Collaborative 3D Visualizations of Geo-Spatial Information for Command and Control

Lars Winkler Pettersson*
Department of Information Technology
Uppsala University

Ulrik Spak†
Swedish National Defence College

Stefan Seipel‡
Department of Information Technology
Uppsala University
Department of Mathematics and Computer Science
University of Gävle

Abstract

We present a prototype command and control system that is based on view-dependent co-located visualizations of geographically related data. It runs on a 3D display environment, in which several users can interact with view consistent visualizations of information. The display system projects four independent stereoscopic image pairs at full resolution upon a custom designed optical screen. It uses head tracking for up to four individual observers to generate distortion free imagery that is rendered on a PC based rendering cluster. We describe the technical platform and system configuration and introduce our unified software architecture that allows integrating multiple rendering processes with head tracking for multiple viewers. We then present results of our current visualization application in the field of military command and control. The command and control system renders view consistent geographical information in a stereoscopic 3D view whereby command and control symbols are presented in a viewpoint adapted way. We summarize our experiences with this new environment and discuss technical soundness and performance.

CR Categories: I.3.2 [Computer Graphics]: Graphics Systems—Distributed/network graphics

Keywords: Distributed Rendering, Networked Virtual Environment, Display Environments, Stereoscopic Projection, GIS

1 Introduction

The demand for visualization in command and control systems follows an increased focus on Network-Centric Warfare [Sundin and Friman 2000]. Information from knowing exact positions and status of individual military objects such as troops, tanks, airplanes, etc. will only give an advantage if the information can be visualized in an efficient and useful way in an environment suitable for collaboration. In our work towards increasing the amount of directly available information to a user, we have chosen to focus on stereoscopic visualization. The large amount of military information related to spatial geographic data makes command and control systems suitable for stereoscopic visualization. We suggest that to achieve efficient stereoscopic visualization in a collaborative environment each user's perspective of the display must be view-dependent with the visualized model co-located in the same presentation volume. The first characteristic, view-dependent perception is important for enhancing the visualization by presenting each user in the display environment with textual tags and symbols oriented towards their gaze. The second characteristic, the visualization of a co-located model, ensure that all the users in the environment perceives the model as one and can interact with it. To describe the visualization technique based on these characteristics we use the term view-dependent co-located visualization.

Computer generated stereoscopic visualization has a vast history. Only a decade after the first computers were brought into existence, volumetric displays were invented [Hirsch 1961; Ketchpel 1964] and a few years after the head mounted display (HMD) [Sutherland 1968]. The stereoscopic visualization techniques developed since then are to different degrees suitable for view-dependent co-located visualization.

Autostereoscopic displays do not require the viewer to use a head worn filter, since the display itself generates a stereoscopic visualization. The two most common classes of autostereoscopic displays for displaying general three-dimensional information are volumetric displays and parallax displays [Halle 1997]. Volumetric displays can have an unlimited number of viewers due to their nature of sweeping out a volume in space, but they do not allow for view-dependent visualization since each point in visualization space can be seen by several users. Parallax displays such as computer generated holograms and lenticular array displays makes view-dependent co-located visualization theoretically possible but practically hard to achieve. In the case with computer generated holograms the data rate needed to produce a stereoscopic visualization is very high and still not practically viable. Lenticular array displays uses an optical layer that for each additional viewer reduces the resolution of the image. For more than two face to face collaborating users this technology will not suffice. HMDs used for displaying stereoscopic visualizations can either represent the surrounding environment through a virtual setup, a video see-through setup or an optical see-through setup [Azuma 1997]. Command and control work is highly dependent on face to face collaboration. HMDs representing the surrounding environment by a virtual or video captured stream will hinder the detection of subtle facial expressions used in face to face collaboration. Optical see-through HMDs could be an efficient alternative for collaborative work, but the technique is sensitive to incorrect head tracking with the requirement of registering six degrees of freedom (6DOF) at below 10 ms latency [Azuma 1997]. Compared to HMDs, stereoscopic screen visualization with

*e-mail:lwp@it.uu.se
†e-mail:ulrik.spak@fhs.mil.se
‡e-mail:ssl@hig.se

shutter glasses or passive polarizing glasses can be achieved efficiently by means of three degrees of freedom tracking of the viewpoint. It is therefore less affected by angular tracking errors and net latency in the registration equipment. Stereoscopic presentation techniques based on active shutters generate frame rate dependent flickering when two or more users share a presentation area in a temporal multiplexing scheme [Agrawala et al. 1997]. In order to avoid flickering when shutter glasses are used for more than one independent observer, the display can be spatially divided into separate viewing zones for each user [Kitamura et al. 2001]. This approach works at the cost of significantly reduced effective resolution and spatial viewing area of the display. A recent paper [Hedley et al. 2002] describes a technique for presentation of geographic information in a multiple-viewer collaborative setting. In their work, view-dependent co-located visualization is accomplished by using opaque head mounted displays for each independent observer. The geographic information is presented in the environment using a video see-through augmented reality (AR) system. The system correlates geometries in a video based stream to fiducials where one layer is used to present the geographic information and others as lenses looking onto the geographic information. Another approach to 3D visualization of geospatial information is VGIS [Lindstrom et al. 1997]. Their work focuses on data handling and level of detail algorithms and not on a multiple viewer collaborative setting. Data is preprocessed and presented using a bottom up technique which acquire and presents data in three stages: Preprocessing offline, Intermediate online processing and Real-time online processing. Their system interacts with simulations using Distributed Interactive Simulations (DIS) and can in the preprocess stage access data from Arc/INFO GIS servers.

2 The Multiple Viewer Display Environment

In our research we use a novel display environment that is composed of several hardware and software components. At the very bottom we find a display which presents simultaneously the content of eight independent image sources. These images are presented within the same physical area on-screen and can therefore be perceived in the same locus by different viewers. The eight co-located images can be used to either provide four independent viewers with individual stereoscopic imagery or to serve eight independent observers with monocular views.

2.1 Display and graphics hardware

The physical display system is essentially an integrated retro-projection display system, in which the projection screen is oriented horizontally. This approach provides a viewing metaphor virtual world that is perceived as a bird's eye view. Similar viewing configurations have been presented earlier for instance in medical application scenarios. What is novel with the display environment presented here is the fact that eight independent digital image projectors are used to rear-project images from eight independent image sources upon the same physical screen area. Each two projectors are projecting from one of the four main directions upon the screen. The square screen has a size of 0.8 by 0.8 meters. Projection images are horizontally adjusted such that the digital image centers align with the centre of the square screen. The visible pixel resolution of each projection image on-screen is 768 by 768 pixels. All eight projection-images are superimposed upon the same physical screen area, but pairs of two are rotated 90 degrees or 180 degrees, respectively in relation to one another. The separation of

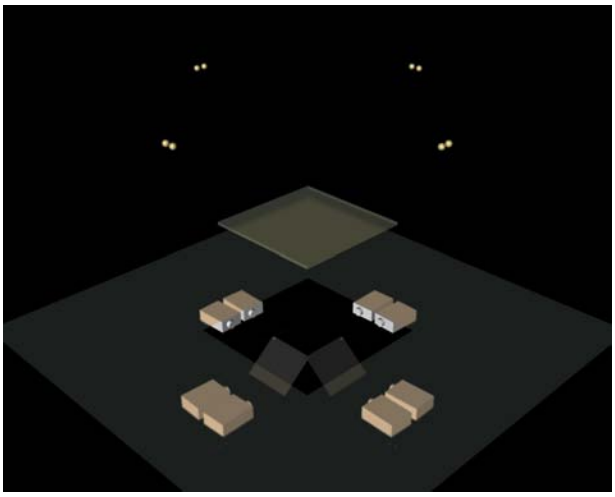


Figure 1: Configuration of the multiple viewer co-located retro projection display.

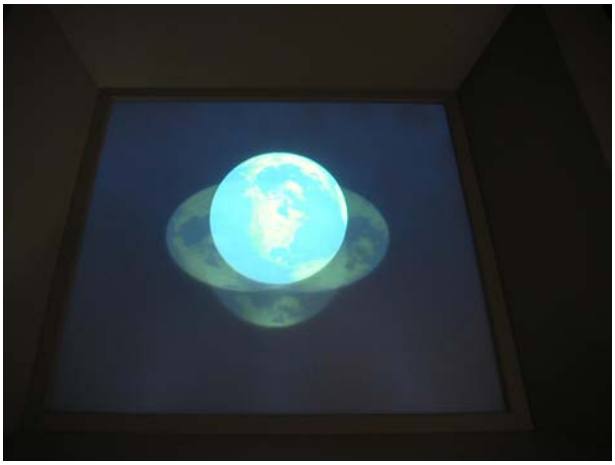


Figure 2: View upon the co-located display as seen by one eye of the observer. The picture shows the viewpoint corrected monocular view as well as (subdued) the pictures aimed for the remaining other viewers.

stereo image pairs is accomplished by using polarizing filters in front of the image projectors rather than using an active shutter system with temporal multiplexing. Figure 1 shows schematically the configuration of the display system and the viewer's eye positions in relation to the horizontal screen. One component of the display that is not further described in this place is a special optical design for the retro-projection screen, which allows the projection images to penetrate the screen primarily along the direction of the optical axis without loosing polarization. In consequence, a viewer being located at one side of the screen perceives only two of the eight projection images and, when using passive stereo glasses, perceives only one of the two images on either eye. Figure 2 shows the viewpoint adapted projection image for an observer looking from one side of the horizontal display. In this picture, also the view of the remaining three observers upon the same object is evident. The picture is taken without polarizing filters, and for each observer only a monocular view has been rendered. This display system is driven by a small cluster of four commercial off-the-shelf computers that are interconnected with a Giga-Ethernet local area network. Each of

the four computers is equipped with a NVidia QuadroFX graphics cards and renders two pipes. Hence, for a four-person configuration, each computer renders the stereo image pair for one observer. For the purpose of head position tracking, a magnetic tracking system is used (Ascension Flock of Birds) that is connected to a separate computer.

2.2 A model for distributed rendering

As is evident from the technical description of the display environment, the rendering process must be delegated upon different rendering nodes, which requires some means of synchronization of both the scene updates and the viewer dependent rendering parameters (e.g. head position and viewport orientation). Distributed rendering and networked VR systems have been described earlier and in various places in literature. These approaches can be divided into two classes: The first class are systems that support rendering into tiled displays (e.g. for very large projection walls). The purpose of these kind of rendering architectures is to distribute graphic states and primitives at a very low level, whereby typically the nodes in a rendering cluster render a sub-portion of one view upon the same scene. A well-known system is the Chromium architecture [Humphreys et al. 2002]. Since these systems work at the very low end of the graphics rendering pipeline, they cannot be used in applications that require node specific 3D scene content. The second class of networked based rendering systems tackles the problem at the high level graphical API. Commonly found in scene-graph programming tools, they provide fully transparent propagation of changes in the scene objects and scene graph. Early typical examples are the DIVE system [Carlsson and Hagsand 1993] or more recently Net Juggler [Allard et al. 2002]. The comfort of having a transparently shared and consistent scene database distributed over a network is achieved at the cost of performance. Still, in most networked VR environments, where the rendering results of various nodes are viewed at different physical locations, certain delays in scene updates can be tolerated. However, for physically co-located view-ports as in tiled displays or as in our case of superimposed displays these distribution models are not feasible, because delay becomes a strongly distracting visual artifact. Typical for our application scenario is an unusual blend of shared data and node-specific states that affect rendering in the local node. The combination of all following requirements renders our application different from most other networked or cluster based rendering applications:

- Large and complex parts of the scenario (e.g. terrain) are static and do not require network propagation
- Only relative few shared objects in the scene do change/move and need to be distributed among rendering nodes
- Some parts of the scene (observer dependent symbol orientation) are node specific content and pertain to the local node only
- View-port orientation and off-axis projection parameters are node/observer specific and change continuously

Due to these particular viewing conditions, none of the existing networked rendering architectures show to be really efficient solutions for these purposes. Systems like Chromium do not cope well with node specific scene content since they forward the graphical primitives from one host application to several rendering nodes. Here the nodes render view-port selectively rather than scene specific content. The classical network based VR systems on the other hand introduce too much overhead considering, in our application, the very few required scene updates that must be maintained almost instantly at all nodes. In order to solve this problem independently from the underlying scene-graph toolkit used for application development, we developed a very slim distribution model for fast exchange of data that must be shared among rendering nodes and among tracking devices connected to the environment. At the core of this distribution model we implemented a virtual shared memory model that allows for applications to allocate and subscribe to arbitrary memory partitions. The shared data repository resides on a server and uses either TCP/IP or UDP services for propagation of state changes between clients and the shared repository. Applications can enforce strict data consistency, when utilizing the built-in locking mechanisms of our shared memory API. However, strict consistency is not always needed and can be sacrificed in favor of improved overall network performance. One typical such example is frequent state changes of animated 3D objects as a result of direct user manipulation. In the practical situation it is more favorable to have a low latency motion of that kind of objects that is visible simultaneously for all viewers even though some of the states along the motion might be lost. Based on this virtual shared memory, our distribution model implements so-called data pools, where information is shared among the clients, that is relevant for distributed rendering and eventually for communication with other applications. Our distribution model encapsulates mainly three types of objects into the shared data pool. They are:

Projectors A projector entity contains configuration parameters for a specific viewing frustum to be rendered by one or several rendering nodes in the cluster. This comprises projection frustum parameters in physical real world coordinates as well rendering pipe parameters (viewports) and references to sensors, to which the viewing position is locked.

Sensors The sensor entity is an abstract object that pertains to the values received by some specific input device. It keeps a description of the semantics of the data delivered by the input device as well as the actual data values measured with the device. A sensor can for instance be a representation of any physical tracking device, but it could also be a logical device that integrates data from different physical input devices

Messages Messages entities are used to communicate messages between processes. Message passing is accomplished by subscribing to message entities. Accessing a message entity broadcast the content of that message entity to all subscribing clients.

More technical details on this distribution model can be found in [Lindkvist 2002]. An experimental performance study of this distribution model has been presented by [Seipel and Ahrenberg 2002]. In the following section we describe, how this distribution model is used in order to integrate multiple rendering processes into one visualization application. Figure 3 illustrates the relations between the different processes involved into the visualization system.

Four *rendering processes* that run concurrently on either network node generate the visual output in the horizontal display. These rendering processes are identical processes and therefore provide the same functionality in regard to scene behavior and user interaction. At startup these render processes initialize their local scene graphs based on a common shared scene database. At this stage, the only difference among the four rendering applications is the parameterization of the off-axis projections, their viewports on-screen and the associated sensor entity for head tracking. This process specific configuration is loaded from a local configuration file that keeps a reference to the actual sensor and projector entities in the shared repository containing the relevant data. The rendering applications do not maintain the actual data by themselves. Instead, sensor and projector data is read only by the rendering processes to update the display appropriately. The actual manipulation of the current

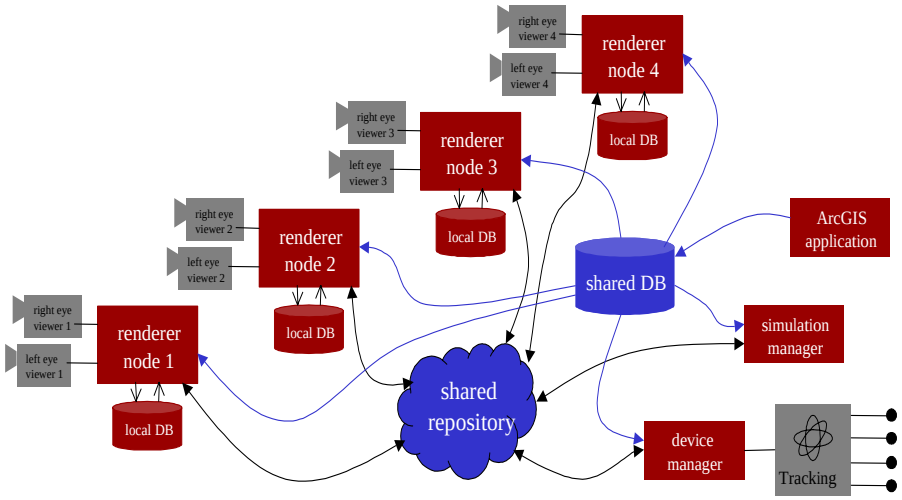


Figure 3: The drawing shows schematically how input and output device (gray), processes (red) and shared resources are communicating to flexibly synchronize the distributed rendering process in the co-located visualization environment.

configuration data for different projectors as well as the update of current tracking data in the shared repository is performed by separate processes. Dedicated *device managers* are running as independent processes in the network and continuously translate tracking values from devices into sensor values shared in the repository. In this process, hardware specific device parameters are abstracted to logical sensor devices that have certain agreed upon properties (e.g. representing a 3D position, or a set of three vectors). For the proper operation of the entire visualization environment, it does not matter what number and actual type of input devices are used to capture the values of a sensor entity. In fact, a device manager process could read measurements from different tracking systems and combine them into a logical sensor entity with higher accuracy. Or it could combine several positional device values into on position/orientation sensor. Also, switching or replacing input device hardware during operation of the visualization is only a matter of starting up a new physical device with its corresponding device manager process. The principle coordination of all visualizations shown in this environment is carried out by a server process, which in this context is called a *simulation manager*. The simulation manager is the administrator of the shared repository and main negotiator between the nodes. It continuously updates almost all entities in the shared repository and it routes messages between the different processes. One of the main tasks of the simulation manager is to update projector entities as a consequence of updated sensor values due to head motion of individual users. Since projector entities are transparently shared through the repository, all applications that have subscribed to a currently updated projector entity will autonomously update their output on screen. In that sense updates of the viewpoints the individual rendering clients can be controlled remotely and simultaneously by other processes. Changes of projector entities (hence client viewing parameters) could also be triggered by other external events. One relevant example is the change of the geographic context and cartographic layers within the 3D visualization. Using commercially available *geographical information systems* (GIS), an operator can chose the desired geographic region and map content and export it to a predefined server partition. The simulation manager monitors continuously all recent updates in that partition and broadcasts messages to the rendering applications to initiate reloading or updating of their local scene data (maps). This is a rather simple and straightforward method to control the flow of geographic visualizations in the environment described here. It proves efficient because it does not require reprogramming of external applications and it uses established GIS tools to retrieve the geo-spatial data. Other tasks performed by the simulation manager are controlling of workflow within the 3D visualization by means of other user interfaces, or to interface with other external programs, which in our case, simulate military scenarios. In all cases of simulations, the simulation manager acts as a computing process and event dispatcher only. The actual visual simulation and animation of graphical objects is handled identically by the rendering clients, which administrate their local scene graphs. Therefore, animated graphical processes in the distributed visualization follow a dead-reckoning approach [Singhal and Zyda 1999], whereby animation of objects is performed autonomously at the local clients without synchronization between the animation steps.

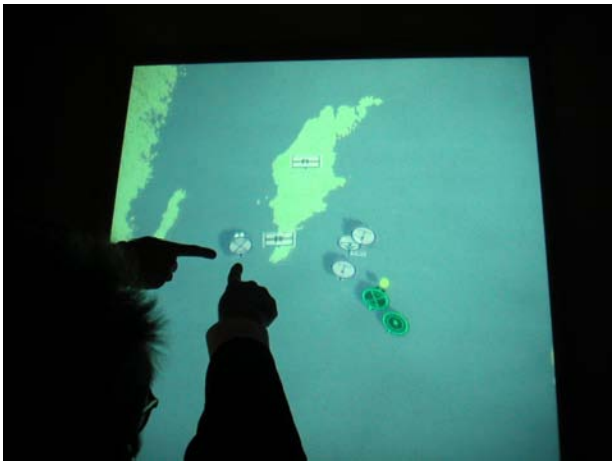
3 View Dependent Content Visualization in Geo-Spatial Context

Visualizations in command and control systems present geo-spatial data as well as content visualization where symbols describing military functions and objects have a prominent role. The described multiple viewer display environment is capable of presenting military entities using 3D objects but research suggests that classification and identification of military entities using 3D objects in most cases are more difficult and takes longer time to perceive than using traditional symbols [Smallman et al. 2001]. We have chosen to use traditional military domain 2D symbols in a 3D environmental setting. Since the multiple viewer display environment is capable of view-dependent co-located visualization, the traditional 2D symbols can be presented independently for each user of the display environment. A trivial but effective view-dependent rendering technique has been used to separate the presentation of military domain symbols from the presentation of geographic informa-

Figure 4: The control program for military domain visualization

tion. The view-dependent rendering technique presents the military domain symbols oriented orthogonally to the gaze of each viewer. This presentation technique where an object is correctly rotated and orthogonal to the viewers gaze is called frontoparallel presentation. Frontoparallel presentation should as well be effective for presenting text labels and other geographically tied symbols.

In figure 5, two users are shown collaborating in the multiple viewer display environment. They are provided with two perspectives of the same geographic area. For each view the symbols are frontoparallel towards the head tracked position of the user. The two views are the view from south, figure 5(a) and the view from the west, figure 5(b). In this scenario the users are discussing one particular symbol, and as can be seen in the figure, the symbol they point to is frontoparallel to both users views while remaining in the same position in the geographic area. The observable difference between frontoparallel and flat horizontal presentation is used to represent inactive objects, which are rendered flat in the horizontal plane but still oriented towards the spectator.



(a) South

(b) West

Figure 5: Two users discuss a situation in the same geographic area with military domain symbols frontoparallel to each users perspective.

4 Discussion and Conclusion

We have used our distribution model based on a shared repository to develop different visualization scenarios for the multiple display visualization environment. From a perceptional point of view, we have encountered that latencies in our distributed rendering process are at a very short time levels, which results in no perceived delays of animated structures among different viewers. A supporting factor in this context is that individual observers in the viewing environment do not actually perceive the views of the other collaborators. Still, the users must be able to get a common sense in regard to the temporal alignment when objects have started moving in the scenario and when they have terminated their motion. This temporal alignment is the common sensation of a shared dynamic scene is however, not as time critical and does not require strict frame-to-frame synchronization of the shared scene graph. Given our current configuration, whereby one rendering node renders the two stereoscopic views for one observer, the rendering of the left-eye and right-eye frames are always strictly frame synchronized, since they are rendered into different viewports within the same physical frame buffer. Hence, undesired parallax artifacts due to potential frame delays on the left eye and right eye, respectively, are not possible. In fact, the distribution model described in this paper would allow for quickly scaling up the rendering process upon 8 independent rendering nodes. This would not even require a recompilation of the system. Instead eight rendering processes would be instantiated on eight network nodes, whereby the configuration file for the local rendering processes would be modified to specify which respective projector and sensor entity should be used. In this configuration, there is a potential risk for frame delays between the observer's left and right eye view. Since our distribution model advocates a dead reckoning approach to distributed animation, asynchronous frames due to different rendering performances of the local nodes are likely. Since this type of configuration is only hypothetical, we have only conducted informal tests on this setting, which showed, that perceivable artifacts become visible. However, our shared repository approach is general enough such as to be used for synchronization of objects animations, even if it would not guarantee strict concurrency. Our measurements from previous experiments showed [Seipel and Ahrenberg 2002], that the

number of frame out-of-synch can be kept to as low as two frames in a sequence of 100 animation frames for a modestly sized scene. One main objective of our shared repository approach was to enable and maintain flexible configurations of viewports and viewer positions. The shared memory model is based on small data packets that are distributed asynchronously in order to minimize low latencies in network propagation. For the viewpoint-dependent visualization this is an important criterion to maintain the illusion of virtual 3D objects. The tracking values supplied by various device managers in our environment, are propagated rapidly to the local rendering nodes in order to accomplish dynamic viewing conditions. Our preliminary user studies indicate that delays in viewing frustum updates using sensor objects are not perceivably longer as compared to configurations that interface directly to the tracking equipment. We measure this in terms of the user's subjective experience of the three dimensional appearance of objects, which is largely dependent on dynamic parallax. In this context we can state, that three-dimensional appearance of virtual objects is equal for configurations that use shared sensor-entities and for configurations that interface immediately to the tracking hardware. Our visualization system has been developed with the goal to support flexible viewport handling and complex viewing parameterizations. This was a property, which we have missed in other previously published distributed rendering architectures (see section two). The solution that we have described here allows for controlling the visualization from other applications than the actual rendering processes. In our current visualization application, different phases of a military scenario are controlled with a pen-based wireless computer interface as shown in figure 4. In fact, a separate process performs the selection of the appropriate viewport and perspective to be rendered by a rendering client at a given time. Rendering clients are naïve processes in regard to viewpoint control. This opens up for new opportunities in collaborative work. The scenario presented above demonstrates collaborative work in the same geographical context. Another potential working situation would be that a workgroup coordinator (from a separate user interface), delegates planning tasks in different topographic regions to the four different users at a time. In this situation, the views upon the geographical visualization would temporarily be split up into distinct local regions, and after task completion, all observers would again gather around the same global view upon the scenario. This feature as well as user triggered private detail views into the geographic context can be managed very flexibly by means of the projector-entities in the shared repository. Another example that demonstrates the flexibility of the distribution model is the case of a user walking around the horizontal display. While walking from one side over to the next side of the screen, the user will, due to the optical properties of the screen, leave one viewing zone and enter the next one. Different rendering nodes will provide the views within the respective viewing zones, and their viewport orientation will shift 90 degrees. In the military domain visualizations it has been assumed that frontoparallel 2D symbols are easier to perceive than 2D symbols presented in the plane of a horizontal display. An experiment comparing the efficiency in identifying frontoparallel and flat horizontal presentation has been carried out and gathered data are being evaluated. Preliminary results indicate that frontoparallel presentation indeed has a lower error rate and faster recognitions times when compared flat horizontal presentation.

5 Future Work

The visualization environment as described in this paper suffers from apparent crosstalk between stereo images pairs, as is the case for most visualization systems that present different information for multiple viewpoints in the same physical display area. However, according to interview studies with domain experts using the visualizations, stereoscopic crosstalk was not reported as major disturbing artifact. In order to quantify this further it would be interesting to study in more detail how tolerant the human visual system is in regard to visual crosstalk between the stereo image pairs and other visual noise. These studies would require a definition and setup of a measuring procedure that is generalized and portable to other display environments. In regard to the frontoparallel presentation of 2D symbols, we are conducting further experimental research on the human's capability in reading this form of presentation as compared to traditional flat presentations.

6 Acknowledgements

The work presented in this paper is result of a sub-project funded by Swedish National Defence College (SNDC) in the framework of project AQUA. Technical details on the display hardware used are proprietary of S. Anders Christenson at SNDC and Peter Segerhammar at VAB and are published elsewhere by their authors. We acknowledge their admission to reproduce schematically the working principle of the multiple projector display device as required for the comprehension of the presented multiple viewer 3D visualization system.

References

AGRAWALA, M., BEERS, A. C., MCDOWALL, I., FRÖHLICH, B., BOLAS, M., AND HANRAHAN, P. 1997. The two-user responsive workbench: support for collaboration through individual views of a shared space. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniquesarrraraa*, ACM Press/Addison-Wesley Publishing Co., 327–332.

ALLARD, J., GOURANTON, V., LECOINTRE, L., MELIN, E., AND RAFFIN, B. 2002. Net juggler: Running vr juggler with multiple displays on a commodity component cluster. In *Proceedings of the IEEE Virtual Reality Conference 2002*, IEEE Computer Society, 273.

AZUMA, R. 1997. A survey of augmented reality. *Presence, Teleoperators and Virtual Environments 6*, 4, 355–385.

CARLSSON, C., AND HAGSAND, O. 1993. Dive - a platform for multi-user virtual environments. *Computers and Graphics 17*, 6, 663–669.

HALLE, M. 1997. Autostereoscopic displays and computer graphics. *Computer Graphics 31*, 2, 58–62.

HEDLEY, N. R., BILLINGHURST, M., POSTNER, L., MAY, R., AND KATO, H. 2002. Explorations in the use of augmented reality for geographic visualization. *Presence: Teleoper. Virtual Environ. 11*, 2, 119–133.

HIRSCH, M., 1961. Three dimensional display apparatus. U. S. Patent No. 2,967,905.

HUMPHREYS, G., HOUSTON, M., NG, R., FRANK, R., AHERN, S., KIRCHNER, P. D., AND KLOSOWSKI, J. T. 2002. Chromium: a stream-processing framework for interactive rendering on clusters. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, 693–702.

KETCHPEL, R. D., 1964. Three-dimensional cathode ray tube. U. S. Patent No. 3,140,415, July.

KITAMURA, Y., KONISHI, T., YAMAMOTO, S., AND KISHINO, F. 2001. Interactive stereoscopic display for three or more users. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 231–240.

LINDKVIST, M. 2002. *A State Sharing Toolkit for Interactive Applications*. Master's thesis, Royal Institute of Technology.

LINDSTROM, P., KOLLER, D., RIBARSKY, W., HODGES, L., AND FAUST, N. 1997. An integrated global gis and visual simulation system. Misc GIT-GVU-97-07, Georgia Institute of Technology.

SEIPEL, S., AND AHRENBERG, L. 2002. Distributed rendering in heterogeneous display environments - a functional framework design and performance assessment. In *Proc. Annual SIGRAD Conference 2002*.

SINGHAL, S., AND ZYDA, M. 1999. *Networked virtual environments: design and implementation*. ACM Press/Addison-Wesley Publishing Co.

SMALLMAN, H. S., OONK, H. M., AND JOHN, M. S. 2001. Symbicons: Advanced symbology for two-dimensional and three-dimensional displays. Tech. Rep. TR-1850, SPAWAR Systems Center San Diego.

SUNDIN, C., AND FRIMAN, H., Eds. 2000. *ROLF 2010 - The Way Ahead and The First Step*. Elanders Gotab.

SUTHERLAND, I. E. 1968. A head-mounted three-dimensional display. In *AFIPS Conference Proceedings*, Thompson Book Co., vol. 33, 757–764.