Autonomous 3D Visualization for Simulation Exercise Support

Rob van Son, Philip Kerbusch, Rick Appleton, Yntze Meijer
TNO Defence, Security and Safety
The Hague, The Netherlands
{rob.vanson,philip.kerbusch,rick.appleton,yntze.meijer}@tno.nl

ABSTRACT

Participants and spectators in simulated exercises often find it challenging to acquire good situation awareness and maintain it during the often hectic sequence of events that occur. The tools that are used to support them in this regard include 2D plan view displays and 3D stealth viewers. However, manually controlled 2D or 3D visualizations often do not provide the desired on-time overview and insight into the events that occur within the simulation. This is particularly the case when the environment is complex and dynamic, contains a large number of entities, or is geographically spread out.

ScreenPlay is an experimentation platform that acts as a virtual director that autonomously controls camera viewpoint positioning and movement in a 3D virtual environment. The system combines a rough behavior description and several view descriptions to produce the desired result, based on storytelling considerations, scenario context, and the events that occur within the simulation. By relying on assumptions of event patterns that are expected to occur in the simulated environment rather than needing to know exactly what will happen, ScreenPlay has a large degree of flexibility with regards to the occurrence and timing of events.

In this paper, we evaluate how autonomous 3D visualization can be used to support large-scale simulated exercises. This has initially been tested in small experiments in 2009. In 2010 it has been used to support visualization for analysis and after action reviews during the JPOW Joint Theatre Missile Defence exercise and in a number of simulated experiments for concept development and experimentation in the naval domain. For these cases, we will evaluate the process of configuring autonomous visualization routines for an exercise and we will discuss and compare the experiences and the results obtained by its application during both mission execution and debriefing.

ABOUT THE AUTHORS

Rob van Son joined TNO Defence, Security and Safety in 2005. He has been involved in various projects concerning the use of modelling and simulation technology for training, decision making, and experimentation. Currently, his research focuses on synthetic natural environment modelling and novel techniques for 3D information visualization and representation. He holds an MSc in Computer Science from Utrecht University with a specialization in computational geometry and virtual environment technologies.

Philip Kerbusch is a research scientist at TNO Defence, Security and Safety. Philip holds an MSc. degree cum laude in Artificial Intelligence from Maastricht University (2007). He has a strong background in computer science and knowledge engineering. His current work involves various applications of artificial intelligence in the defence and safety domain, specializing in behaviour modelling for serious games and simulation.

Rick Appleton is a software engineer specializing in 3D graphics and optimization. After obtaining his MSc degree in Aerospace Engineering from the Delft University of Technology specializing in Real-Time Simulation, he worked in the computer games industry for three years before joining Alten PTS as a

technical consultant. He is currently contracting at TNO Defence, Security and Safety to improve their proprietary 3D visualization software.

Yntze Meijer joined TNO Defence, Security and Safety in 2005. He is an operational analyst and has been involved in various projects concerning defence of ships against under and above water threats and power projection from the sea. Currently, he leads the naval surface fire support research program and the surface ship self defence project. He holds an MSc Aerospace Engineering from Delft University of Technology with a specialization in astrodynamics and satellite systems..

Autonomous 3D Visualization for Simulation Exercise Support

Rob van Son, Philip Kerbusch, Rick Appleton, Yntze Meijer
TNO Defence, Security and Safety
The Hague, The Netherlands
{rob.vanson,philip.kerbusch,rick.appleton,yntze.meijer}@tno.nl

INTRODUCTION

In the past decade, 3D visualization has become an accessible and commonly used tool for the visualization of training and experimentation sessions that are supported by simulation. In combination with 2D plan view displays, 3D has the potential to provide and maintain an increased level of situation awareness for operators and instructors. It is our belief, however, that this potential has not been fully realized; 3D is not as effective as it could be. This is particularly the case in environments that are complex and dynamic, contain a large number of entities, or are geographically spread out.

The use of intelligent and autonomous 3D visualization can, in the authors' opinion, help to make 3D visualization more effective and contribute to a higher and maintained level of situation awareness, resulting in improved training and experiment effectiveness. In the past few years, we have developed ScreenPlay, an experimentation framework for real-time autonomous 3D visualization control. The central theme in this research has been the concept that the desired visualization behaviour can be modelled a priori and, at the same time, maintain a degree of flexibility with regards to the events that occur in the simulation.

In this paper, we report our experiences on the use and value of autonomous visualization for supporting 3D simulated exercises and experiments. Two cases with varying goals and scope were used to evaluate and improve the use of autonomous visualization. For these cases, we describe the process of configuring the visualization and evaluate its value and impact.

In the next section, the issue of situation awareness in 3D simulation environments is explored in detail. Next, the ScreenPlay framework and its design are described. Following are two sections describing our experiences with two cases in which autonomous 3D visualization was employed, in which visualization goals, setup and configuration and the eventual visualization are described and evaluated. The final sections of the paper summarize our conclusion and provide recommendations and future research topics concerning autonomous visualization.

IMPROVING SITUATION AWARENESS IN 3D VIRTUAL ENVIRONMENTS

Situation awareness (SA) is defined by Endsley (2000) as "knowing what's going on around you". More specifically, it has been generally defined as "the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future". From this definition, three levels of (increasing) SA can be extracted: Perception, comprehension, and projection (Endsley, 1988).

Durbin et al. (1998) sum up the significant advantages that a 3D visualization potentially offers over a 2D plan view for acquiring and maintaining situation awareness, including clutter reduction, 3D information perception and terrain perception.

Although SA will always require a operator- or user-specific context to give it full meaning, it is our observation that 3D visualizations for simulated exercises in general often prove ineffective when it comes to providing and maintaining good SA, overview, and insight.

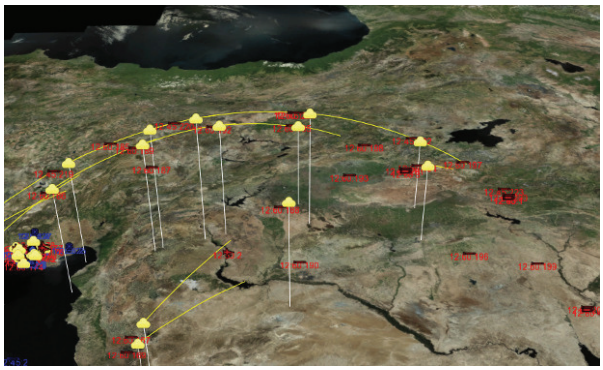


Figure 1. An example complex simulation environment (JPOW) with a large geographic scale and a large number of entities and interactions.

At the base of this ineffectiveness lies the complexity of the simulation environment. It will become increasingly difficult to visualize that what is truly important and interesting with regards to the simulation goals when geographic scale, terrain complexity, number of entities and the number of interactions between these entities increase.

To some extent, these problems can be resolved by employing (expert) manpower to manually control the 3D visualization and/or the addition of multiple (2D or 3D) viewer stations. This, however, does not solve certain information presentation problems. First of all, 3D camera control often is, especially for spectators, not smooth and may cause disorientation. Also, an expert user of a 3D viewer may not be a domain expert and therefore may not be aware of the intended narrative, i.e. exactly when, what and how events should be shown to spectators. Lastly, the visualization is typically not reproducable and will require exactly the same manpower and camera behaviour for the next simulation run.

A final issue that is becoming increasingly pertinent and is directly related to the problems involved with employing additional manpower and 3D viewer stations, is the unrelenting and increasing costs that are involved in setting up and operating 3D visualizations.

The issues mentioned above are as much a problem for real-time observation and monitoring as they are for visualizations for after action reviews. Even though during debriefing the user may have the capability to control simulation time more extensively and the operator possesses knowledge about exactly what has happened, time and capacity for handling the large flow of collected data is typically limited and emphasizes the need for an effective, relevant, on-time 3D visualization (Schavemaker-Piva et al., 2008).

Traditional 3D visualization for Simulation Exercises

A large number of 3D stealth viewer applications for visualization of high-complexity and large-scale virtual environments are currently available. Examples include AGI's Satellite Toolkit (Analytical Graphics, Inc., 2010) and TNO's own proprietary 3D viewer, JViewer.

Stealth viewpoint control is typically centered around the camera itself and relative to the terrain. Positioning a camera properly means manipulating its degrees of freedom to have it move into the desired position and orientation. Typically, view modes exist that allow the operator to attach the camera to a single, possibly moving entity and manipulate the camera relative to that entity.

Apart from camera navigation, 3D viewers provide additional information to acquire insight in a situation. This information is projected in the 3D environment and is presented in the form of text labels, tracklines, terrain overlays, visibility cones, et cetera.

It can be concluded that, despite several features that can help with respect to situation awareness, the resulting visualizations of typical 3D stealth viewers depend, to a very large extent, on the skill and experience level of their operators. One particular issue is the lack of camera control mechanisms that truly assist the spectator by, instead of focusing on the camera and its degrees of freedom, focusing on the events that need to be visualized.

Autonomous 3D visualization

In computer games, autonomous camera control mechanisms usually have the (limited) responsibility of keeping the player in view at all cost (Haigh-Hutchison, 2008). In these situations, cameras often operate in a confined, indoor environment and employ specialized routines to satisfy player visibility and occlusion avoidance. Camera control in computer games is typically based on a large amount of scripted, non-emergent behaviour as the course of action in a game is often linear or predictable.

Previous work on autonomous camera control focuses mainly on the use of so-called cinematographic idioms (He et al., 1996, Amerson & Kime, 2001). These are static structures that represent a predefined collection of camera shots that are activated in a certain order or become triggered based on events that occur in the environment. These idioms usually describe small, predictable situations such as a conversation or someone entering a room: Idiom-based systems are strongly based on assumptions of events that will consecutively occur in the virtual environment and are often linked to high-level narrative planners. Planning algorithms have the advantage of being strongly goal-driven, but allow little control to the designer and possibly unexpected and sub-optimal results.

Ting-Chieh et al. (2004) propose finite state machines as a solution for modelling story sequences and alternating between cinematographic idioms. Although finite state machines are easy to understand and to model and allow an almost infinite amount of control,

they tend to become hard to understand and difficult to maintain as their size increases.

From this short summary, it can be concluded that autonomous visualization of large-scale simulation environments is a topic that so far has not been explored extensively. Given its specific characteristics, specific solutions are required to deal with with them.

THE SCREENPLAY FRAMEWORK

ScreenPlay (Van Son et al., 2010) is a framework, initially developed in 2009, that was designed to control the position and orientation of a camera or viewpoint in a 3D virtual environment. It builds upon the results of TNO's research on the debriefing of fighter pilots (Schavemaker-Piva et al., 2008), in which autonomous camera control was a major element. It was shown that, by means of this feature, support could effectively be provided for on-time and relevant information presentation for the debriefing of fighter pilots.

ScreenPlay is based on a number of key principles that differ notably from traditional camera control in 3D virtual environments:

- Autonomous operation at runtime;
- A strong emphasis on the events that need to be visualized instead of on the camera and its movements;
- The incorporation of a priori knowledge about the development of a scenario or storyline and the events that occur during a simulation.

The ScreenPlay framework consists of two main components: the Camera Operator Agent and the Director Agent. These components are described in further detail in the following paragraphs.

Camera Operator Agent

The Camera Operator Agent is in control of the 3D virtual camera or viewpoint. Given 1) the view contents (i.e. the entities that are to be kept in view) and 2) a description of the view, it is responsible for calculating the desired or optimal camera position and orientation for each timeframe of the simulation.

View contents

The view contents consists of the entities, both moving and static, that are to be visualized and optional descriptions of the type of situation and their roles in this situation. These situations and roles can be used to define relationships and differences among the entities, and ultimately depict scene-specific elements that are used for camera positioning calculations, such as a point of interest and a line of action. An example situation would be a firefight, in which one entity is the attacker, another one a target, and perhaps a third one a projectile being fired.This situation is displayed in Figure 2.

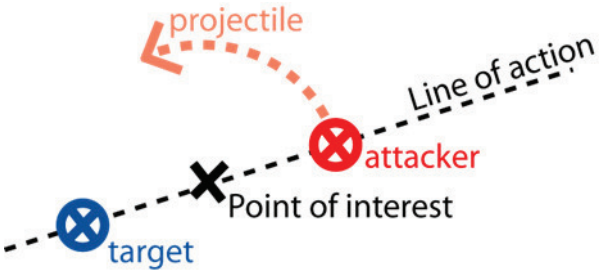


Figure 2. Example of view contents (attacker, target and projectile) and the data that is derived for camera positioning (point of interest, line of action).

View description

A view description consists of a set of one or more view descriptors. View descriptors are constraints, generic with regards to the view contents, that can be combined to describe a view and can be parameterized. Examples of view descriptors are "look at entities from a certain angle α" or "keep all specified entities in view". Together with the view contents, these are used as input for a constraint solver which determines the optimal camera position and orientation.

All view descriptions are referenced by ScreenPlay's other main component; the Director Agent.

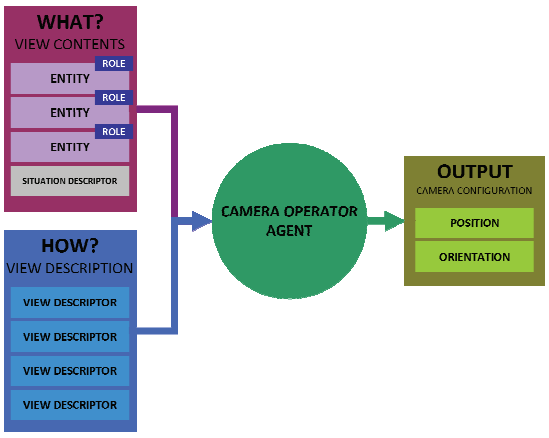


Figure 3. Schematic overview of the Camera Operator Agent. View contents and a view description are required as input to calculate the new camera configuration.

Director Agent

To describe The Director Agent effectively, it is best to compare it to a real-world director. However, in this case, we are not referring to the director of a movie in which the actions of all actors and the scenes can be predetermined and corrected without limit. Instead, a proper analogy would be that with the director of a live sports broadcast, in which the director has a global sense (based on training and experience) of how a game should develop and knows what is important at a certain time in the game. However, the director also has to deal with a certain degree of unpredictability with regards to the actions of the actors and the events that occur. Subsequently, he has to instruct his camera operator(s) to focus on certain events or switch between available footage.
Similarly, an autonomous camera control system for real-time visualization of simulations should possess a degree of knowledge about the events that could possibly occur within a particular exercise or experiment and the ability to respond to them in a flexible, to-the-point manner.

The Director Agent is a system responsible for a number of things:

- Keeping track of the narrative structure of a simulation;
- Responding to events that occur in the simulation;
- And, ultimately, instructing the Camera Operator Agent to switch to a certain view and/or certain view contents.

Director behavior specification
Central in the Director Agent system is its "brain" or behavior specification. A real-world director possesses knowledge about a story hierarchy (Brown, 2002), either explicitly by means of a storyboard or implicitly in his mind. He knows the sequences within a story, the scenes that make up a sequence, and, finally, the shots that together constitute a scene. This implies that the Director Agent should be aware of both the larger narrative of the simulation as well as local contextualisation.

For the Director Agent's behavior specification, Behavior Trees are used. Behavior Trees (Champandard, 2007) allow for behavior to be constructed in a modular, tree-like fashion with a small number of re-usable nodes (Millington & Funge, 2009). Behavior trees combine many of the advantages of both high-level, goal-driven planning algorithms and low-level, understandable finite state machines. Examples of these nodes are Sequence, Parallel, and Random Selector.

The leaf nodes of the Director Agent's behavior tree consist of the actions that it can perform. These include:

- Setting the Camera Operator's view contents;
- Setting the Camera Operator's view description;
- Waiting until a certain event occurs;
- Waiting a certain amount of time.

Figure 4 displays an example of a simple behavior tree in ScreenPlay. As can be depicted from the image, the behavior tree shows a large degree of similarity with respect to typical narrative structures used in movies and other visual media.

Expressing responsive and complex behavior
The availability of event data determines the user's flexibility to model complex, responsive behavior. Events are required to know about what occurs in the simulation environment and to respond to them in the desired manner. The framework has been designed to

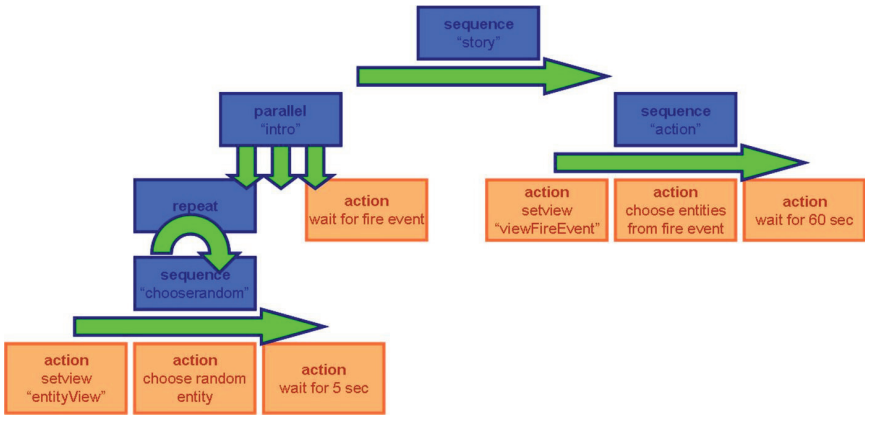


Figure 4. Behaviour tree example representing a storyline consisting of an intro and and action sequence (node names are between quotes). In the intro sequence, ScreenPlay keeps choosing random entities to view until a fire event occurs and will then view those entities in a particular way.

support default simulation events (e.g. entity position and orientation updates, fire events, detonation events) and is capable of generating of its own as the result of basis analysis (e.g. when two entities come within a specified distance of each other).

Additional features that were implemented were filters (e.g. to respond to a fire event only when it is generated by a hostile entity) and decorators, which compress otherwise complex behavior trees with reusable behavior into a single node. Besides that, a "blackboard" exists that allows the framework to temporarily store data. This is particularly useful when previously acquired information is needed, such as in the case where the camera needs to navigate from an old position to a new one.

The Director Agent's behavior is specified entirely in XML. Also, subtrees can be specified separately and referred to from any point within the behavior tree. This allows for a large degree of reusability and the gradual build-up of useful assets for future visualizations.

CASE STUDY: EXPERIMENTATION FOR NAVAL SURFACE FIRE SUPPORT

The Naval Surface Fire Support (NSFS) research program supports the future introduction of long range high precision fire support for forces on land delivered by naval vessels. This capability opens up new deployment and engagement options to the benefit of all branches of the armed forces.

The program investigates possible problems with its future deployment and recommends improvements in procedures and support to ensure an optimal use of the new capability at the moment it is introduced.
In the NSFS program, TNO combines domain expertise, ranging from NATO handbooks to operational commanders, with a synthetic environment in which the new capability itself is modelled and all involved personnel can interact according to detailed scenarios. This allows future users not only to test and develop foreseen procedures together but also to indicate need for support at different stages in the process when using the new capability together.

Simulation is necessary to give clear insight in the workings of, for example, a rocket assisted GPS/INU(Global Positioning System/Inertial Navigation Unit) guided steerable grenade during fire support. Without line of sight during the whole trajectory and shooting through a much larger volume of airspace, the naval officers have to enhance their procedures while fire support teams on land can increase their options (and hence their procedures). The pinpoint accuracy however remains limited with a certain circular error of probability upon impact and a ranged lethal radius depending upon targets.
The complexity of the scenarios made it crucial to have a good visualization of what happened during a run: a fleet tens or even hundreds of kilometers off the coast, projectiles flying for minutes with high angle trajectories and targets in a terrain where meters matter, and, last but not least, the actions of the operators.

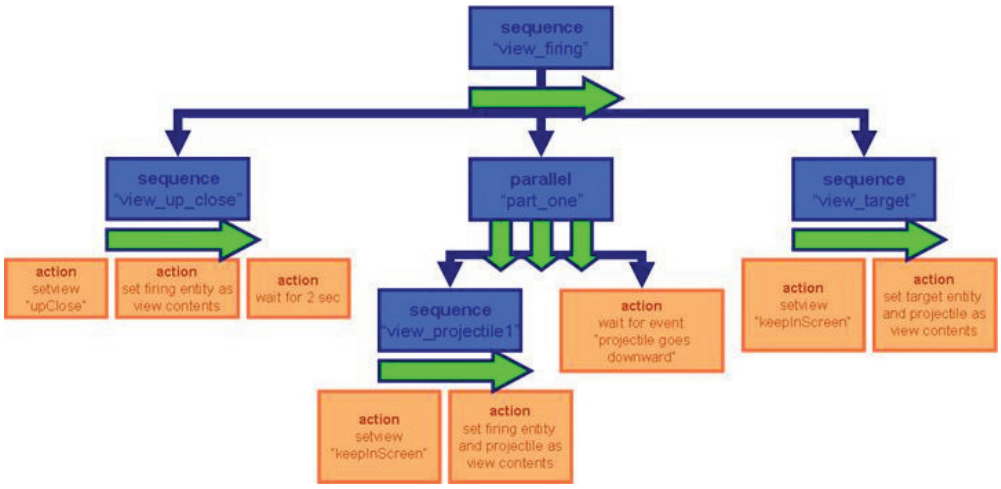


Figure 5. Behaviour tree example representing a projectile firing from a naval vessel to a target for NSFS. A sequence of views is set: First an up close view of the firing vessel, then a view that keeps the firing vessel and the projectile in screen while the projectile moves away, and, finally, a shot that uses the same view description to view the projectile and the target.

The naval forces need to understand the impact of the chosen trajectories and the spread of the projectiles. The forces on land need to understand the characteristics as well to make the best use of it, while the supporting arms coordination cell should know what it is managing.
A static 3D viewer would be no good. It cannot capture the immense flight paths and the impact area at the same time. Several static viewers, focused on different aspects, improve the situation but increase the number of computers needed considerably. However, even worse, it is not flexible: The interesting areas to focus on should be known in advance leaving no room for experimentation or deviations from the prescripted course (being either the scenario or the projectile). The human in the loop options to steer the camera gave poor results due to unsmooth camera steering and late responses to the virtual environment.

ScreenPlay Configuration for NSFS

During the NSFS experiment, two 3D viewer stations were in use: one with a manually controlled version of TNO's 3D viewing application JViewer, and one version of JViewer with the ScreenPlay framework controlling the camera.
ScreenPlay's behavior specification was constructed together with the experiment session leader and roughly describes the following narrative:

- Until the first projectile has been fired, cycle through views on the convoy of friendly naval vessels, hostile targets and the mission area as a whole;
- When the first projectile has been fired, follow the projectile with a sequence of shots that begin close to the firing vessel, zoom out to show the projectile's trajectory, then focus on the target and the projectile as it homes in;
- After impact, observe target for some time, then zoom out to a global perspective and maintain this until the new projectile is fired.

The behavior tree for this narrative is visualized as a simplified version in Figure 5.

Results

Figures 6a to 6f give an impression of the images generated using autonomous visualization techniques during the simulation run.

As this experiment was one of the first events in which autonomous 3D visualization was employed, a reasonable amount of time was used to determine exactly which views were required and how they would fit into the narrative. As the behavior specification is completely reusable, it is expected that this effort will repay itself in the near future.

The viewer setup presented an opportunity to compare a manual 3D viewing setup with one that is augmented with a framework capable of autonomous 3D visualization.

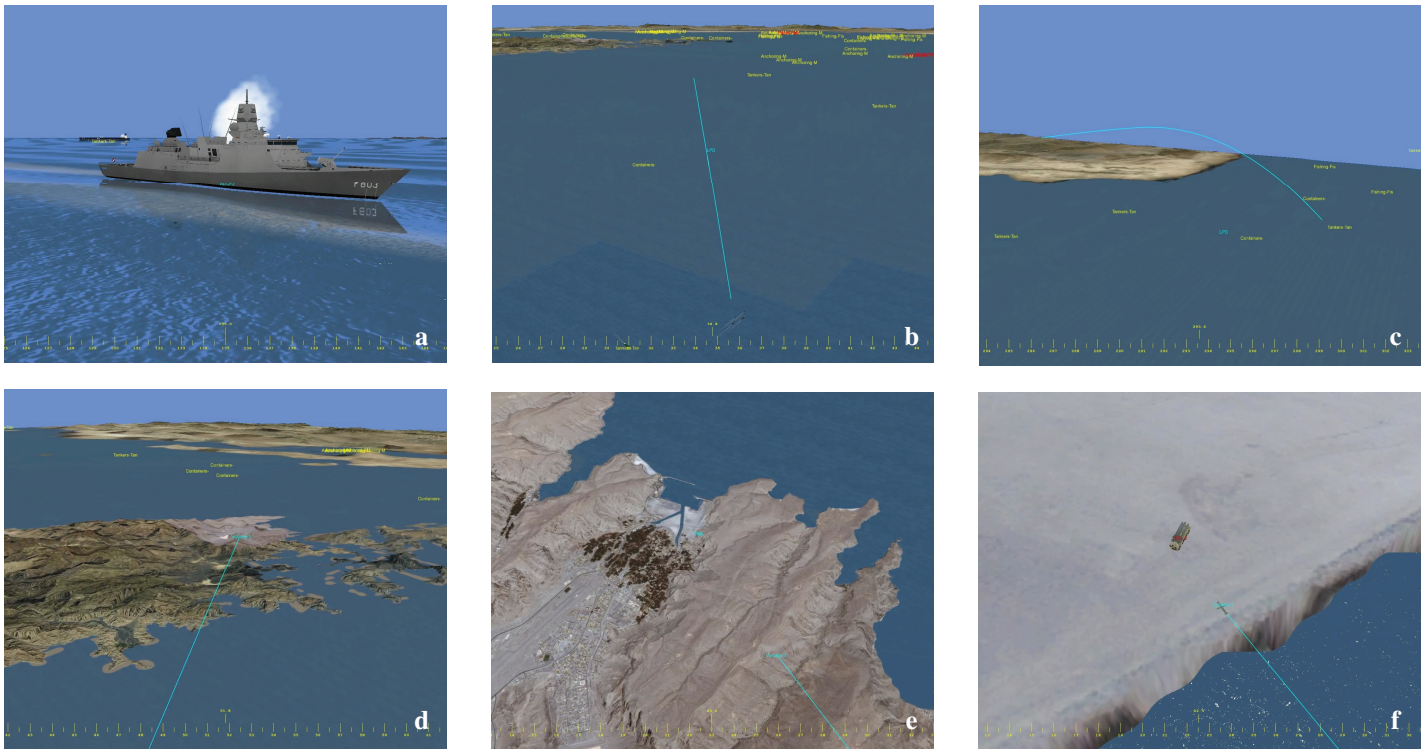


Figure 6a to 6f. Image sequence showing various dynamic shots consecutively produced by ScreenPlay during the firing of a projectile. The projectile is visualized by the light blue trajectory line. Figure 6a shows the launch from the naval vessel, Figures 6b and 6c show different views with the projectile and the vessel in screen, Figure 6d shows the projectile at the climax of its trajectory, Figures 6e and 6f show the projectile and the target as they come closer, until impact.

The manual setup had one advantage over the autonomous setup: if an important event was about to happen at a certain time (e.g. a launch) the camera could be steered to the expected event location beforehand to acquire a good view of it. A dynamic 3D viewer gave the best results however. It provided direct insight in what was actually happening due to its smooth transitions between areas of interest, which made it understandable for the public. It aided greatly in the after action replay with all participants and, last but not least; it saved a crew member during the experiment.

CASE STUDY: TRAINING IN A JOINT AIR MISSILE DEFENCE EXERCISE

The Exercise Joint Project Optic Windmill (JPOW) is a world leading Integrated Air Missile Defence (IAMD) exercise. JPOW focuses on both collective training and experimentation with novel air and missile defence concepts, tactics, procedures and future capabilities in a collective, multinational live, virtual and constructive environment (Jacobs et al., 2009). In 2010, JPOW's eleventh iteration, JPOW XI, took place.

The JPOW XI simulation environment is extraordinary in scale and scope. The exercise environment is set in a fictional version of our world and spans an area with a size of approximately 4000 by 2500 kilometers. Also, a large number of operators, ranging from tens to hundreds, participate in the exercise.

This environment calls for solutions that assist operators and spectators in acquiring and maintaining overview and insight into an extremely complex and dynamic environment.

For 3D visualization during the exercises, one standalone setup using autonomous visualization was deployed that was used to provide both functional and visually appealing real-time 3D visualizations of the exercise area. Because of the need for standalone operation, these stations were given a user interface that allowed interaction by any user (operator or spectator) by means of a touch screen. This was a particular challenge since this implied that the expert was moved completely out-of-the-loop. Users were given a limited number of view modes, two of which were supported by the autonomous visualization framework, each with a different kind of behavior.

These view modes are:

- Free Look – A basic view mode that allows user to manually position the viewpoint in the 3D environment;
- Group View – A view mode that automatically cycles between groups of entities that have been defined as pertinent for a particular exercise day;
- Intercept View – A view mode that waits for intercept actions against hostile projectiles and then attempts to visualize this intercept from launch to interception.

Besides these view modes, users were able to select one of three Battle Management Areas for a global overview of that environment. These views were also generated by the ScreenPlay framework. An impression of the user interface is shown in Figure 7.

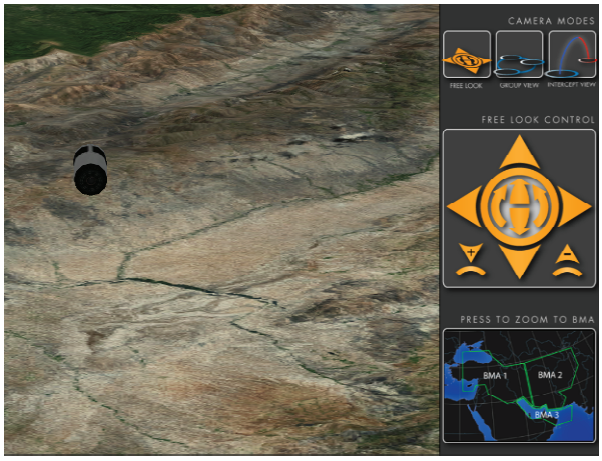


Figure 7. Close-up view of the ScreenPlay control GUI, showing controls to swith between ScreenPlay view modes, control the Free Look view, and zoom to a selected Battle Management Area.

Three additional autonomous 3D visualization applications were used to support the conceptual JPOW Joint Analysis Team in their needs to provide quick and effective debriefings at the end of each exercise day. The viewer applications functioned as a means to observe the exercise runs and mark interesting events and as visualization tool for debriefing, a solution was required that was generic and needed little to no modification to display new analysis results.
One of the particular issues that they focused on was the occurrence of so-called "leakers"; hostile projectiles that somehow were not intercepted. One of the exercise goals was to improve debriefing. As there was only limited time available for debriefing, a solution was required that was generic and needed little to no modification to display new analysis results.
For this particular case, autonomous visualization was controlled from an analysis module. By clicking on a particular event generated by the module, e.g. the occurrence a so-called "leaker" missile, an autonomous 3D visualization was instantiated with a generic behavior tree with event-specific view contents.

ScreenPlay Configuration for JPOW

In this section, we describe two visualization behavior specifications that were used for JPOW. These have both been used to describe the Intercept View narrative, but each have a different focus: One is meant for functional visualization, while the other one is meant for a more attractive (and perhaps less insightful) visualization. The functional behavior specification roughly describes the following narrative:

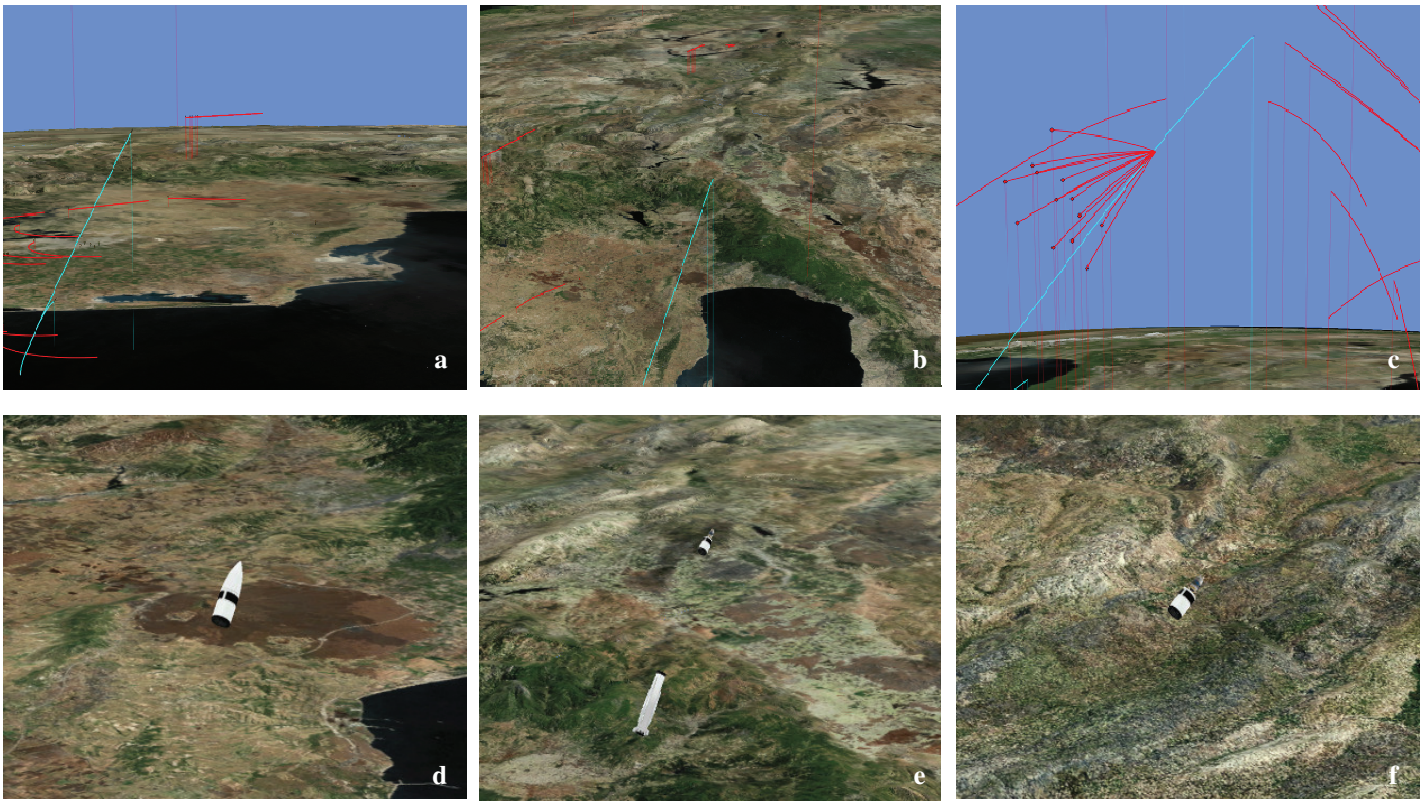
- Provide a global overview of the exercise area;
- If an intercept is initiated (i.e. the occurrence of a fire event with specific filters with regards to entity alignment and type), show both the launching unit and the intercept projectile;
- After a few seconds, focus solely on projectile;
- When the actual intercept occurs, keep point of impact in screen to view the effects of the interception;
- After a few seconds, go back to global overview of the exercise area and wait for the next intercept.

The second behavior specification follows similar guidelines, but employs different views. On launch, it focuses immediately on the intercept projectile from a close-up view and will follow it until impact. After impact has occurred, it will zoom out to view the effects of the interception.

Results

Figures 8a to 8f give a visual impression of the results that have been achieved with the Intercept View mode, both in functional and "appealing" mode.

The results achieved in JPOW show that it is possible to have a generic ScreenPlay setup that can be used throughout an exercise that lasts for multiple days and focuses on different aspects of Integrated Air Missile Defence. Minor configuration issues involving the improvement of view descriptions and the specification of a new set of entities of interest for the next run were resolved inbetween exercise runs.



Figures 8a to 8f. The top row of images gives an impression of the images generated by the functional version of the Intercept View. The bottom row of images gives an impression of various flight stages of the same intercept projectile in a visually different style.

Autonomous visualization has also shown to be a versatile solution for various 3D visualization problems. It has been employed for different applications (autonomous real-time visualization and debriefing) and has shown to be capable of providing both functional (with respect to SA) and visually appealing images.

Compared to the NSFS case study, the autonomous visualization could be tailored less specifically to a particular scenario. This resulted in issues with regards to camera motion: Transitions of the camera between distant positions occasionally showed to be problematic and caused disorientation. Often, it was not clear to where the camera was moving, and along which path. Therefore, spectators needed additional time to orient themselves in the environment of the new view.

It should also be noted that a visually appealing visualization may, for a large part, depend on the quality of the visual content: The high-fidelity missile model shown in Figures 8d to 8f greatly improves the visual experience.

RESULTS AND CONCLUSIONS

In this paper, we have described two case studies in which ScreenPlay, a framework for autonomous 3D visualization, has been implemented and evaluated. These two case studies differ with regards to the goals that were to be achieved with the simulation environment, and even more in terms of domain, scope, and size.
The application of autonomous 3D visualization in two very different cases underpin its potential, versatility and general applicability, both in terms of supporting various scenarios and added value for real-time visualization and debriefing. Although the ScreenPlay framework has currently only been connected to a proprietary 3D viewer (JViewer), we expect it can be connected to any other 3D viewing application with little effort.

Configuration

At this point, someone with ScreenPlay development and configuration experience is still required to specify the views and behavior specifications required for the

framework to function. This requires close interaction with the person responsible for scenario development. Also, configuration at this time still takes a notable amount of time and requires testing and tweaking beforehand to achieve an optimal visualization.
However, comparing the experiences from the Naval Surface Fire Support case with the Joint Missile Defence case, we noted that, as experience and the amount of reusable assets (i.e. ready-to-use behavior tree and view specifications) increase, configuring visualization routines for a particular scenario becomes easier and requires less effort and testing. This is also notable in the level of abstraction in configuration: whereas in the previous case a large focus was on low-level technical configuration of single views, in the latter case this focus has shifted to a higher level with more emphasis on the story and the image that was to be conveyed to spectators.

Situation awareness

Although so far there have been no formal investigations on the value of autonomous 3D visualization with regards to situation awareness, we can draw a number of conclusions. First of all, we can conclude that such a visualization typically performs well when it comes to rendering static images or images from a relatively stable position with regards to the entities that are to be displayed.
One major point of improvement is camera motion during transitions from one view to another: When the camera needs to move between two geographically distant locations, the (simple) motion planning tends to confuse spectators and may temporarily put them in a situation where they don't know where they are or what they are looking at.
Another related feature that is currently lacking is the ability to show effects, such as a fade to black for a transition from one view to the other. Currently, a flying motion or a hard "cut" are the only ways to perform a transition between views.

Summary

Our experiences so far lead us to conclude that the use of autonomous 3D visualization techniques can be beneficial with regards to the general goal of acquiring and maintaining situation awareness and indeed can make 3D visualization more effective. By this we mean not just effective in terms of visual quality, but also in terms of workload and manpower required to make things work. More so, as ready-to-use assets and experience with the increase, the quality of the visualization is expected to increase while the configuration workload for a single scenario will decrease.
It should be noted, however, that real-time autonomous visualization has not yet reached the movie-like quality that spectators often appear to expect. This remains an issue, even though the spectators' frame of reference might not be an entirely fair one for comparison.

In the next section, we describe the various research topics, that are adressed currently or will be investigated in the future, that focus on improving the ScreenPlay framework towards increasing the quality and effectiveness of the produced visualizations.

BRINGING AUTONOMOUS 3D VISUALIZATION TO THE NEXT LEVEL

The results that have been achieved so far give reason to believe that autonomous 3D visualization, if used properly, can lead to a more effective application of 3D visualization.

Improving ease of configuration

A downside to the growing list of features and assets is the increasing complexity of the system. ScreenPlay currently requires an expert user with a large amount of in-depth knowledge to do the configuration. Ideally, control of the visualization would be in the hands of a scenario developer, experiement leader, or instructor.
To handle this issue now and in the future, any tool supporting autonomous visualization will need to continue to find a balance between providing high-detail behavior specifications versus abstract, directly usable, domain-specific building blocks. Also, a graphical editor interface can assist greatly in making the behavior modelling prcess accessible.

Improving situation awareness

We believe that improving the visual quality of the produced images will be a key factor for the success and acceptance of this novel technique.
First of all, camera motion planning is considered to be an issue that limits the current quality of the produced results. Situation-aware camera navigation is a research topic currently being adressed at TNO and is expected to deliver new results by the end of 2010.
Secondly, research emphasis should also be on non-camera-related visual improvements. The current behavior specification provides the Director Agent with a narrative context that could potentially be used for much more than just camera control. Examples include context-specific visual augmentation, such as

pinpointing what event and which entities the camera is looking at, and effects, such as a fade to black during a transition.

A third research topic is the use of other cues to the current position and movement in the virtual environment. Possible examples of these cues are overview map displays and textual augmentation that denote the current location.

New applications

So far, autonomous visualization has mainly been employed for simulation exercises that took place in a large theater, with interactions between entities over large distances.

An infantry training environment (e.g. Virtual Battlespace System 2 (Bohemia Interactive Simulations, 2010)) will provide new challenges that must be tackled for autonomous visualization to be of benefit. This is particularly the case for camera positioning and navigation in small urban spaces. This is also a topic currently under investigation.

REFERENCES

Amerson, D. & Kime, S. (2001). Real-Time Cinematic Camera Control for Interactive Narratives. *Proceedings of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment*.

Brown, B. (2002). Cinematography: Image Making for Cinematographers, Directors and Videographers. Oxford: Focal.

Champandard, A.J. (2007). Understanding Behavior Trees. Retrieved June 25, 2010, from http://aigamedev.com/open/articles/bt-overview/

Durbin, J., Swan, J.E., Colbert, B., Crowe, J., King, R., King, T., Scannell, C., Wartell, Z., & Welsh, T. (1998). Battlefield Visualization on the Responsive Workbench. *Proceedings of the IEEE Conference on Visualization '98*, pp. 463-466.

Endsley, M. R. (1988). Design and Evaluation for Situation Awareness Enhancement. *Proceedings of The Human Factors Society 32nd Annual Meeting*, pp. 97-101.

Endsley, M. R. (2000). Theoretical Underpinnings of Situation Awareness: A Critical Review. *Situation Awareness Analysis and Measurement.* Mahwah, NJ: LEA.

Haigh-Hutchison, M. (2008). Real-Time Cameras. A Guide for Game Designers and Developers. Morgan Kaufmann.

He, L., Cohen, M.F., & Salesin, D.H. (1996). The Virtual Cinematographer: A Paradigm for Real-time Camera Control and Directing. *Proceedings of SIGGRAPH 1996*, pp. 217-224.

Jacobs, L.R.M.A., Cornelisse, E., & Schavemaker-Piva, O. (2006). Innovative Debrief Solutions for Mission Training and Simulation: Making Fighter Pilot Training More Effective. *Proceedings of I/ITSEC 2006*, pp. 215-224.

Jacobs, L.R.M.A., Van de Wiel, R., Bosch, J. & Olthoff, R. (2009). Supporting Collective Training & Thinking in Joint Project Optic Windmill. *Proceedings of I/ITSEC 2009*, pp. 1698-1710.

Millington, I. & Funge, J. (2009). Behavior Trees. *Artifical Intelligence for Games, Second Edition*, pp. 334-371. Morgan Kaufmann.

Schavemaker-Piva, O., Van Son, R., Jacobs, L.R.M.A., van Maarseveen, R.A. & Stallman, A. (2008). Innovative Debriefing Solutions to Enhance Fighter Pilot Training. *Proceedings of I/ITSEC 2008*, pp. 1335-1344.

Ting-Chieh, L., Zen-Chung, S. & Yu-Ting, T. (2004). Cinematic Camera Control in 3D Computer Games. *Proceedings of WSCG 2004*, pp. xx-xx.

Van Son, R., Kerbusch, P.J.M. & Appleton, R. (2010). Cinematographic Camera Control Using Behavior Trees. *Proceedings of International Workshop on Motion in Games 2010*.

Analytical Graphics, Inc. (2010). AGI STK. Retrieved June 25, 2010, from http://www.stk.com

Bohemia Interactive Simulations (2010). VBS2 VTK. Retrieved June 25, 2010, from http://www.bisimulations.com/products/vbs2