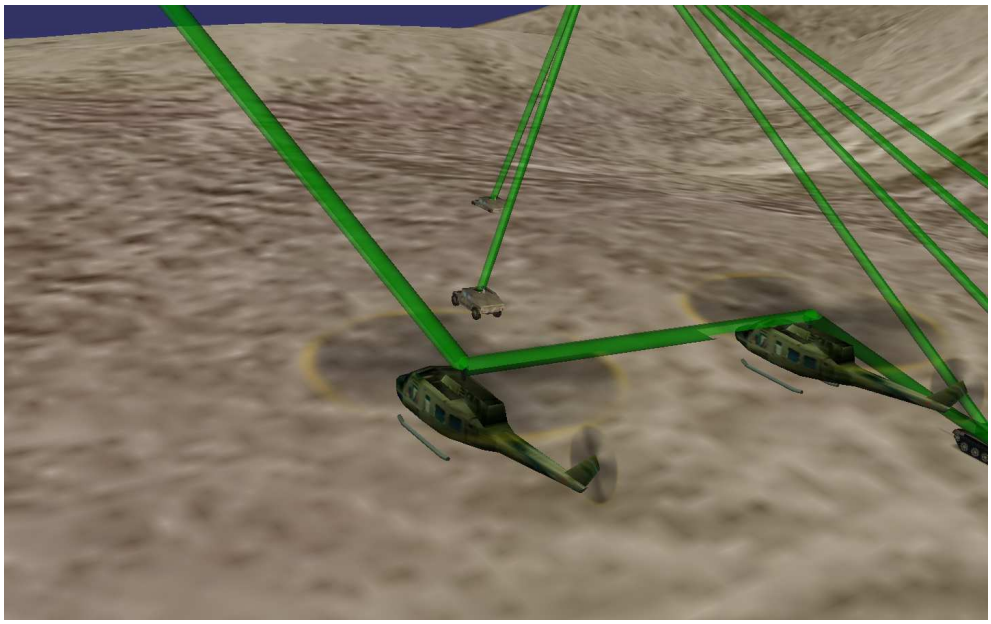
extensive programming. Positioning nodes and loading terrain were straightforward through OSG's intuitive use of the Cartesian coordinate system for populating the scene. OSG's built-in functions for efficiently rendering geometric shapes supported the desire to animate cylinders and spheres to indicate data flow in the network. Finally, since OSG supports numerous 3D model file types, finding example models to use in the scene was also straightforward.

4.3 Simulator/Visualization Synchronization

The simulation is synchronized with the visualization using the three step process as outlined in Section 3.3. First the scene is created using the 3D tool, including placement of network assets on the desired terrain. Communication links can then be populated between assets as desired by the user, allowing the user to examine the links for possible obstacles due to hills, trees, or other objects. This allows the user to reposition assets or add new assets to ensure successful communication. Next this scene is recreated in 2D in NS-2, preserving spatial relationships as much as possible. Each network node in the 3D scene must have a corresponding node in the 2D simu-



(a) The helicopter in the foreground blocks view of the second Humvee



(b) By repositioning the camera, the second Humvee can be seen

Figure 22: These images demonstrate how camera control can help alleviate node occlusion

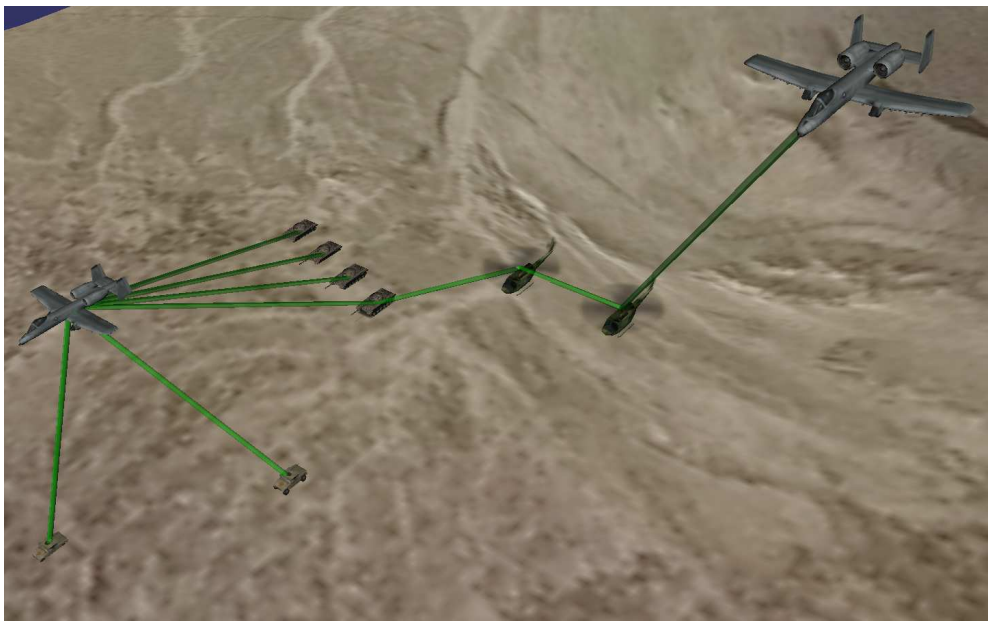


Figure 23: A populated 3D scene featuring assets, terrain, and communication links

lation. After the simulation is executed, the trace file can then be loaded into the 3D scene for visualization of network traffic. Each of these steps is explored in further detail in the following sections.

4.3.1 Scene Creation and Validation. Creating the scene requires the user to define the number, type, and location of battlefield assets to create the desired scenario. Once the scene is populated with the appropriate assets, the user must define animation paths if necessary for any mobile nodes in the network. Finally, communication links can be drawn between communication nodes, allowing the user to investigate the scene for potential collisions. After the scene is created, the user can visually inspect the scene to verify it meets the user's needs. Figure 23 depicts a populated scene with tanks, Humvees, helicopters and A-10s positioned on a realistic terrain, connected with desired communication links. It can be seen that no communication links are hindered by terrain features or other objects.

4.3.1.1 Asset Population. Network assets in the 3D scene are populated based on user input. The user is able to specify the number, type, and location

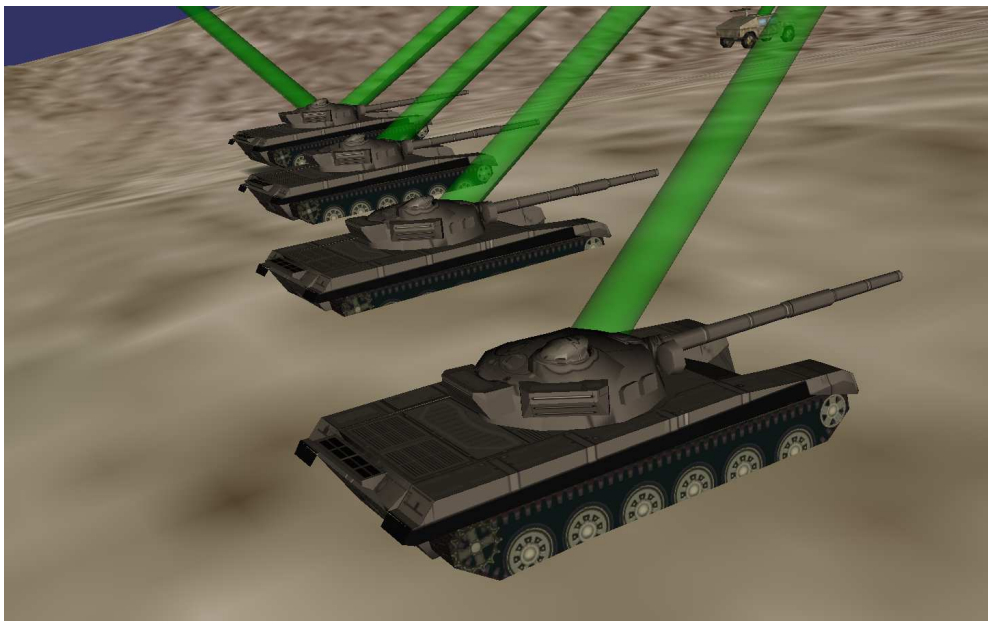


Figure 24: These tank assets are partially underground

of assets as well as terrain for the scene. Node location is specified by Cartesian coordinates (*X, Y, Z*). Ground nodes are identified by a *Z* component of zero, and these assets are automatically positioned just above the terrain. Due to hills and valleys in different terrains, this requires updating the *Z* component to be higher or lower as necessary. Figure 24 demonstrates what would happen if the *Z* component is not automatically adjusted. The tanks in this figure are located partially underground. Air and sea assets are positioned using all three Cartesian coordinates to allow air assets to be positioned high above the terrain and sea assets to be partially submerged in a body of water in the terrain.

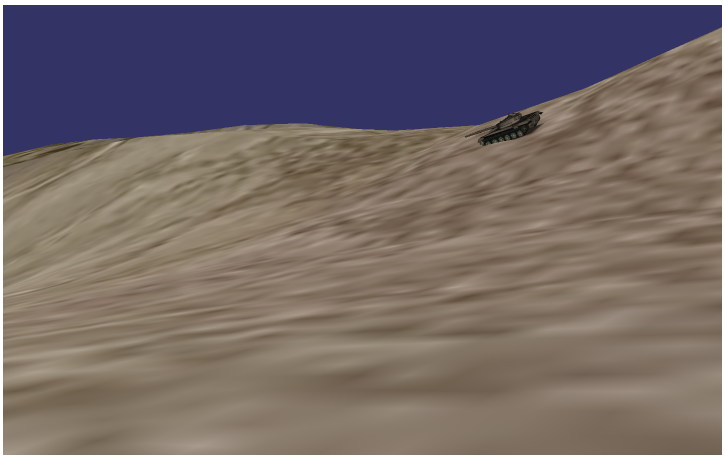
4.3.1.2 Animation Paths. The user is also able to define circular animation paths for mobile assets such as airplanes and tanks. These paths are defined by the Cartesian coordinates of the center of the circular path, the desired radius of the circle, and finally the time for one revolution, which determines the speed of the mobile asset. Ground nodes automatically have their *Z* component updated to allow them to traverse the scene over terrain features, as seen in Figure 25. In this

figure, the tank is automatically adjusted to follow the terrain, including tilting when required.

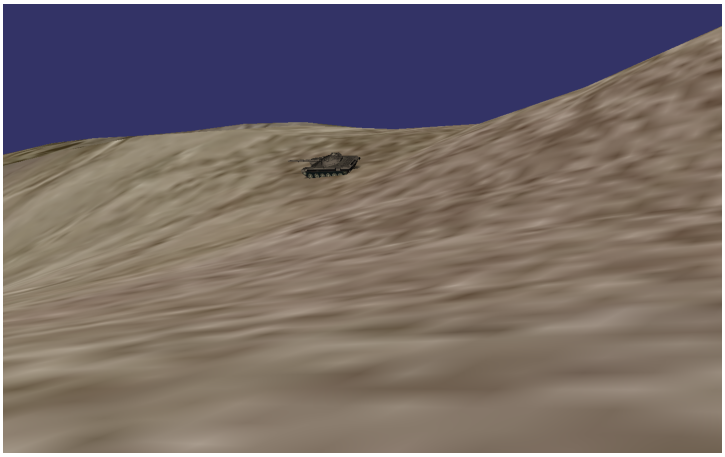
4.3.1.3 Communication Links. Depending on the selected terrain, there is the potential for features such as mountains or valleys to impede communication. In these circumstances, the user can populate desired communication links between nodes and visually inspect the scene for collisions. In the event that a link is blocked by a mountain, tree, or other feature, the user can then reposition the network node if desired or add an additional asset to route communication around the object. If the user does not wish to reposition nodes, the visualization will still execute normally, though it may be visually confusing. For example, data packets may disappear into a mountain and then reappear on the other side. Displaying communication links before simulation assists the user in identifying possible intersections, which can then be removed by fine-tuning the scenario.

Visualizing potential communication links prior to simulation adds further credibility to the network simulation. The user can ensure that the NS-2 simulation is built only using those links that are not hindered by other scene objects. This provides a significant analysis tool for the user, since battlefield scenarios can be inspected for communication barriers before being simulated and deployed. For example, Figure 26 depicts a simple example of signal blockage due to terrain interference. In this scene, two tanks are on opposite sides of a hill and cannot effectively communicate. Upon seeing this, the user can add an air asset to the scenario to allow the tanks to communicate successfully.

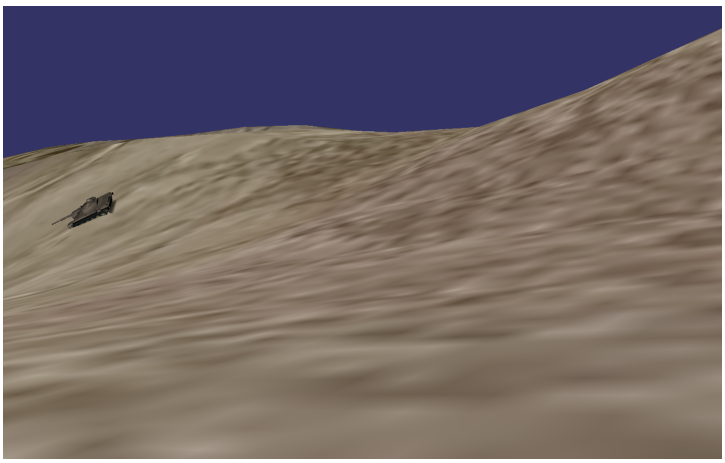
4.3.2 Simulation. After the scene has been defined by the user, the NS-2 simulation must be created, maintaining asset position and animation. Based on the 3D scene described previously, seen in Figure 23, an equivalent 2D network is built in NS-2. This 2D network can be seen in Figure 27, shown side by side with the 3D scene. Node placement in this 2D network is roughly equivalent to that in the 3D scene, and the communication links have been drawn accordingly.



(a)

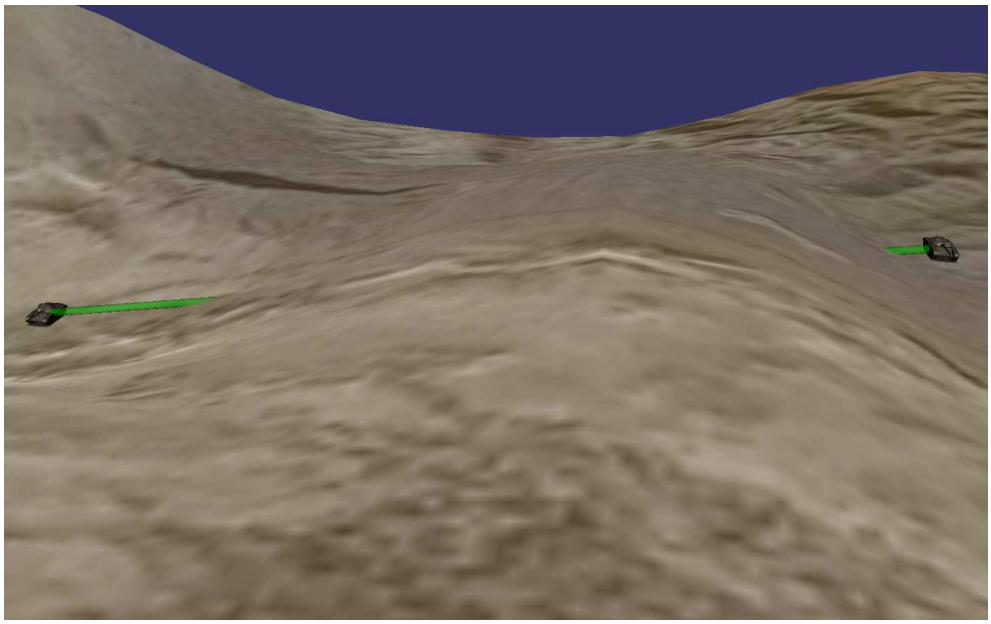


(b)

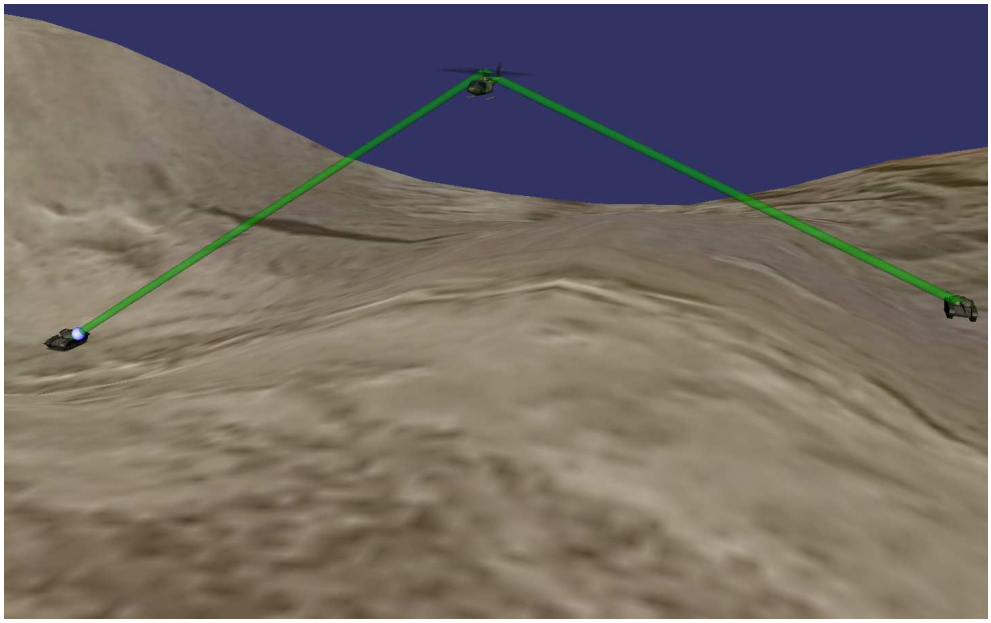


(c)

Figure 25: This series of images depicts a mobile tank asset traveling through the scene, automatically adjusting for the changing terrain



(a) These two tank assets cannot communicate due to terrain blockage



(b) By adding an additional air asset, communication can be restored

Figure 26: Populating communication links prior to simulation allows the user to discover scenarios in which assets cannot communicate due to terrain blockage

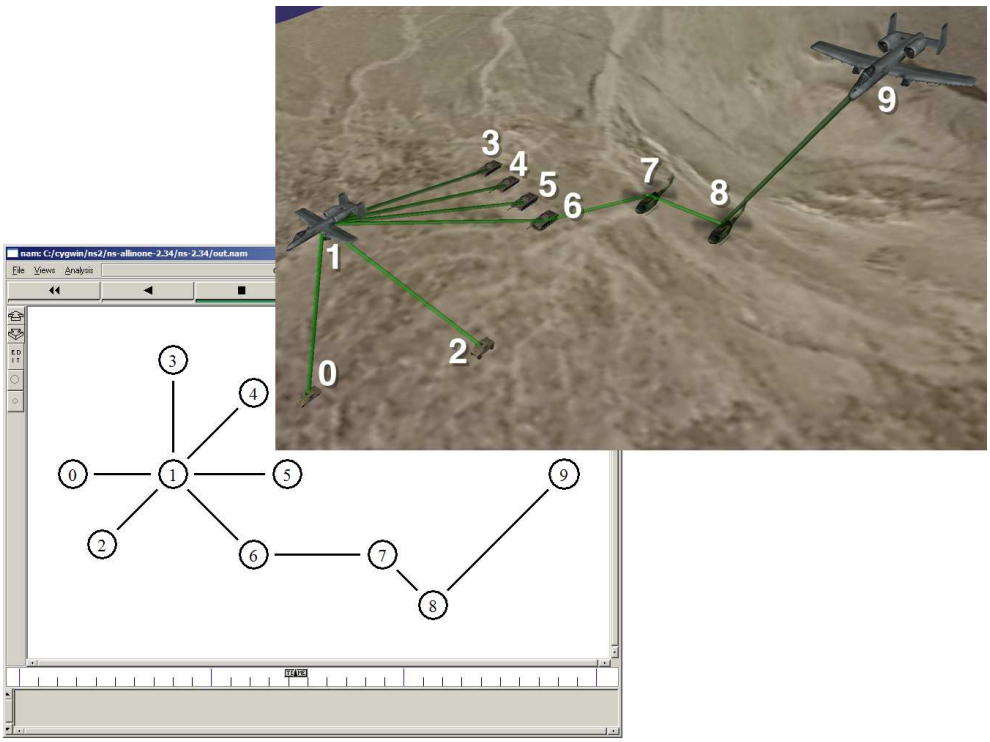


Figure 27: There is a one-to-one mapping from the 2D NS-2 simulation to the 3D visualization

The NS-2 simulation used to generate the input file for the 3D visualization is fairly trivial in nature, though it is sufficient to show the capability of the tool. Latencies are defined between nodes that roughly reflect the relative positions of assets in the 3D scene. For example, the two helicopters have the shortest latency between them since they are closest together. For the sake of simplicity, each link is defined to be a full duplex communication path with a bandwidth of 5 megabits per second, and network packet size is fixed at one thousand bytes. These parameters can easily be changed in the simulation to accurately reflect real-world operations, but they are adequate for the sake of demonstration.

Traffic is generated by 4 nodes, with 4 separate designations, generating traffic along all possible links. Figure 28 shows each of these 4 network flows in a different color. Firstly, traffic is generated by one Humvee (node 0), destined for the distant A-10 (node 9), requiring the nodes along the way to forward the traffic as necessary. Secondly, traffic is generated by a different Humvee (node 2), destined for one tank

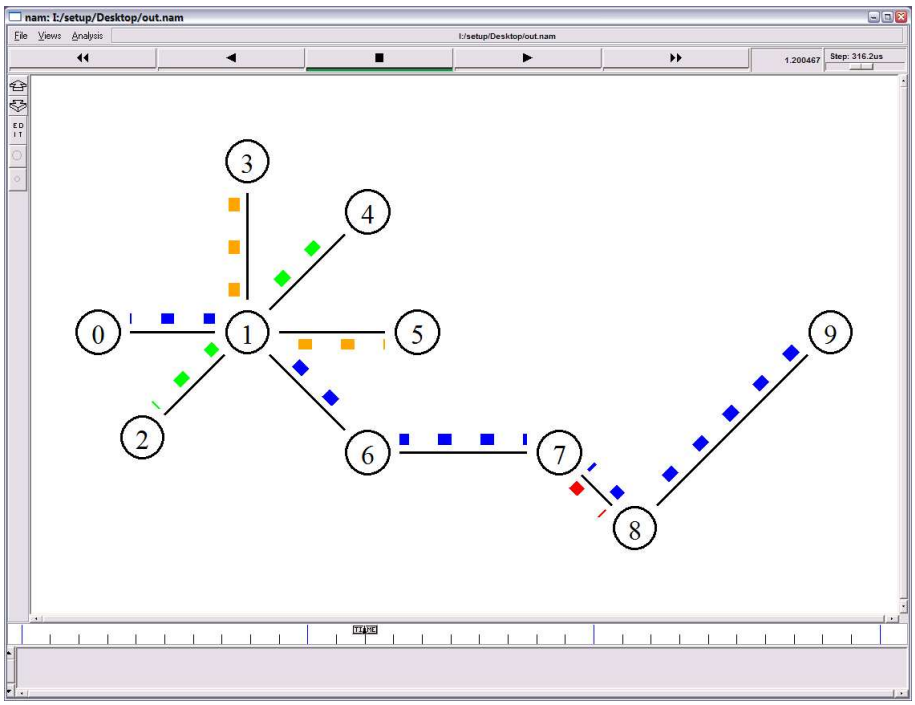


Figure 28: Color-coded Nam visualization of network traffic paths

node (node 4). Thirdly, traffic is generated by a different second tank (node 5) with destination a third tank (node 3). Finally, traffic is sent from one helicopter to the other (node 8 to node 7). The trace file from this simulation will next be imported into the 3D scene for network visualization.

4.3.3 Network Visualization. The final step is to import the NS-2 trace file into the created 3D scene, which allows the user to see the resulting network traffic flows. The 3D visualization tool will interpret the trace file and automatically animate packets based on the selected animation scheme. The standard animation scheme simply animates a visualized packet for each simulated packet, as depicted in Figure 29. This figure shows data packets traveling along each communication link from two different camera angles, defined by the events in the simulation trace file.

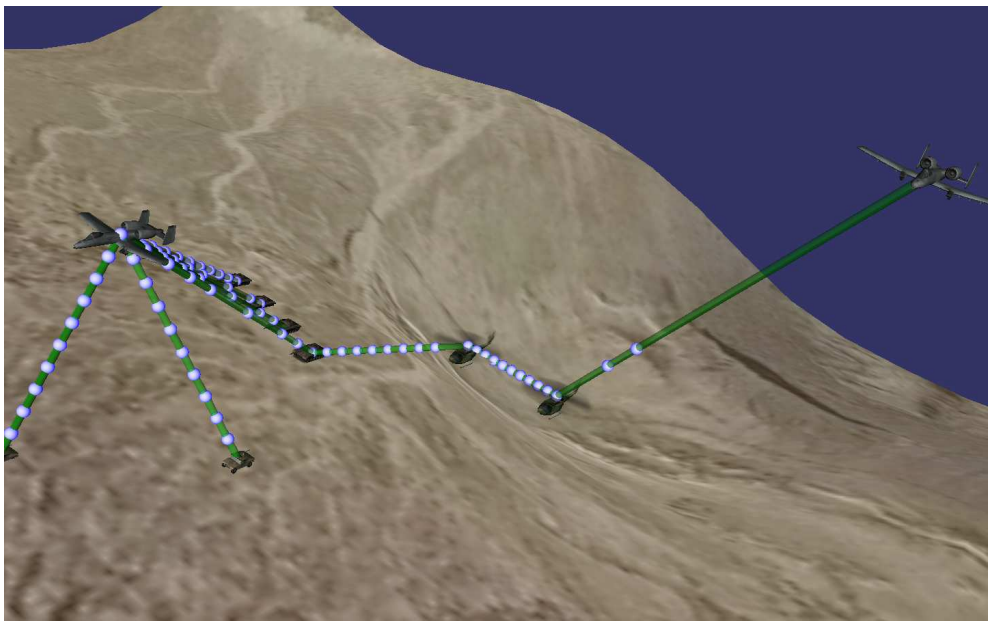


Figure 29: This image depicts network traffic in the 3D scene, driven by the trace file generated during the network simulation

4.4 Measures of Success Revisited

In Chapter III, Section 3.4, measures of success were outlined to answer the following two questions: Is the created visualization an effective *information visualization*? Is the created visualization an effective *communication network visualization*? Each of these questions is explored in the sections that follow, based on previously identified criteria.

4.4.1 Effective Information Visualization. Before delving into specific computer communication network aspects, it is first necessary to determine whether the created 3D visualization is effective at conveying information. Reprinted from Section 3.4.1, the following seven measures of success are explored one by one in the sections that follow. Evaluating information visualization is subjective in nature, though every attempt has been made to be as objective as possible.

1. Allow users to have control of time and space [19]
2. Facilitate narration of events [10]

3. Stand on its own [15]
4. Be based on non-visual data [15]
5. Create a result that is readable and recognizable [15]
6. Adapt to serve multiple needs [34]
7. Make effective use of spatial layout and grouping [6]

4.4.1.1 User Control of Time and Space. User control of the scene camera was previously discussed in Section 4.1.6, though there was no discussion of user control over the time domain. Firstly, the user is free to move the camera to any position in the scene, which effectively solves the previously discussed node occlusion problem, and also allows the user to view the network in any manner desired. For example, the user may wish to view the network in a 2D manner, which can be done by positioning the camera in a top-down overhead view, provided that no asset is directly above another asset. This type of 2D view can be seen in Figure 30. This view provides a bird's eye view of the network scene, though elevation differences are obviously lost. However, for most scenarios a 3D view is more appropriate, since not all scenes lend themselves to this viewpoint. Since nodes are often mobile as the scene executes, it is necessary for the user to adjust the scene camera to track these movements. To fully evaluate the 3D battlefield scenario, the view is free to rotate, pan, and zoom throughout the scene and investigate the relationships between nodes.

Secondly, the user would ideally have full control over the time domain, with the ability to fast forward, rewind, and pause as required to understand the network. However, this goal was not achievable in the permitted time frame. The network animation simply displays network events in sequential order, and does not allow the user to rewind or pause the scene. This capability could be added in a future version, but is not functional at this time.

4.4.1.2 Narration of Events. The created visualization does an excellent job at facilitating story-telling. Since the visualization itself conveys a vast

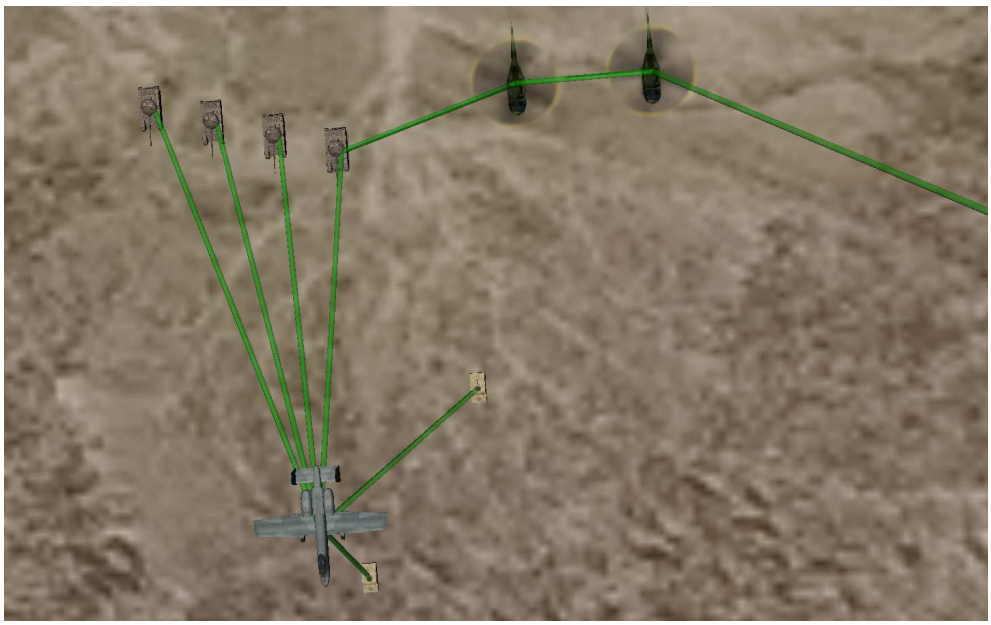


Figure 30: An overhead view gives a 2D representation of the network

amount of information, such as terrain conditions, deployed assets, and the geo-location of each asset, the presenter's job is made much easier. The presenter is therefore free to dive more deeply into the details of the operation, without having to give an extensive explanation of the scene itself. Communication officers can focus more of their effort into explaining the data transmission in the scene and those being briefed can quickly comprehend what is happening. The 3D visualization tool is a great communication aid, which enables effective and efficient narration of battlefield events.

4.4.1.3 Independence. Even without an explanation, the animated scene conveys a story of a battlefield communication scenario. Since nodes are represented by their real-life counterparts, the user does not need to reference a legend to determine which node is which. It is fundamentally clear which nodes are airborne and which nodes are currently communicating. The scene can be understood on its own, without the need to reference or understand the original NS-2 trace data. In summary, the scene stands on its own.

4.4.1.4 Based on Non-visual Data. As was previously seen in Figure 15, the NS-2 output trace data is highly non-visual data. For someone unfamiliar with NS-2, the trace data resembles a random string of letters and numbers. For someone that is familiar with NS-2, the network events can be followed, though doing so is extremely tedious. The created visualization takes this non-visual data and represents it in an intuitive, visual manner.

4.4.1.5 Readable and Recognizable. Since the created 3D visualization uses real-world models and terrain, the resulting image has strong visual credibility. This allows the user to process the battlefield scenario with little conscious thought, since the majority of data is processed by the human visual system. The scene highly resembles a real-world scene, which results in a visualization that is easily readable and recognizable.

4.4.1.6 Adaptability. The created visualization as designed is useful for depicting network events on the battlefield, though it can be altered to suit other needs. Firstly, when used as designed, the 3D visualization can be used to represent any number of battlefield scenarios, from small-scale operations to large-scale operations involving dozens of assets. In the event that network communication is not of interest, the tool is also useful for visualizing battlefield operations alone. The tool in its current state is useful for getting an overview of the battlefield, though could be useful for simulating full operations with the addition of more robust mobility functionality. Aside from these military applications, the 3D models can easily be replaced with any number of other models to create a non-military scene as desired by the user. This flexibility in the created visualization tool shows that is adaptable to serve multiple needs.

4.4.1.7 Spatial Layout and Grouping. Spatial layout is important to any visualization and is especially important when accurately depicting a realistic battlefield scene. The created 3D visualization allows assets to be positioned exactly

as they would be in the real world. Retaining the interspatial relationships between network nodes results in a scene that is easily understood by the user.

4.4.2 Effective Communication Network Visualization. Aside from determining whether the created 3D visualization tool is effective at information visualization, it is also necessary to determine whether it is effective at network visualization. The three following measures of success were established in Section 3.4.2 and are reprinted here for ease of reference. These measures of success are explored one by one in the sections that follow. Once again, every effort has been made to be as objective as possible, though evaluating visualization is highly subjective.

1. Accurately display network simulation events [34]
2. Be efficient with regard to data display [34]
3. Provide equivalent capability as existing visualizations

4.4.2.1 Accuracy and Efficiency. The first two measures of success for network visualization will be discussed in this section, primarily because accuracy and efficiency are inversely related. Providing a more accurate network visualization results in an overcrowded, busy scene, while a visualization with high data efficiency lacks detail. The standard animation scheme mimics the packet animation in Nam, in which every simulated packet has a corresponding visualized packet. This results in a network visualization that is highly accurate with regard to network traffic, as shown in Figure 31. In this figure, network data packets are flowing from each source to the destination specified by the network simulation trace file.

The standard animation scheme is not without drawbacks, however. It can be potentially overwhelming to the viewer, especially when combined with mobile nodes. For the network analyst that needs all details of ongoing communication, this animation scheme works very well since it is the most accurate. However, for a user that wants a high level overview, this animation scheme becomes incredibly busy for anything but the lightest of traffic. In a scenario in which there is a high volume

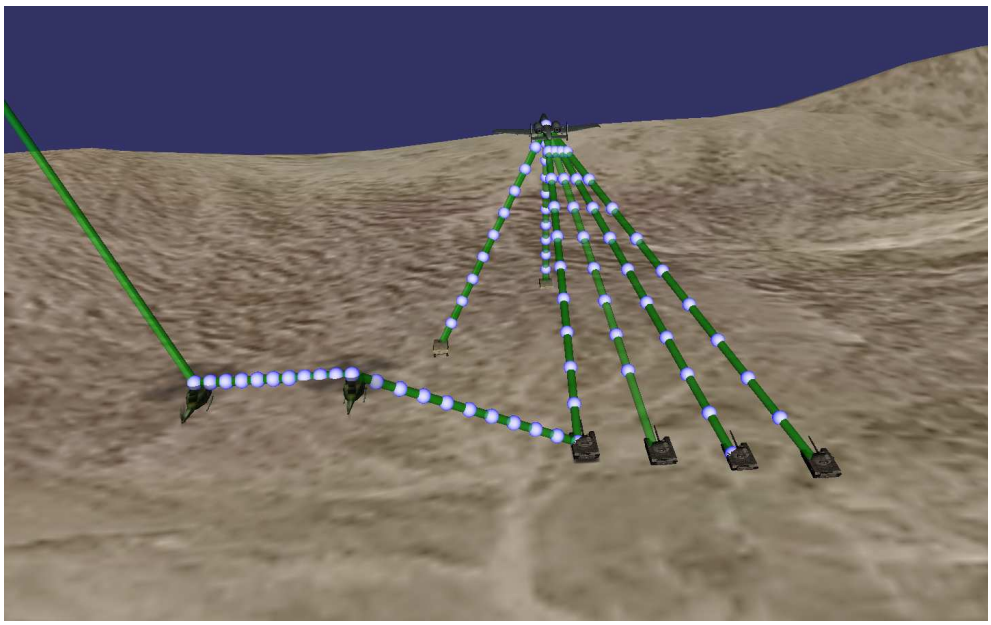


Figure 31: This image demonstrates the standard animation scheme, in which packets are flowing from source to destination

of traffic flowing over each link, the links can potentially become overpopulated to the point that it becomes hard to determine the direction traffic is flowing. In the event of full duplex links, in which traffic is flowing in a bidirectional manner, it is extremely difficult to follow the communication flow amongst network entities. Figure 32 demonstrates this problem, in which it can be seen that communication links become saturated to the point of appearing as solid white lines.

The standard animation scheme is the first step in successfully combining 3D network visualization with packet animation. The standard scheme is useful for scenarios featuring light data traffic, but alternate packet animation schemes are needed for high traffic scenarios.

4.4.2.2 Comparison to Existing Visualizations. Table 2 shows a comparison of this research with other network visualization tools. It is important to note that these tools were evaluated with a battlefield scenario in mind. For example, Nam and iNSpect are useful for describing the state of a network, but they are poor tools for describing a battlefield scenario in particular due to a lack of visual credibility.

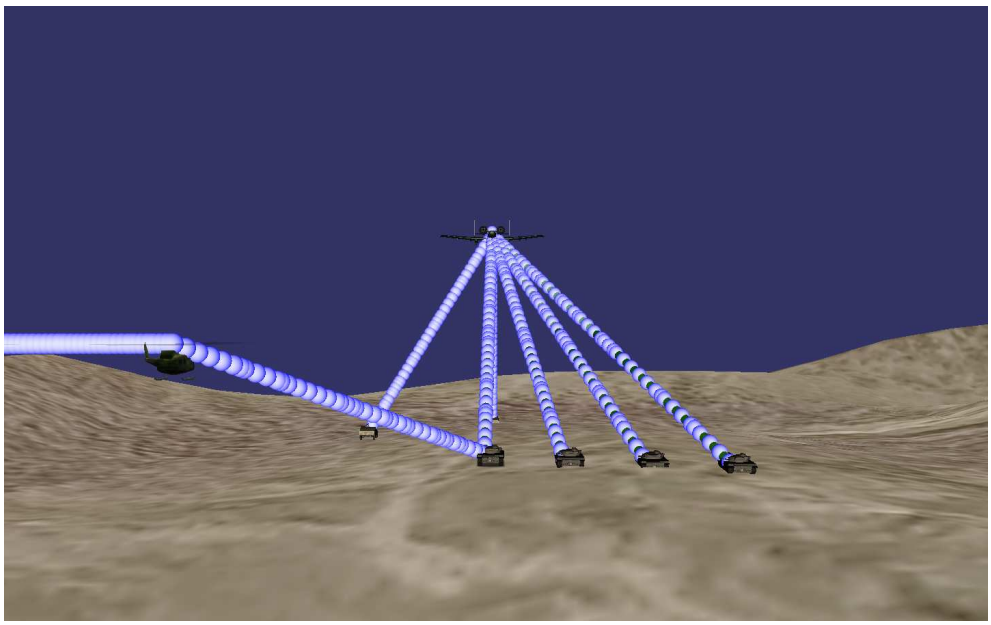


Figure 32: In instances of extremely high network activity, communication links can become saturated, which increases the difficulty of interpreting the visualization

NetViz is slightly better since it uses military assets to represent network nodes, but the 3D tools provide the most clear scenes of battlefield environments, requiring minimal explanation. Similarly, the 2D tools Nam, iNSpect, and Netviz stand on their own for general network visualization, but in the case of battlefield scenarios, they require extensive explanation compared with the 3D tools. This table summarizes the key differences between the tool created as a result of this research; however, more detailed comparisons to Nam and OPNET 3DNV follow in subsequent paragraphs.

Nam. Since the created visualization tool runs on an output file intended for Nam, it is the easiest 2D tool to compare against, though most of the comparisons apply to the other 2D visualizations as well. The visualization tool created through this research compares favorably with Nam with regard to packet animation and realism and unfavorably with regard to secondary network traits and user control over time. Firstly, the packet animation in the standard animation scheme in the created tool is equivalent to that of Nam; each simulated packet generates a visualized packet. On the other hand, Nam shows current queue status at each node

Table 2: Comparison of this research with other network visualization tools

	Nam	iNSpect	NetViz	OPNET 3D	This Research
High Visual Credibility				✓	✓
Realistic Assets			✓	✓	✓
Realistic Terrain				✓	✓
Realistic Mobility				✓	✓
Packet Animation	✓		✓		✓
Current Queue State	✓		✓		
Collision Detection				During Sim	Prior to Sim
User Time Controls	✓	✓	✓	✓	
User Camera Controls	N/A	N/A	N/A	✓	✓
Facilitates Story Telling			✓	✓	✓
Stands on its Own				✓	✓

and indicates when a packet is dropped due to a full queue. The created tool is incapable of this in its current state. From a perspective of realism, Nam's limited 2D representation of the network is obviously incapable of matching the 3D visualization tool's visually credible battlefield scene. Lastly, the time slider in Nam is an extremely useful feature when analyzing data networks; the 3D visualization tool would greatly benefit if it had similar capability.

OPNET 3DNV. Comparisons to OPNET's 3DNV suite are necessary, simply because it was created for many of the same reasons. Considering the commercial nature of OPNET, it is interesting that OPNET's 3DNV package makes no attempt to animate packets. Faced with the difficult task of balancing animated packets with mobile assets, OPNET chose to avoid this problem altogether by only visualizing connectivity. From a realism perspective, the visualization tool created as part of this research and the OPNET tool are similar. Realistic assets are positioned on a terrain in much the same manner, and links are drawn between communicating nodes in a similar manner. The ability to view animated packets gives the tool created by this research a significant advantage over the OPNET tool, which only displays connectivity. The OPNET tool is sufficient for a high-level overview of battlefield communications, but digging into the details of the scene requires the ability

to investigate traffic on a packet level, which is not possible with the OPNET tool. One area that distinguishes the OPNET tool from the tool created by this research is the handling of collision detection. In the OPNET tool, collisions are identified as the simulation executes, whereas in the tool created by this research, collisions can be found prior to simulation during the scene creation and validation step. Identifying collisions early reduces the need for running multiple simulations, which can save time for complex scenarios. Overall the 3D visualization tool created by this research shares much of the same functionality as OPNET 3DNV. Being a commercial product, the OPNET tool is more user friendly and robust. The tool created by this research is more expandable and customizable to suit the user's needs, since the source code can be directly modified.

4.4.3 Discussion. In the previous two sections, we have seen that the created visualization meets most of the proposed measures of success. In the area of information visualization, the created 3D visualization fully satisfies six of the seven criteria and partially satisfies the remaining criteria, only lacking with respect to user control over the time domain. This remaining criteria could be solved with further software development.

The standard packet animation scheme is very accurate with regard to network simulation events, but not efficient with regard to data display during high traffic scenarios. This is an area that warrants further development, by implementing the other packet animation schemes outlined in Chapter 3. Accuracy and efficiency are two sides of the same coin, and it is often difficult to increase one without decreasing the other. However, by implementing many options, the user would be able to explore which scheme is most useful for a given scenario.

4.5 Summary

The examples in this chapter illustrate the capabilities and functionality of the created 3D network visualization tool. The result is a flexible visualization tool that

can be used for network animation as well as other applications. The rendered scene has a high degree of visual credibility, allowing the user to quickly process the scene visually. The tool could be improved through the implementation of time controls for the user and the implementation of additional animation schemes, but it is still a useful tool despite these shortcomings. The tool compares favorably with existing 2D network visualizations and shares many features of OPNET 3DNV. The tool needs further refinement, but satisfies the core functionality of network visualization. Based on the previously established criteria, the created 3D network visualization is both effective at information visualization in general, and more specifically, communication network visualization. Furthermore, by developing this tool in an open source environment, the door is opened for future network visualization research and this research provides a significant contribution to the network visualization research community.

V. Contributions and Future Work

This research provides three significant contributions with regard to network visualization. Firstly, the created tool is an excellent communication aid that facilitates effective communication. Displaying the battlefield network scenario in a realistic and intuitive manner shortens the time required for the presenter to explain the scene. This means a communication planner, presenting the scenario to the commander, has more time to address issues such as terrain masking, packet loss, node movement, or other areas. Secondly, this research explores combining 3D network visualization with real-time network packet animation. This packet animation provides a significant capability to communication planners by allowing them to see all ongoing network traffic in a realistic, 3D manner. Lastly, this research provides a significant contribution to the network visualization community through its flexible 3D visualization tool. This tool can not only be used to represent battlefield communication as discussed, but can be expanded to represent other 3D network communication needs, such as mobile ad hoc or sensor networks.

The 3D visualization tool currently provides a significant capability, though it can be further improved by new features and enhancements to existing features. In particular, user control over the time domain would greatly improve the tool, as would more robust asset mobility. Allowing the user to move forward and backward in time would facilitate the inspection of network events, while allowing assets to move in a linear fashion would allow for a more accurate representation of the real world. A Heads-Up Display (HUD) could provide additional capability by displaying current network statistics or scene data. For example, the average packets transmitted per second could be displayed in one corner of the screen, providing a metric for current network load. Another area for future development is to implement the other packet animation schemes described in Chapter 3, including the addition of color to the proposed schemes. Links could be color coded depending on current utilization of its bandwidth, or packets could be color coded based on size. These are just a few of the ways the tool could be improved by future enhancements.

5.1 Contributions

5.1.1 Communication Aid. The created 3D network visualization tool directly supports military personnel by providing a tool that enables effective and efficient communication. The tool can be used to explain operational scenarios in a realistic, visually credible manner. The visualization carries a lot of visual information, such as terrain conditions, deployed assets, and animation paths for each asset. The visualization helps to paint a clear picture of the battlefield, allowing communication officers to focus on the details of the ongoing data transmission, rather than explaining the scene itself.

5.1.2 Packet Animation. OPNET 3DNV provides many of the same features as the tool created for this research, though it avoids the problem of effective packet animation by only displaying connectivity. This may be sufficient for a high level overview of a battlefield scene, but a communication planner may need further detail. This research investigates the benefits of combining 3D network visualization with packet animation and the resulting effects on accuracy and efficiency. The packet animation scheme explored in this research provides a solid foundation for the investigation of more complex animation schemes.

5.1.3 Network Visualization. This research benefits the network visualization community by providing an open-source, flexible, 3D network visualization tool that can be tailored to meet many needs beyond the military applications discussed. The tool could be used to visualize any type of network communication, especially those that benefit from a 3D view. Mobile ad hoc networks as well as wireless sensor networks could benefit from a 3D network visualization, where distance between communicating nodes and any interfering obstacles could be examined.

5.2 Future Work

5.2.1 Time Controls. The visualization tool would benefit most from the implementation of time controls for the user. This would allow the user to move forward and backward in time as necessary to investigate significant network events. This would require adjusting the current position in the NS-2 trace file as well as determining the current location of network assets. This issue is fairly straightforward in concept, but presents many problems when trying to implement. For example, rewinding the simulation requires determining the current state at a prior point in time, which includes both the current NS-2 simulation events as well as any previous events that would affect the visualization. Determining how far back to go in the NS-2 trace file is a difficult problem to solve.

5.2.2 Linear Mobility Paths. The created visualization allows assets to be mobile along circular paths, but the tool could be improved by implementing a sequential waypoint system. This system would allow assets to move along linear paths to each subsequent waypoint and allow for more complex animation paths. Adding this capability would help the tool better mimic realistic battlefield scenarios by allowing assets to pursue multiple disparate goals.

5.2.3 Heads-up Display. The scene could be further enhanced by displaying real-time network statistics based on the current and past network state. Statistics can be displayed by using OSG's built-in text display functions, creating a useful HUD. This allows the user to see things such as throughput, end-to-end delay, and packet loss statistics as the scene executes. A compass could be added to one corner, helping the user remain oriented as the scene events unfold. The average packets transmitted per second could be displayed in one corner of the screen, providing a metric for current network load. There are many possibilities that would be created through the addition of a HUD to the visualization.

5.2.4 Other Packet Animation Schemes. Aside from the standard packet animation scheme that was implemented, there is room for improvement through the implementation of the other schemes outlined in Chapter 3. The visualization tool created by this research can be greatly expanded upon by exploring other animation schemes or by creating a hybrid of two schemes. Aside from creating completely new animation schemes, there is also the possibility of adding color to the links and packets in the visualization. Communication links could be color coded, allowing the user to quickly estimate the amount of bandwidth currently being used on a particular link. Packets could be displayed in color, with different colors indicating different packet size thresholds. For example, a red link could indicate that the channel is using its maximum available bandwidth, while a blue link could indicate a relatively free communication channel. The core functionality of the tool is already in place, so exploring additional packet animation schemes should be a straightforward endeavor.

5.3 Conclusion

This research has shown the unique capabilities when displaying a battlefield scenario in a realistic 3D manner and the advantages over 2D visualization techniques for these types of scenarios. Though it is in its beginning stages, this visualization provides a promising capability that can be further expanded upon in the future. Expanding this visualization can provide a robust framework for accurate portrayal of battlefield scenarios, which becomes more important as the battlefield becomes increasingly connected.

Bibliography

1. Autodesk. "3ds Max - 3D Modeling, Animation, and Rendering Software". URL `http://usa.autodesk.com/adsk/servlet/pc/index?id=13567410&siteID=123112`. January 19, 2010.

2. Belue, J. Mark. *Network Visualization Design Using Prefuse Visualization Toolkit*. Master's thesis, Air Force Institute of Technology, March 2008.

3. Belue, J. Mark, Stuart H. Kurkowski, Scott R. Graham, Kenneth M. Hopkinson, Ryan W. Thomas, and Joshua W. Abernathy. "Research and Analysis of Simulation-based Networks through Multi-Objective Visualization". *Winter Simulation Conference*, 1216–1224, 2008.

4. Chen, C. "Top 10 unsolved information visualization problems". *Computer Graphics and Applications, IEEE*, 25; 25(4):12–16, 2005.

5. Cubic Defense Applications. "One Common Data Link Solution". URL `http://www.cubic.com/cda1/Prod_&_Serv/C4ISR_Prod_&_Sys/Data_Links/index.html`. January 22, 2010.

6. Dunne, Cody and Ben Shneiderman. *Improving graph drawing readability by incorporating readability metrics: a software tool for network analysts*. Technical Report HCIL-2009-13, 2009. URL `http://www.cs.umd.edu/localphp/hcil/tech-reports-search.php?number=2009-13`. December 9, 2009.

7. Dzambaski, Aleksander, Dimmitar Trajanov, Sonja Filiposka, and Aksenti Grnarov. "Ad hoc networks simulations with real 3D terrains". *15th Telecommunications Forum 2007*, 95–98, November 2007.

8. Elmqvist, N. and P. Tsigas. "A Taxonomy of 3D Occlusion Management for Visualization". *Visualization and Computer Graphics, IEEE Transactions on*, 14(5):1095–1109, Sept.-Oct. 2008. ISSN 1077-2626.

9. Estrin, Deborah, Mark Handley, John Heidemann, Steven McCanne, Ya Xu, and Haobo Yu. "Network Visualization with nam, the VINT Network Animator". *Computer*, 33(11):63–68, 2000.

10. Few, Stephen. "What Ordinary People Need Most from Information Visualization Today", August 2008. URL `http://www.perceptualedge.com/articles/visual_business_intelligence/what_people_need_from_infovis.pdf`. December 5, 2009.

11. Hertlein, Ulrich. "OpenSceneGraph". URL `http://www.sandbox.de/osg/`. January 19, 2010.

12. House, D. H., A. S. Bair, and C. Ware. "An approach to the perceptual optimization of complex visualizations". *Visualization and Computer Graphics, IEEE Transactions on*, 12; 12(4):509–521, 2006.

13. Hunt, Andy and Thomas Hermann. "The Importance of Interaction in Sonification". Stephen Barrass and Paul Vickers (editors), *Proceedings of the Int. Conference on Auditory Display (ICAD 2004)*. International Community for Auditory Display (ICAD), International Community for Auditory Display (ICAD), Sydney, Australia, 7 2004. ISBN 1-74108-048-7.

14. Kitware, Inc. "The Visualization Toolkit". URL `http://www.vtk.org`. January 14, 2010.

15. Kosara, R. "Visualization Criticism - The Missing Link Between Information Visualization and Art". *Information Visualization, 2007. IV '07. 11th International Conference*, 631–636, 2007.

16. Kurkowski, Stuart, Tracy Camp, Neil Mushell, and Michael Colagrosso. "A Visualization and Analysis Tool for NS-2 Wireless Simulations: iNSpect". *IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems*, 503–506, 2005.

17. Lau, A. and A. V. Moere. "Towards a Model of Information Aesthetics in Information Visualization". *Information Visualization, 2007. IV '07. 11th International Conference*, 87–92, 2007.

18. Martz, Paul. *OpenSceneGraph Quick Start Guide*. Skew Matrix Software LLC, 2007.

19. Nakakoji, K., A. Takashima, and Y. Yamamoto. "Cognitive effects of animated visualization in exploratory visual data analysis". *Information Visualisation, 2001. Proceedings. Fifth International Conference on*, 77–84, 2001.

20. National Defense and the Canadian Forces. "Joint Fires Support". URL `http://www.cfd-cdf.forces.gc.ca/sites/page-eng.asp?page=1218`. January 22, 2010.

21. North, C. "Toward measuring visualization insight". *Computer Graphics and Applications, IEEE*, 26; 26(3):6–9, 2006.

22. Olson, Curtis L. "FlightGear Flight Simulator". URL `http://www.flightgear.org/Downloads/aircraft/`. January 19, 2010.

23. OpenSceneGraph. "OSG Introduction". URL `http://www.openscenegraph.org/projects/osg/wiki/About/Introduction`. December 3, 2009.

24. OPNET Technologies, Inc. "OPNET Modeler 3D Network Visualization". URL `http://www.opnet.com/solutions/network_rd/high_fidelity_modeling.html#3d`. December 5, 2009.

25. OPNET Technologies, Inc. "Teaching and Research with OPNET: The Military Academy Perspective". URL http://www.opnet.com/events/webinars/webinar_archived/index.html. December 8, 2009.

26. OPNET Technologies, Inc. "OPNET Modeler Documentation", 2008.

27. Picard, R. W. "Affective Computing for HCI". *Proceedings HCI, 1999, Munich, Germany*, 1999. URL http://affect.media.mit.edu/pdfs/99.picard-hci.pdf. November 29, 2009.

28. Rappaport, Theodore S. *Wireless Communications Principles and Practice*. Prentice Hall, 2nd edition, 2001. ISBN 978-0-13-042232-3.

29. Robertson, George, Jock Mackinlay, and Stuart Card. "Information Visualization Using 3D Interactive Animation". *Communications of the ACM*, 36(4):57–71, 1993, April 1993.

30. Scheuermann, Björn, Holger Füβler, Matthias Transier, Marcel Busse, Martin Mauve, and Wolfgang Effelsberg. "Huginn: a 3D visualizer for wireless ns-2 traces". *MSWiM '05: Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, 143–150, 2005.

31. Schroeder, William J., Kenneth M. Martin, and William E. Lorenson. "The Design and Implementation of an Object-Oriented Toolkit for 3D Graphics and Visualization". URL http://www.vtk.org/VTK/img/dioot.pdf. January 14, 2010.

32. Sullivan, Joseph A. "OpenSceneGraph Tutorials". URL https://www.movesinstitute.org/Sullivan/OSGTutorials/. January 19, 2010.

33. Turbo Squid, Inc. "3D Models, 3D Modeling Textures and Plugins". URL http://www.turbosquid.com. July 14, 2009.

34. Ward, M. "Overview of Data Visualization". URL http://web.cs.wpi.edu/~matt/courses/cs563/talks/datavis.html. November 15, 2009.

35. van Wijk, Jarke J. "Views on Visualization". *IEEE Transactions on Visualization and Computer Graphics*, 12(4):1000–433, 2006. ISSN 1077-2626.

36. Wright, W. "Research report: information animation applications in the capital markets". *Information Visualization, 1995. Proceedings.*, 19–25, 1995.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
18–03–2010	Master's Thesis	Aug 2008 — Mar 2010

4. TITLE AND SUBTITLE

Modeling Computer Communication Networks in a Realistic 3D Environment

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Rowell, Charles R. Jr., Capt, USAF

5d. PROJECT NUMBER

JON# ENG 10-336

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GCE/ENG/10-05

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

Air Force Office of Scientific Research
Attn: Robert J. Bonneau
875 N Randolph, Ste 325, Rm 3112
Arlington, VA 22203
703-696-9545
robert.bonneau@afosr.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFOSR/RSL

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approval for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Communication network simulations have typically been visualized in the past through 2D representations, but this is insufficient for battlefield network scenarios. Visual representations of battlefield networks greatly benefit from 3D visualization due to its ability to retain asset location. This research investigates the feasibility of modeling a typical battlefield communication network in a realistic 3D manner and discusses the effects of doing so. The result is an open source, 3D network visualization tool that can create highly intuitive connected battlefield scenes, enabling the user to quickly comprehend network state. It highlights mobile assets, packet movement, and node connectivity while allowing the viewer to interact with the scene.

15. SUBJECT TERMS

network visualization, 3D visualization, network simulation, visualization

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Lt Col Stuart Kurkowski, PhD
U	U	U	UU	98	19b. TELEPHONE NUMBER *(include area code)* (937) 255–3636, ext 7228; stuart.kurkowski@afit.edu