Trends in Networked Collaborative Virtual Environments
Igor S. Pandzic, Chris Joslin, Nadia Magnenat Thalmann
MIRALab – CUI, University of Geneva
24 rue du Général-Dufour, CH1211 Geneva 4, Switzerland
{Christopher.Joslin, Igor.Pandzic, Nadia.Thalmann}@cui.unige.ch

Abstract
Networked Collaborative Virtual Environments (NCVE) are systems allowing multiple geographically distant users to share the same 3D Virtual Environment using the network. The paper presents an overview of the developments in the field of NCVE in the past decade with an introduction of research challenges and solutions for such systems and a brief presentation of systems that brought major developments to the NCVE field. As a case study, we present a new generation NCVE system VPARK.

Keywords: Networked Collaborative Virtual Environments, collaborative work

Introduction
Trends towards networked applications and Computer Supported Collaborative Work, together with a wide interest for graphical systems and Virtual Environments, have in the recent years raised interest for research in the field of Networked Collaborative Virtual Environments (NCVEs). NCVEs are systems that allow multiple geographically distant users to share a common three-dimensional Virtual Environment (VE). NCVEs are a powerful tool for communication and collaboration, with potential applications ranging from entertainment and teleshopping to engineering and medicine. Therefore it is not surprising that in the recent years we have seen active research on this topic in both academic and industrial research establishments. A number of working systems exist [Barrus96, Carlsson93, Macedonia94, Ohya95, Singh95, Zyda93]. They differ largely in networking solutions, number of users supported, interaction capabilities and application scope [Macedonia97], but share the same basic principle.

Challenges in NCVE system design
The concept of a NCVE is fairly simple (see Figure 1). Two or more users can view the Virtual Environment (VE) on their work stations. Each work station has a local copy of the VE. As the actions occur on one work station, they are propagated through the network to other work stations in order to keep all copies of VE synchronized. The users become themselves part of the VE, placed within the VE at the position of their view point and represented by a graphical embodiment called avatar. This allows the users to see each other and visualize the other users’ actions.

Despite the fairly simple concept, the design of NCVE systems involves a complex interaction of several domains of Computer Science:

- **networking**, involving communication of various types of data with varying requirements in terms of bitrate, error resilience and latency
- **virtual environment simulation**, involving visual data base management and rendering techniques with real time optimizations
- **human-computer interaction**, involving support of different devices and paradigms
- **virtual human simulation**, involving real time animation/deformation of bodies and faces
• artificial intelligence (in case autonomous virtual humans are involved), involving decision making processes and autonomous behaviors

In the following section we will explore the particular challenges that arise in the implementation of a NCVE system.

![Figure 1: Principles of Networked Collaborative Virtual Environments](image)

**Scalability**

Scaleability of NCVE systems is a measure of how well the system behaves when the number of users increases. We discuss scalability in a separate section because most of the research topics in the following sections have an impact on scalability, and some of them are exclusively aimed at improving it.

An ideal scalable NCVE system would be the one that could support infinite number of users without any degradation of the quality of service to each user. This is obviously impossible, therefore the real systems try to achieve graceful degradation of quality of service with increasing number of users in such a way that it disturbs the already connected users minimally. This usually relies on assumptions about users’ needs and behavior. For example, the user might be able to achieve high quality communication with other users that are close to them in the virtual environment, and only rudimentary communication with those that are further apart or in other rooms, imitating the real-life behavior.

**Network topologies**

Figure 2a) schematically represents a session of a NCVE with several participating hosts. If an event occurs at host 1, it is in general necessary for a message about that event to reach all other hosts. How this message is transferred is a question of network topology, i.e. the inside of the cloud in Figure 2a).

Transmitting any event that happens on any host to all other hosts requires a lot of network traffic, growing with \(O(N^2)\) where \(N\) is the number of users. Fortunately, not all events are essential for all hosts. For example, if two users are very far from each other in the virtual world and can not see each other, they do not need to know about each other’s movements until they are close enough to see each other; therefore their respective hosts do not need to exchange messages. Deciding which hosts need to receive which messages, and pruning the unnecessary messages is called Area of Interest Management (AOIM) or filtering. We will discuss AOIM strategies in more detail in the section on space structuring. In this section we only discuss AOIM in terms of its deployment within different network topologies that we present.
When discussing different network topologies for NCVEs another important issue is **session management**. This includes the procedures for a new user to join a session, leave the session, as well as a strategy for maintaining persistent virtual worlds when no users are present.

The main network topology solutions for NCVE systems are presented in Figure 2 as follows:

- b) peer-to-peer
- c) multicast
- d) client/server
- e) multiple servers

**Space structuring**

In case of a simple virtual environment consisting of a single room or similar simple space, space structuring is not an issue. However, it becomes an important issue if one tries to model large scale environments like cities or battlefields inhabited by a large number of users. It is simply impossible to keep such large structures monolithic. Memory and download time are just some of problems. Multitudes of users that will inhabit these complex environments must be managed smartly in order to avoid network congestion. Space structuring is closely tied to AOIM which, as mentioned in the previous section, is a strategy to reduce the total network traffic by sending messages to hosts on an as-needed basis.

Another problem that occurs with large scale environments is that of coordinate inaccuracy. As the environment grows larger, the precision with which coordinates can be represented drops because of the inherent imprecision of large floating point numbers. For example, a 32 bit floating point number of the order of magnitude 10E6 has a precision of 0.06. This means that an object positioned in a virtual environment at 1000 km from the origin can be placed with only 6 cm precision [Barrus96].

In Figure 3 we present following typical strategies for space structuring:

- a) separate servers; each server manages a separate world, with discrete links between them
- b) uniform geometrical structure; the world is divided into uniform cells
- c) free geometrical structure; the world is divided into freely shaped parts
- d) user-centered dynamic structure; the grouping of users is dynamic and can be personalized (this strategy can be combined with the others)
Real time graphical simulation

Real time 3D graphical simulation is a vast research area and we do not intend to make any sort of extensive survey as it would be out of place in this work. However, real time 3D graphics being at the core of NCVEs, the topic does deserve attention. This mostly means that the designer of an NCVE system will have either to choose a real time graphics toolkit such as IRIS Performer [Rohlf94] or OpenGL Optimizer, or design a 3D graphics environment supporting similar functionalities that are nowadays standard, such as a hierarchical scene graph, level of detail management, multi-threading of graphical processes.

User representation

User representation determines the way users perceive each other in the VE, and is therefore an extremely important factor for the quality of a NCVE system.

The solutions range from primitive cube-like appearances [Greenhalgh95], non-articulated human-like or cartoon-like avatars [Benford95] to articulated body representations using rigid body segments [Barrus96, Carlsson93, Pratt97]. Ohya et al. [Ohya95] report the use of human representations with animated bodies and faces in a virtual teleconferencing application.

The Virtual Life Network (VLNET) [Capin99, Pandzic97] system introduced high-level, fully deformable virtual humans with various possibilities for facial and gestural communication.

Human communication

NCVE systems are basically communication systems so communication and collaboration between people should be on the top of the agenda. Actually, the fact that the users share the same virtual environment and can interact with it simultaneously enhances their ability to communicate with each other. However, the means of communication that are taken for granted in real life - speech, facial expressions, gestures - are not necessarily supported in NCVE systems due to technical difficulties. Most systems support audio communication, and very often there is a text-based chat capability [Barrus96, Greenhalgh95]. Some systems [Carlsson93, Pratt97] include a means of gestural communication by choosing some predefined gestures. The natural human communication is richer than this. Facial expressions, lip movement, body postures and gestures all play important roles in our everyday communication. Ideally, all these means of communication should be incorporated seamlessly in the Virtual Environment, preferably in a non-intrusive way. Ohya et al. [Ohya95] recognize this need and present a system where facial expressions are tracked using tape markers while body and hands carry magnetic trackers, allowing both face and body to be synthesized. In our own work [Pandzic97] we propose further improvements in the area of facial communication.

Landmark NCVE Systems

In the following sections we present several representative NCVE systems. The systems presented here are either representative of a particular system category (academic, commercial, game) or they introduced novel approaches and developments at the time of their conception.

NPSNET

NPSNET is a networked virtual environment developed at the Computer Science Department of the Naval Postgraduate School (NPS) in Monterey, California [Zyda93, Macedonia94]. It is developed specifically for large-scale military simulations. It is one of the first NCVE systems and has known several generations of new and improved versions.
NPSNET can be used to simulate an air, ground, nautical (surface or submersible) or virtual vehicle, as well as human subjects. A virtual vehicle, or stealth vehicle, is a vehicle that can navigate in the virtual world but has no graphical representation and is therefore not seen by others. The standard user interface devices for navigation include a flight control system (throttle and stick), a six degree of freedom SpaceBall, and/or a keyboard. The system models movement on the surface of the earth (land or sea), below the surface of the sea and in the atmosphere. Other entities in the simulation are controlled by users on other workstations, who can either be human participants, rule-based autonomous entities, or entities with scripted behavior. The virtual environment is populated not only by users’ vehicles/bodies, but also by other static and dynamic objects that can produce movements and audio/visual effects.

NPSNET uses the Distributed Interactive Simulation (DIS 2.03) protocol [IEEE 93] for application level communication among independently developed simulators (e.g. legacy aircraft simulators, constructive models, and real field instrumented vehicles).

**DIVE**

DIVE (Distributed Interactive Virtual Environment) [Carlsson93] is developed at the Swedish Institute of Computer Science. The DIVE system is a tool kit for building distributed VR applications in a heterogeneous network environment. It is another early system that continues to be developed and improved over the years.

The DIVE run-time environment consists of a set of communicating processes, running on nodes distributed within a Local Area Network (LAN) or Wide Area Network (WAN). The processes, representing either human users or autonomous applications, have access to a number of databases, which they update concurrently. Each database contains a number of descriptions of graphical objects that together constitute a virtual world. Objects can be added or modified dynamically, and concurrently using a distributed locking mechanism. Multicast protocols are used for the communication simulating a large shared memory for a process group through the network [Birman91].

**BRICKNET**

The BrickNet toolkit [Singh95], developed at the National University of Singapore, provides functionalities geared towards enabling faster and easier creation of networked virtual worlds. It eliminates the need for the developer to learn about low level graphics, device handling and network programming by providing higher level support for graphical, behavioral and network modeling of virtual worlds. BrickNet provides the developer with a “virtual world shell” which is customized by populating it with objects of interest, by modifying its behavioral properties and by specifying the objects’ network behavior. This enables the developer to quickly create networked virtual worlds.

BrickNet applies the multiple server network topology Servers, distributed over network, communicate with one another to provide a service to the client without its explicit knowledge of multiple servers or inter-server communication.

BrickNet introduces an object sharing strategy which sets it apart from the classic NCVE mindset. Instead of all users sharing the same virtual world, in BrickNet each user controls their own virtual world with a set of objects of their choice. They can then expose these objects to the others and share them, or choose to keep them private. The user can request to share other users’ objects providing they are exposed. So, rather than a single shared environment, BrickNet is a set of “overlapping” user-owned environments that share certain segments as negotiated between the users.
BrickNet does not incorporate any user representation, so the users are body-less in the virtual environment and their presence is manifested only implicitly through their actions on the objects.

<table>
<thead>
<tr>
<th></th>
<th>User representation</th>
<th>Audio communication</th>
<th>Text communication</th>
<th>Gestural communication</th>
<th>Facial communication</th>
<th>Network topology</th>
<th>Space structuring</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPSNET</td>
<td>Articulated rigid-segment body</td>
<td>Yes</td>
<td>Yes</td>
<td>Predefined gestures/behaviors</td>
<td>No</td>
<td>Multicast</td>
<td>Uniform</td>
<td>Naval Postgraduate School, Monterey, CA, USA</td>
</tr>
<tr>
<td>DIVE</td>
<td>Articulated rigid-segment body</td>
<td>Yes</td>
<td>Yes</td>
<td>Predefined gestures</td>
<td>No</td>
<td>Multicast</td>
<td>Dynamic</td>
<td>Swedish Institute of Computer Science, Stockholm Sweden</td>
</tr>
<tr>
<td>BrickNet</td>
<td>None</td>
<td>N/A</td>
<td>N/A</td>
<td>No</td>
<td>No</td>
<td>Multiple servers</td>
<td>Separate servers</td>
<td>University of Singapore</td>
</tr>
<tr>
<td>VLNET</td>
<td>Fully deformable body</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Client/server</td>
<td>Separate servers</td>
<td>University of Geneva &amp; EPFL, Switzerland</td>
</tr>
<tr>
<td>BLAXXUN</td>
<td>Articulated rigid-segment body</td>
<td>No¹</td>
<td>Yes</td>
<td>Predefined gestures</td>
<td>No</td>
<td>Client/server</td>
<td>Separate servers</td>
<td>Blaxxun Interactive Inc., Germany &amp; USA</td>
</tr>
<tr>
<td>DOOM</td>
<td>Articulated rigid-segment body</td>
<td>No²</td>
<td>No</td>
<td>Predefined gestures</td>
<td>No</td>
<td>N/A</td>
<td>Free</td>
<td>id Software, USA</td>
</tr>
</tbody>
</table>

¹Text-to-Speech is available to convert chat messages into speech.  
²The users basically shoot each other in the game; this kind of “communication” is adequately supported by sound effects.

Figure 4: Overview of representative NCVE systems

**VLNET**

Virtual Life Network (VLNET) [Capin99, Pandzic97] is a client/server NCVE system that introduced high-level Virtual Humans for user representation and communication. The system also supports a modular architecture concept allowing extended system capabilities using plug-ins or to connect the system to complete applications, thus giving a graphical, networked front-end to those applications. In particular, gestural and facial communication are supported. The facial communication includes live video analysis for tracking of facial features, video texturing with face region tracking, predefined emotions and lip synchronization to the audio signal [Pandzic96].

**BLAXXUN COMMUNITY PLATFORM**

This system from Blaxxun Interactive Inc. [Blaxxun] is the first commercially successful NCVE system. It is used to create on-line 3D communities where people can meet, chat or even trade. Through a set of modules, the system supports a rich set of functionalities such as pre-programmed agents, personal homes that can be set up by participants, personalized avatars, access rights control, text-to-speech, etc. It is a client/server system. The client is a free of charge and comes in form of a Web browser plug-in, and the commercial benefit is obtained from customers installing the servers.
for their users. Blaxxun claims that the Community Server can support more than 20000 users simultaneously with a T3 (45 Mbits) connection; these numbers are reduced by the server’s CPU capacity. The company claims that a Pentium 500 machine running Linux supports approx. 2000 simultaneous users. It is however less clear how each client would cope with information from so many other clients.

**DOOM**

Doom is a computer game that reached tremendous popularity due largely to the networked aspect: groups of players can share the game through the network. In the first version the network traffic was not at all optimized and created large network traffic on the LANs where the game was played. It is remarkable that approx. 15 million copies of the shareware version of Doom were downloaded, and approx. 250000 copies of the full product were sold. Doom has been criticized for its content (basically, it is shooting monsters) but it is probably the most-used example of a NCVE system!

**Case study: the VPARK system**

Virtual Park, or VPARK, was designed and created on a Windows Operating System (OS), which makes it almost unique. The system itself was derived partially from the previous system, based on UNIX, called VLNET [Capin97, Pandzic97]. VPARK actually consists of two systems, the NVE System and the Attraction Builder. An Attraction is much like any other attraction (such as a circus act or a theatre production) it consists of actors playing a role and giving a performance. In this instance the Attraction itself is virtual, and constructed of virtual objects and actors. The Attraction Builder enables the assembly of these different components into one Attraction. It does not actually create objects and actors, but takes existing ones and animates them according to a set of rules. These actor and object descriptions use standard formats (VRML97 for Objects and HANIM/MPEG4 [HANIM, MPEG] for the Actors) and therefore can be obtained basically anywhere. There are many different animations that can be applied to these objects/actors; the following list gives an overview:

- Animation of Objects using Key frames
- Body Animation, of Actor, using MPEG4 Body Animation Parameters (BAP)
- Face Animation, of Actor, using MPEG4 Face Animation Parameters (FAP)
- Speech using a Text-to-Speech/Viseme Converter
- Proximity Sensors to create Animated Sensors triggered by proximity
- Touch Sensors used to initiate a sequence through the touching of a sensors
- User interactive breakpoints, allowing users to interact with an Actor (given simple questions)
- Background Sounds to add effects to different situations

The Attraction runs along a special timeline that, under normal circumstances, runs from beginning to end with specific points on the line initiating a specific animation sequence. Alternatively with the breakpoints, touch sensors and proximity sensors the time-line can be stopped and specific points jumped to, making the attraction completely interactive (depending on the amount placed into the scene), for more details see [Seo00].

The NVE System takes the output of the Attraction Builder (a compressed file, which contains all the data necessary to run the attraction, i.e. objects, avatars and animation information). The System itself consists of a Client/Server Network configuration, with the Server controlling the Clients dataflow between each other and the Clients themselves representing the users. The basic architecture was taken from the previous system VLNET, with some basic changes to incorporate it into the PC architecture. The VLNET architecture on UNIX systems used separate processes to
perform specific tasks (such as animation, capture from devices etc) and a shared memory scheme to provide communication between these processes. However under the Windows OS, this is not feasible as the shared memory architecture is too slow and also Threads are more popular on under Windows and therefore were chosen as the multitasking container. Threads do not need special memory to communicate; as they can share the same memory space [Seo00] gives more details on this specific architecture. In addition to this architecture change for concurrency and communication, the entire Windows OS System was design around plugins to enable an expandable system with which units could be added, changed or removed without need to recompile the system. Each plugin is responsible for a certain task, whilst there are a few which are necessary to the general functionality of the system (such as the Scene Manager, containing the Scene Graph, and the Network Manager), the rest of the plugins are optional (such as a Flock of Birds Driver, or Speech Engine) and can be added or removed as needed. The entire system is constructed out of these plugins that perform separate tasks, the current ones are as follows:

- Scene Manager – Controls all aspects of the Scene Graph
- Network Manager – Controls Data flow between Client and Server
- Navigation – Controls Users movements around Scene
- Speech – Enables Speech Communication in the NVE
- Audio – Adds Audio Communication to the NVE
- Text – Used for sending simple Text Messages to other Clients
- Motion Capture System – To connect to Motion Tracking Systems

The Network is a basic Client/Server architecture and consists of one server that holds the information regarding the Attraction (from the Attraction Builder), and Clients that connect to it to download this information. After a Client has connected, the Client and Server negotiate the necessary Channels; there are 4 channels (Control, Stream, Update and File) that are used for different types of data. The Control and Update Channels are necessary and come as standard, the former is used for inter Client/Server communication and the latter for updates to the Scene (both use Error control). The Stream Channel is for Video and Audio streams and is not error controlled. The File Channel is used to upload and download files, but if the bandwidth is small (as the files are often between 500K and 2M) the File channel connection can be omitted, instead the Client can opt for Caching. The caching mechanism is used to determine whether download is necessary (or if the File Channel is not present), if so the compressed file containing the Attraction is sent to the Client using the File Channel, otherwise a local copy of the file is used. The Scene Manager sets up the scene according to the file it has received and then data regarding the movement of Avatars (Users) and Objects (from picking) is transferred via the Update Channel to the Server that distributes it accordingly. Apart from the distribution of data, the Server has two other main functions: It firstly acts as a database for all Clients, providing a master role in terms of data validity and secondly it provides a filtering mechanism to reduce bandwidth. The Database stores the location of all Users Avatars (as more than one can be used), their postures (in terms of body animation) and any other miscellaneous information. The filtering works only for the stream channel at present and is used to determine whether a Clients Avatars (as they see the scene through the Avatars) can see or hear a video or audio stream, the Server determines their position through its database. If the visual or auditory information will not reach the avatar, then the information is filtered from the out-going data stream (which saves bandwidth).

To exemplify both systems a specific scenario is given, this was used to test and validate both systems and also to enable extensive testing. The scenario is an Attraction for a virtually represented
teacher (attached to a motion tracking system at one location) to teach a virtual represented student (also attached to a motion tracking system in another geographically remote location). Both teacher and student were cloned by our in-house cloning system, to obtain a virtual copy of both humans, and attached to a motion tracking system (one at University of Geneva and the other at Ecole Polytechnique Federal de Lausanne, both in Switzerland). The Motion Tracking System (as shown in Figure 5) is attached to the separate limbs of the user and tracks the exact movement of the users body and head. This information is compressed and sent via the Server to the other Client. An overlaid musical sequence is used to enable the teacher and student to synchronize with each other, both teacher and student can see each other (virtually) on a screen (as shown in Figure 6) and therefore the teacher is able to see what the student is doing wrong and the student can watch the teacher to see what should be done.

The system is fully interactive, allowing each participant the ability not only to see the exact movements of their counterparts, but also to talk with each other. This type of scenario is classical of an NVE System being used to its maximum benefit and certainly is difficult to replace with other conventional systems (such as Video Conferencing). If more users were to join the session, then the teacher would have no problem in clearly visualizing the class’s response. To increase the teacher sense of submersion and also to enable a clearer perception of the situation a lightweight head mounted display could be used, although as dance typically uses great movement, the display should be rugged and should secure to the teacher so that the movement is not restricted.

![Figure 5: Teacher in Motion Tracking Equipment](image1) ![Figure 6: Virtual Teacher and Student Dance together](image2)

**Conclusions**

We have presented the basic concepts and technological and research challenges of Networked Collaborative Virtual Environments, and briefly introduced a contemporary, PC-based NCVE system being developed jointly by the University of Geneva and Swiss Federal Institute of Technology in Lausanne (EPFL). It is obvious that, as technological challenges are overcome, NCVE systems become more and more powerful communication tools. We are currently seeing first serious commercial deployments of this kind of technology. A wider usage is expected in the years to come as the technology becomes more accessible.
Acknowledgments
The VPARK development is financed by the European Commission through the project VPARK in the ACTS programme. The previous research leading to the insights presented in this paper has been financed by the Swiss SPP programme and the ACTS project COVEN.

References


