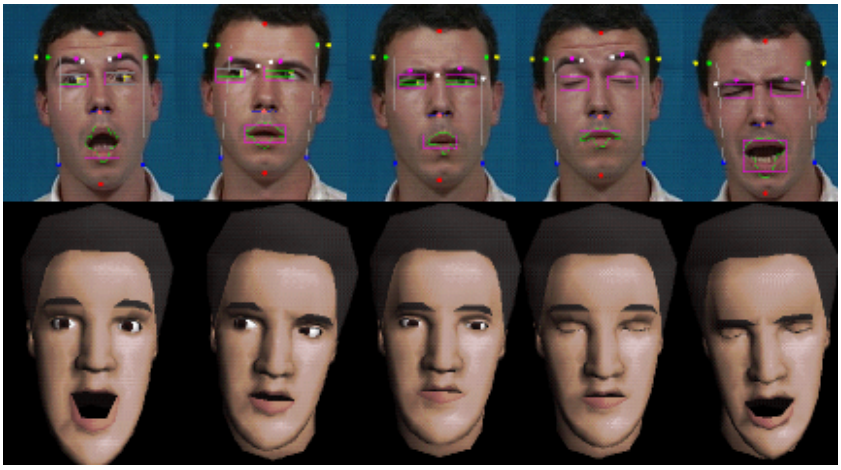
Facial Communication in Networked Collaborative Virtual Environments

-

La communication faciale dans les environnements virtuels en réseau

Igor Sunday Pandzic



Facial Communication in Networked Collaborative Virtual Environments

La communication faciale dans les environnements virtuels en réseau

THÈSE

présentée à la Faculté des sciences économiques et sociales
de l'Université de Genève

par Igor Sunday Pandzic

pour l'obtention du grade de
Docteur ès sciences économiques et sociales,
mention systèmes d'information

Membres du jury de thèse

Mme N. Magnenat Thalmann, Professeur, Université de Genève, directrice de thèse
M. Jean-Philippe Vial, Professeur, Université de Genève, Président du jury
Dr. Laurent Moccozet, Université de Genève
M. Murat Kunt, Professeur, Ecole Polytechnique Fédérale de Lausanne
M. Michael Zyda, Professeur, Naval Postgraduate School, Monterey, CA, USA

Thèse no 466

Genève, 1998

La Faculté des sciences économiques et sociales, sur préavis du jury, a autorisé l'impression de la présente thèse, sans entendre par là, émettre aucune opinion sur les propositions qui s'y trouvent énoncées et qui n'engagent que la responsabilité de leur auteur.

Genève, le 9 Mars 1998

Le doyen
Beat BÜRGENMEIER

Impression d'après le manuscrit de l'auteur

To my family

Membres du jury de thèse

Prof. N. Magnenat Thalmann, Directrice de Thèse
Département de Système d'Informations,
Faculté des Sciences Economiques et Sociales,
Université de Genève

Prof. J-P. Vial, Président du Jury
Ecole des Hautes Etudes Commerciales
Faculté des Sciences Economiques et Sociales,
Université de Genève

Dr. L. Moccozet,
Département de Système d'Informations,
Faculté des Sciences Economiques et Sociales,
Université de Genève

Experts externes:

Prof. Murat Kunt
Département d'Informatique
Ecole Polytechnique Fédérale de Lausanne

Prof. Michael Zyda
Naval Postgraduate School,
Monterey, CA, USA

Acknowledgement

This work has been done at MIRALab, University of Geneva, under the direction of Prof. Nadia Magnenat Thalmann, and in collaboration with the Computer Graphics Laboratory of the Swiss Federal Institute of Technology in Lausanne (LIG, EPFL). Since its foundation by Prof. Nadia Magnenat Thalmann in 1989 at the University of Geneva, MIRALab has dedicated most of its research efforts in the field of virtual human simulation. The work presented here is a contribution to the global project of the laboratory.

This research is partly financed by “Le Programme Prioritaire en Telecommunications du Fonds National Suisse de la Recherche Scientifique” and the European ACTS project VIDAS (VIDeo ASsisted with audio coding and representation). In particular, our participation in the work of the MPEG committee is in the framework of the ACTS project VIDAS.

My special thanks to the director of my thesis, Prof. Nadia Magnenat Thalmann for her guidance, support and encouragement.

My deepest appreciation to my friend and colleague Tolga Capin of LIG, EPFL who collaborated in this work since the beginning, as well as to Elwin Lee who brought a significant contribution in the later phase of the work.

My thanks to Prem Kalra for valuable advice during the work, as well as a very thorough reading of the manuscript.

Finally, I extend my thanks to all members of MIRALab, University of Geneva and LIG, EPFL, both past and present, for their cooperation and support during this work.

Abstract

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.

One of the particularly important research challenges in NCVEs is the user representation, the way how participants are graphically represented in the VE. This can range from very simple block-like representation to highly realistic Virtual Humans with articulated bodies and faces. It is observed that most of the existing NCVE systems use rather simple representations, however research indicates that better user representation can improve users' sense of presence in the environment and their ability to communicate with each other.

In this thesis we analyze the problems involved with using sophisticated Virtual Humans for user representation and propose a framework for NCVE integrating Virtual Humans in an optimal and flexible way. One of the salient characteristics of NCVEs is that they allow distant users to feel as if they were *together*. To enhance this feeling it is important to allow natural means of communication. Most of the existing NCVE systems support audio and/or textual communication. However, facial expressions, lip movement and gestures, which are very important components of our everyday communication, are usually not supported.

Based on the NCVE framework developed as the first part of this work, we propose four techniques to support facial communication in NCVEs:

- mapping of the video of participant's real face on the virtual face
- model-based coding of facial expressions
- speech-based lip movement

- use of predefined facial expressions

We analyze the advantages and disadvantages of all proposed approaches with respect to the obtained quality of facial expressions, required bandwidth and suitability for different applications.

In parallel with our work on NCVEs, we participate in the work of ISO/IEC JTC1/SC29/WG11 - better known as MPEG. Within the MPEG-4 Ad Hoc Group on Face and Body Animation we have provided a major contribution to the specification of Face Animation Parameters and Face Definition Parameters. This experience has lead us to believe that there is a strong potential relation of MPEG-4 standard to NCVEs and that it will be possible in near future to build rich multimedia 3D networked environments based on this standard. As a part of our work we analyze the potential usage of MPEG-4 for NCVE systems.

Table of Contents

1. 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) [Durlach95]. NCVEs are systems that allow multiple geographically distant users to interact in a common virtual environment. The users themselves are represented within the environment using a graphical embodiment.

Networked Collaborative Virtual Environment (NCVE) systems are suitable for numerous collaborative applications ranging from games to medicine [Doenges97, Zyda97], for example:

- Virtual teleconferencing with multimedia object exchange
- All sorts of collaborative work involving 3D design
- Multi-user game environments
- Teleshopping involving 3D models, images, sound (e.g. real estate, furniture, cars)
- Medical applications (distance diagnostics, virtual surgery for training)
- Distance learning/training
- Virtual Studio/Set with Networked Media Integration
- Virtual travel agency

In this chapter we present the motivation and objectives of the undertaken research and we outline the organization of the rest of the thesis.

1.1 Motivation

Networked Collaborative Virtual Environments (NCVE) have been an active area of research for several years now, and 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.

Several aspects of NCVE systems have been subject to thorough research with interesting results: scaleability and network topologies [Macedonia94, Singh95, Funkhouser96], efficient space structuring [Barrus96, Benford95], real time simulation [Rohlf94], feeling of presence in NCVEs [Benford95, Welch96, Hendrix96, Tromp95].

However, some aspects are still missing or not enough developed, in particular the participant representation and some aspects of human communication within NCVEs.

Within Networked Collaborative Virtual Environments, the participant representation can have several important functions:

- perception
- localization
- identification
- visualization of interest focus
- visualization of actions
- communication

We believe that the simulation of highly realistic Virtual Humans [Boulic95] for participant representation in NCVEs can fulfill these functions. However, in most existing systems the participant representation is rather crude, often using non-articulated objects as embodiments. It is necessary to provide a framework for including highly realistic, articulated and deformable Virtual Humans into the NCVE, capable of both body and face animation. This framework should include:

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

Each of the involved components represents in itself an area of research. When combining them together the interaction between components and their impact on each other have to be considered. This makes the development of the proposed framework a complex task and imposes a careful design of software architecture to use.

The Networked Collaborative Virtual Environments are often described as systems that permit the users to feel as if they were together in a shared Virtual Environment. Indeed, the feeling of "being together" is extremely important for collaboration. Probably the most important aspect of being together with someone, either in the real or a virtual world, is the ability to communicate. A function that Virtual Humans can fulfill, and that we find particularly interesting, is allowing more natural communication through facial expressions and gestures. Facial expressions, lip movements, body postures and gestures all play an important role in our everyday communication. Therefore we find that they should have their place also within the NCVE systems.

In most of the existing NCVE systems the communication between participants is restricted to text messages and/or to audio communication [Barrus96, Greenhalgh95, Singh95]. Some systems [Carlsson93, Pratt97] includes a means of gestural communication by choosing some predefined gestures or simple behaviors. The natural human communication is richer than this. Recognizing this problem, Ohya et al. [Ohya95] present a virtual teleconferencing system where facial expressions are tracked using tape markers while body and hands carry magnetic trackers, allowing both face and body movements to be synthesized.

We propose a flexible framework for including Virtual Humans in Networked Collaborative Virtual Environments, and use this framework to explore different means of communication in such environments. We concentrate on facial communication in particular.

1.2 Objectives

Our Ph.D. work concentrates on two main goals:

1. Development of a flexible framework for including Virtual Humans in Networked Collaborative Virtual Environments
2. Development of algorithms for facial communication within the mentioned framework

The first goal is in the development of a versatile and flexible NCVE architecture allowing the integration of Virtual Humans into a NCVE. Versatility and flexibility in this context means that the architecture should be open for easy extensions and suitable not only for development of various collaborative applications, but also as a testbed for integration of new techniques in NCVE.

In the second phase, the goal is to develop means for facial communication within the NCVE framework developed in the first phase. Although the face is a very important communication channel, conveying emotions and facilitating understanding through lip reading, facial communication is very scarcely supported in current NCVE systems. Our work strives to fill this gap by providing several methods of facial communication in NCVE. We propose four different methods, varying in bandwidth and computing power required, as well as quality and application scope.

- *Video texturing:* mapping of the user's facial video on the face of virtual human. This method offers high quality facial images suitable for video conferencing in 3D spaces, but it requires relatively high bandwidth as well as considerable computing power for compression/decompression of video.
- *Model based coding of facial expressions:* using image analysis, track facial features in real time based on camera input, then use the extracted parameters to reproduce the expressions on the remote virtual face. This method potentially offers good quality of facial expression and lip movement reproduction for 3D conferencing applications at an extremely low bitrate. However, computational complexity is quite high and algorithms for the extraction and tracking of facial features are not yet mature.

- *Predefined emotions:* user chooses emotions to reflect on his/her face from a menu. This method has low requirements on both bandwidth and CPU. It is suitable for chat applications.

- *Lip movement synthesis from speech:* lip movement is generated based on speech signal. This method potentially provides good quality of lip movement reproduction for easier speech comprehension, at an extremely low bitrate. However, computing requirements for speech signal processing may be prohibitively high, resulting in a tradeoff between the quality of lip movement reproduction and computing resources.

An additional goal of this thesis is to provide an analysis of the potential usability of the currently developed MPEG-4 standard in the field of NCVEs. This analysis is based on our active participation in the development of the MPEG-4 standard which is going on in parallel with our work on NCVEs.

1.3 Organization

The rest of this thesis is divided into five chapters.

Chapter 2 provides an overview of Networked Collaborative Virtual Environments (NCVEs). First, in section 2.1 we present the main research challenges in the domain of NCVE: scaleability, network topologies, space structuring, real time simulation, user representation, human communication and the sense of presence. Next, in section 2.2 we present the major existing NCVE systems. We analyze in particular the solutions for user representation and human communication in the existing systems, as these areas are the principal focus of our own research.

In chapter 3 we discuss the motivations and challenges involved in including Virtual Humans (VH) in the NCVE systems. In section 3.1 we provide a brief overview of Virtual Humans as a research domain. Section 3.2 provides the motivation for including VH in NCVE systems. In section 3.3 we outline some of the complexities involved with introduction of VH in NCVE and sketch some guidelines for a software architecture to meet this task. Sections 3.4, 3.5, 3.6 and 3.7 of this chapter deal with problems and requirements for particular tasks involving VH: navigation in the VE, networking, human communication and support for autonomous behaviors, respectively. The final section summarizes the main points brought up in this chapter as basic considerations for our work in this thesis.

So far we have dealt with previous work, motivations, challenges and requirements, outlining only briefly our own ideas about the solutions. The last chapters present our contribution.

Chapter 4 presents the Virtual Life Network (VLNET) system, a flexible framework for Virtual Humans in Networked Collaborative Virtual Environments. After an introduction to VLNET, we deal in separate sections with the VLNET server, VLNET client, navigation support and support for autonomous behaviors in VLNET. The VLNET system is developed in a joint effort between MIRALab, University of Geneva and LIG, EPFL. While the general system architecture is a result of a long term collaboration, development of particular modules was carried out by individuals at either laboratory. In section 4.6 we present the distribution of work in detail, clearly identifying the parts of the system that are part of the work presented in this thesis.

Based on the VLNET system, in chapter 5 we present different means of facial communication we have developed. The four sections of this chapter present video texturing of the face, model based coding of facial expressions, lip movement synthesis from speech and predefined expressions or emotions.

In chapter 6 we present an analysis of the potential relation of the future MPEG-4 standard to the Networked Collaborative Virtual Environments based on the experience from our active participation in ISO/IEC JTC1/SC29/WG11 (MPEG) Ad Hoc Group on Face and Body Animation.

Finally in chapter 7 we conclude by summarizing our contribution, outlining the potential applications and presenting ideas for future work.

2. Networked Collaborative Virtual Environments

Networked Collaborative Virtual Environments (NCVEs) are systems that allow multiple geographically distant users to evolve in a common virtual environment. The users themselves are represented within the environment using a graphical embodiment.

Figure 1 schematically presents the basic principle of the NCVE. Each workstation has a copy of the virtual environment. The user can evolve within the environment and interact with it. All events that have an impact on the environment are transmitted to other sites so that all environments can be updated and kept consistent, giving the impression for the users of being in the same, unique environment. The users become a part of the environment, embodied by a graphical representation that should ideally be human-like.

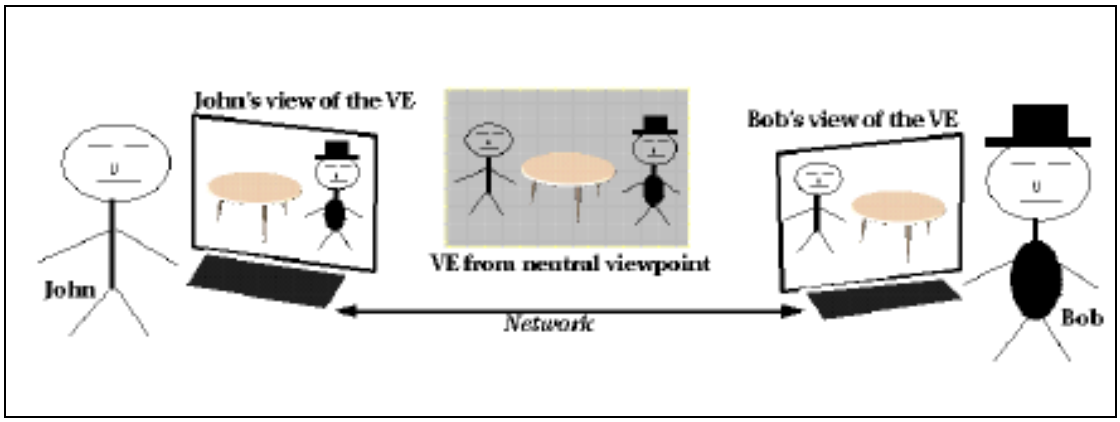


Figure 1: Principles of Networked Collaborative Virtual Environments

In the next sections we will make a survey of main research challenges concerning NCVEs and present several existing NCVE systems.

2.1 NCVE research challenges

NCVE simulation is a research area which presents researchers with a multitude of challenges. *Scaleability* of NCVE to large numbers of users is the holy grail for a lot of researchers in the field. Keeping the environment representation consistent and synchronized for all users requires a careful design *of network topology* to connect the users and the *spatial structure* of the virtual world to divide large scale environments into more manageable partitions. *Real time 3D graphics* is a larger area of research that by no means restricts its usability to NCVEs, but does represent a crucial component of any NCVE system. The *graphical representation of users* in the virtual environment is extremely important because this is how the users perceive each other when using NCVEs, so it has a direct impact on the quality of service offered to the user. Natural *human communication* using speech, facial expressions and gestures is very important to support collaboration, though few systems try to go beyond audio communication. Finally, the degree of *presence* that the user feels in a virtual environment is accepted as one of most important measurements of the quality of simulation. A lot of research has been done on nature, genesis and modification of presence. The notion of presence in NCVEs is extended to mutual presence and awareness, i.e. the feeling of being together with other users.

2.1.1 Scaleability

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

An ideal scaleable 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 user 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 him/her in the virtual environment, and only rudimentary communication with those that are further apart or in other rooms, imitating the real-life behavior.

2.1.2 Network topologies

Figure 2 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 2.

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.

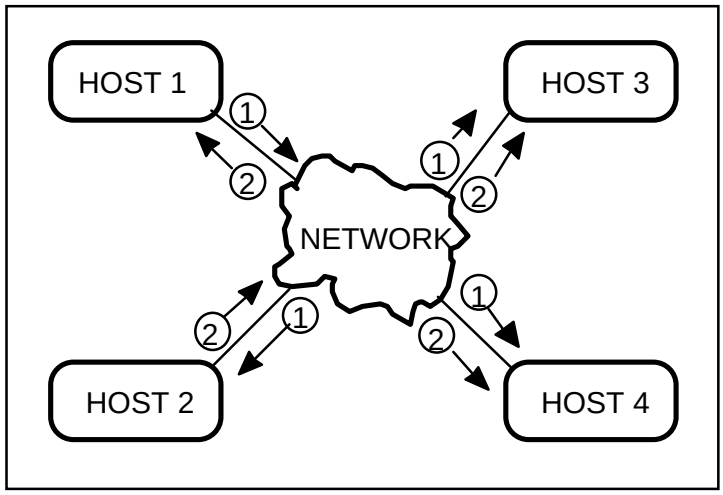


Figure 2: Simplified View of Networking for Collaborative Virtual Environments

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.

In the following subsections we discuss four main network topology solutions for NCVE systems:

- peer-to-peer
- multicast
- client/server
- multiple servers

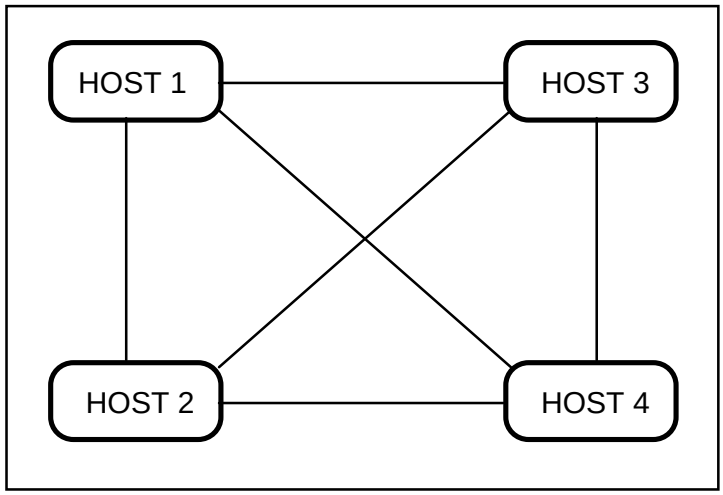


Figure 3: Schematic view of the peer-to-peer network topology

2.1.2.1 Peer-to-peer topology

In the peer-to-peer topology each host needs to send messages directly to all other hosts as shown in Figure 3. This is obviously the most primitive and least practical strategy. Session management is complicated because a new user needs to connect to all hosts already participating in the session, and it is unclear where he/she can get an up-to-date version of the virtual world. Unless a special service process runs somewhere, the world will not persist when there are no more users in a session. AOIM can be implemented on each host but this also poses problems because the AOIM algorithm itself needs up-to-date information about users' positions and if this information is filtered out too much the whole AOIM scheme might collapse.

An example of a system using this topology is the MR Toolkit [Shaw93], which is however not a fully operational NCVE system but rather a toolkit for building VR applications that can possibly be distributed.

2.1.2.2 Multicast topology

Instead of sending messages to all hosts, a message is sent to a multicast address as shown in Figure 4, which allows efficient transfer of the message to all members of the particular multicast group. In this configuration AOIM is done implicitly by joining and leaving multicast groups. This is particularly convenient for geometry-based AOIM strategies, i.e. where groups of users that communicate with each other are determined based on their presence in a certain geometrical space. However, for more complex AOIM strategies this approach might lack flexibility.

As for the session management, inconveniences listed for the peer-to-peer approach persist in this approach.

Also, this approach is practical only on networks that allow multicasting, which might require some configuration work.

Examples of this approach are NPSNET [Macedonia94] and DIVE [Carlsson93].

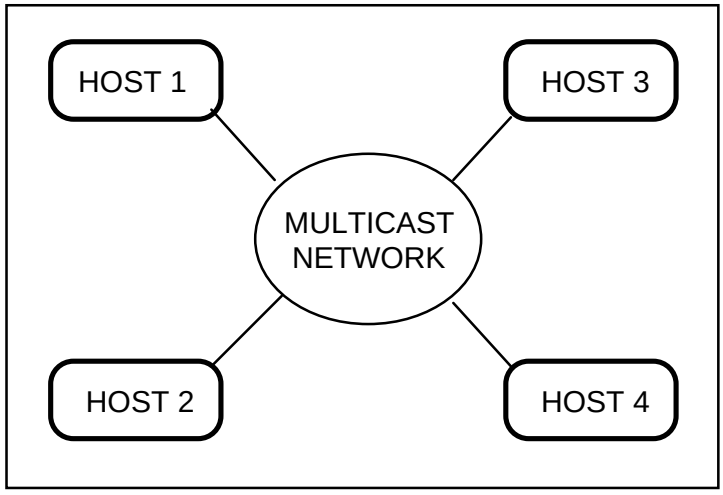


Figure 4: Schematic view of the multicast network topology

2.1.2.3 Client/server topology

Instead of sending messages directly to the hosts, they are sent to a server, and the server proceeds them to the hosts as illustrated in Figure 5. The server holds all the up-to-date information pertaining to the virtual world. It takes care of the session management and keeps the virtual world persistent when no users are connected.

Any AOIM strategy can be implemented in the server, since it holds all relevant information and controls all network traffic.

The inconvenience is that all traffic passes through a single server which will necessarily become congested when the number of users grows.

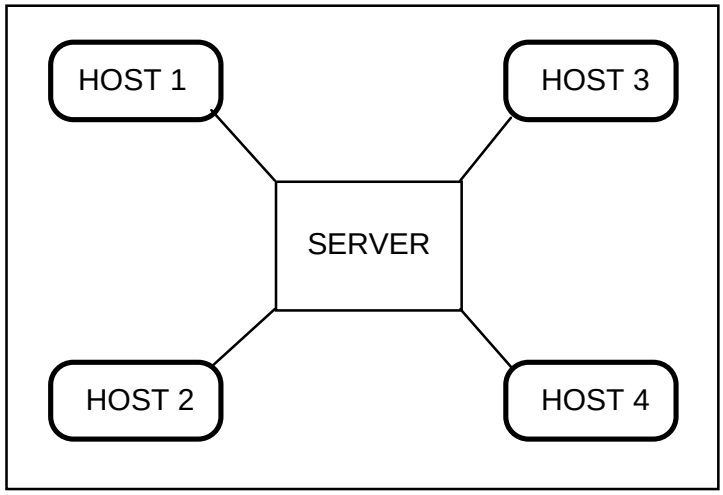


Figure 5: Schematic view of the client/server network topology

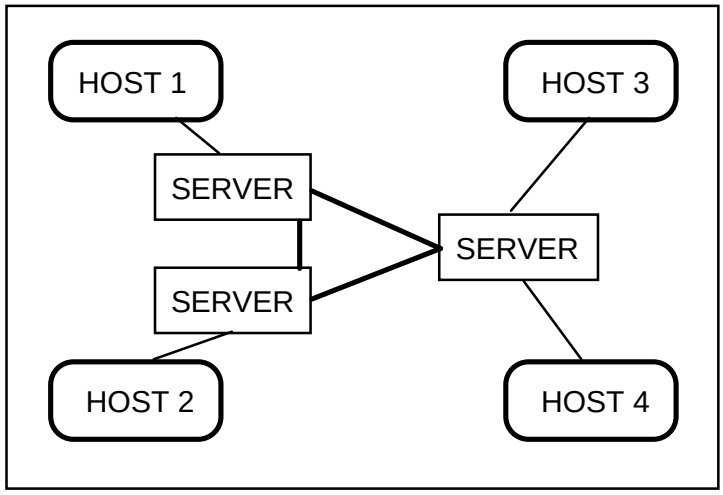


Figure 6: Schematic view of the multiple servers network topology

2.1.2.4 Multiple servers topology

This approach keeps the advantages of the simple client/server approach, but by sharing the network traffic burden between the multiple servers as shown in Figure 6 allows more users to connect to the same world. However, it might introduce extra latency due to longer path of messages.

Also, session management and AOIM become more complex.

Examples of this approach are the BrickNet system [Singh95] and the RING system [Funkhouser95].

2.1.3 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 large numbers of users. It is simply impossible to keep such large structures monolithic, already because of memory and download time problems. Also, the 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. Therefore we will discuss AOIM in parallel with the space structuring.

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].

We present following strategies for space structuring:

- separate servers
- uniform geometrical structure
- free geometrical structure
- user-centered dynamic structure

2.1.3.1 Separate servers

This is the simplest concept of space structuring and resembles the organization of the pages of the World Wide Web. Each world is independent from the rest, but can have links to other worlds just like a Web page has links to other pages as illustrated in Figure 7.

AOIM strategy in this configuration is implicit: the worlds are completely separate and no messages are passed between them.

The advantages of this approach are the relative simplicity of implementation and limitless scaleability. Also, the problem of inaccuracy of large coordinate systems is solved by having separate (and smaller) coordinate system for each world.

However, there are disadvantages. Links between the worlds are only on discrete points - it is impossible to have a continuous boundary. They are unidirectional, so there is no possibility to go back through the same link unless there is a link in the other direction too. Just as with web pages, complex sets are difficult to maintain.

VRML [VRML97], though it is not a real NCVE system, is an example of this strategy.

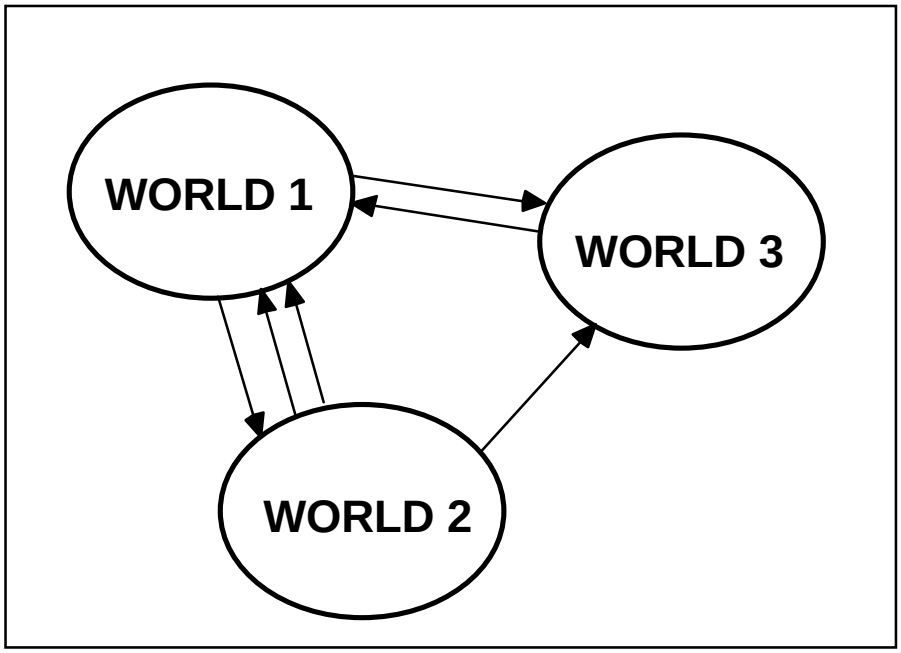


Figure 7: Space structuring with separate servers

2.1.3.2 Uniform geometrical structure

In this approach the world is partitioned in uniformly sized and shaped portions called cells, as illustrated in Figure 8. An example of this approach is NPSNET [Macedonia94] using hexagonal cells.

A user can communicate only with the users in the same cell and in the neighboring cells. Considering the simple uniform cell layout, the cell neighborhood information is easy to maintain as the user moves from cell to cell.

This approach scales well for large number of users, though uniform partitioning of the world might not be flexible enough for all applications. The problem of inaccuracy of large coordinates is not solved because the whole world is in a single huge coordinate system.

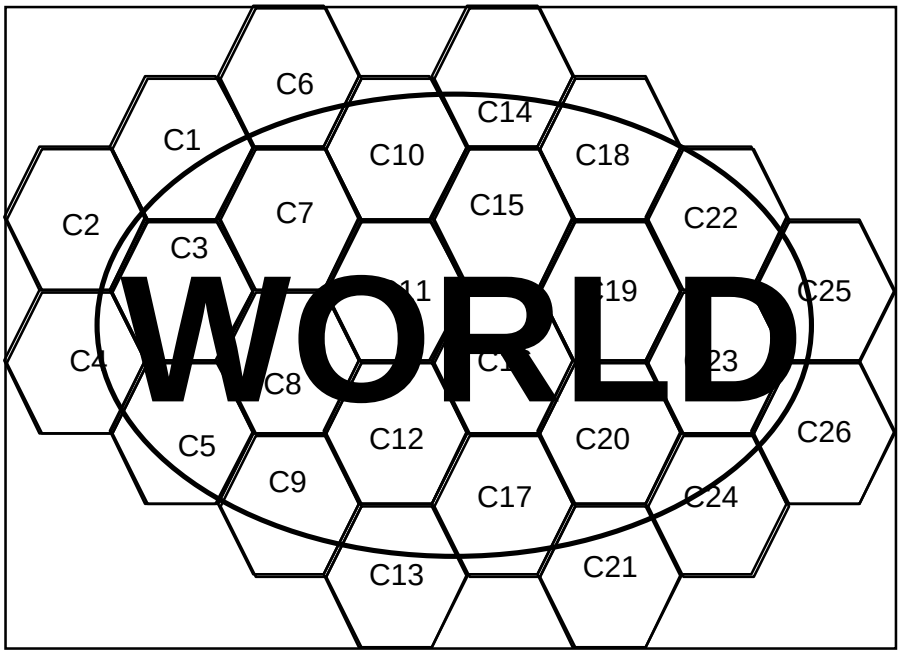


Figure 8: Uniform geometrical space structure

2.1.3.3 Free geometrical structure

Barrus et al. [Barrus96] introduce the concept of locales which inherits good characteristics of the first two approaches. Rather than partitioning the world in uniform cells, the world is composed of sub-worlds called locales (see Figure 9). Unlike the first approach with independent worlds, communication between locales is allowed on a neighborhood basis. Much more flexibility is provided than in the uniform cell approach: shape and size of each locale are arbitrary, neighborhood information is user-defined as well as the transformations between neighboring locales which are defined by transformation matrices. Each local has its own coordinate system, solving the problem of inaccuracy of large worlds. For operations that span more than one locale, transformation matrices between locales are used.

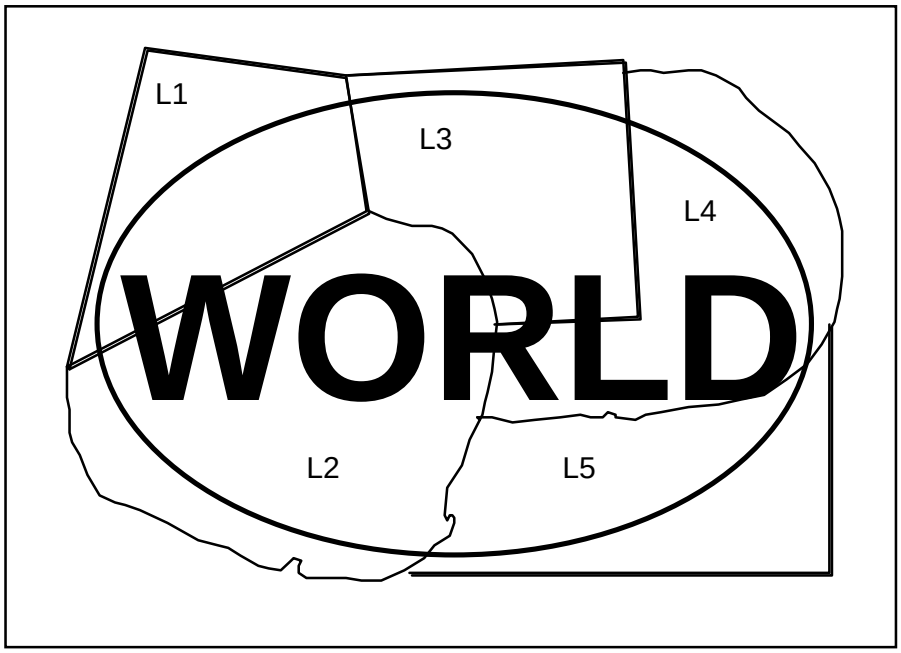


Figure 9: Free geometrical space structure

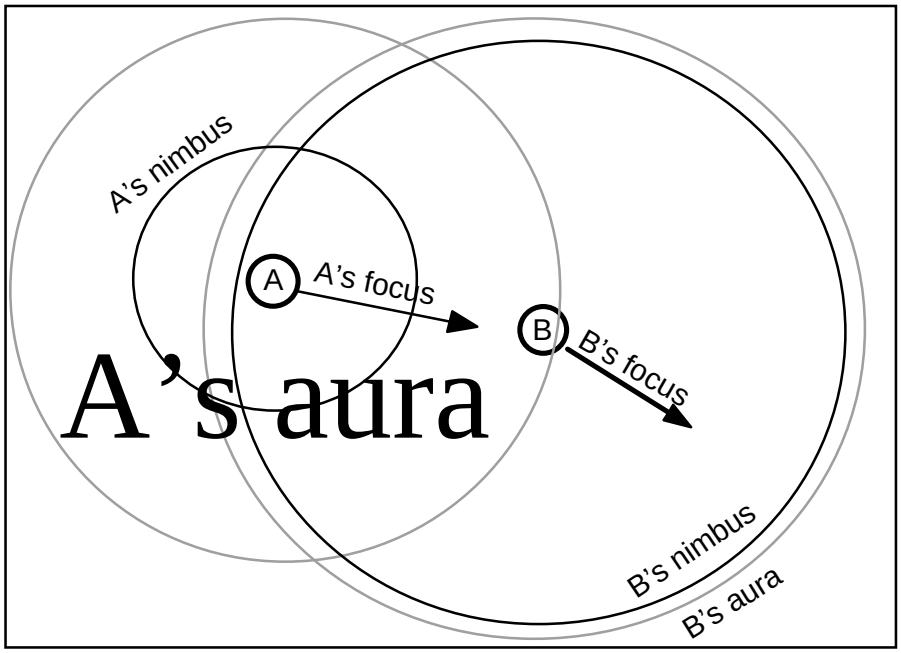


Figure 10: User-centered dynamic space structure - aura, focus and nimbus

2.1.3.4 User-centered dynamic structure

Fahlen, Benford et al. [Fahlen93, Benford95] introduce the notion of aura, focus and nimbus (Figure 10). This concept allows for a fine-grained dynamic management

of space structure and users' (or objects') awareness of each other. This approach does not replace the three approaches mentioned so far, but can be implemented together with any of them.

The concept of aura, focus and nimbus is somewhat more complex than the space structuring approaches introduced so far, therefore we will present it over several following subsections.

2.1.3.4.a) Space and objects

The most fundamental concept in this approach is *space* itself. Space is inhabited by *objects* which might represent people, information or other computer artifacts. Any interaction between objects occurs through some *medium*. A medium might represent a typical communication medium (e.g. audio, vision or text) or perhaps some other kind of object specific interface. Each object might be capable of interfacing through a combination of media/interfaces and objects may negotiate compatible media whenever they meet in space.

2.1.3.4.b) Aura

The first problem in any large-scale environment is determining which objects are capable of interacting with which others at any given time. *Aura* is defined to be a sub-space which effectively bounds the presence of an object within a given medium and which acts as an enabler of potential interaction [Fahlen93]. Objects carry their auras with them when they move through space and when two auras collide, interaction between the objects in the medium becomes a possibility. It is the surrounding environment that monitors for aura collisions between objects. When such collisions occur, the environment takes the necessary steps to put objects in contact with one another (e.g. exchange of object Ids, addresses, references or establishment of associations or connections). Thus, aura acts as a fundamental technological enabler of interaction and is the most elementary way of identifying a subspace associated with an object. An aura can have any shape and size and need not be around the object whose aura it is. Nor need it be contiguous in space. Also, each object will typically possess different auras for different media (e.g. different sizes and shapes). Thus, as I approach you across a space, you may be able to see me before you can hear me because my visual aura is larger than my audio aura.

2.1.3.4.c) Focus, nimbus and awareness

Once aura has been used to determine the potential for object interactions, the objects themselves are subsequently responsible for controlling these interactions. This is achieved on the basis of quantifiable levels of awareness between them. The measure of awareness between two objects need not be mutually symmetrical. A's awareness of B need not equal B's awareness of A. As with aura, awareness levels are medium specific. Awareness between objects in a given medium is manipulated via *focus* and *nimbus*, further subspaces within which an object chooses to direct either its presence or its attention. More specifically, if you are an object in space:

- The more an object is within your focus, the more aware you are of it.
- The more an object is within your nimbus, the more aware it is of you.

The notion of spatial focus as a way of directing attention and hence filtering information is intuitively familiar from our everyday experience (e.g. the concept of visual focus). The notion of nimbus requires a little more explanation. In general terms, a nimbus is a sub-space in which an object makes some aspect of itself available to others. This could be its presence, identity, activity or some combination of these. Nimbus allows objects to try to influence others (i.e. to project themselves or their activity to try to be heard or seen).

Objects negotiate levels of awareness by using their foci and nimbi in order to try to make others more aware of them or to make themselves more aware of others. Awareness levels are calculated from a combination of nimbus and focus. More specifically, given that interaction has first been enabled through aura collision:

The level of awareness that object A has of object B in medium M is a function of A's focus in M and B's nimbus in M.

2.1.3.4.d) Adapters and boundaries

Next, Benford et al. [Benford95] consider how aura, focus and nimbus, and hence awareness, are manipulated by objects in order to manage interactions. They envisage four primary means of manipulation:

- Aura, focus and nimbus may most often be *implicitly* manipulated through spatial actions such as movement and orientation. Thus, as I move, my aura, focus and nimbus might automatically follow me.

- They may on occasion be *explicitly* manipulated through a few key parameters. For example, I might deliberately focus in or out (i.e. change focal length) by simply moving a mouse or joystick.

- They may be manipulated through various *adapter* objects which modify them in some way and which might be represented in terms of natural metaphors such as picking up a tool. Adapters support interaction styles beyond basic mingling. In essence, an adapter is an object which, when picked up, amplifies or attenuates aura, focus and nimbus. For example, a user might conceive of picking up a "microphone". In terms of spatial model, a microphone adapter would then amplify their audio aura and nimbus. As second example, the user might sit at a virtual table. This adapter object would fold their aura, foci and nimbi for several media into a common space with other people already seated at the table, thus allowing a semi-private discussion with a shared space. In effect, the introduction of adapter objects provides for a more extensible model.

- Finally, aura, focus and nimbus may be manipulated through *boundaries* in space. Boundaries divide space into different areas and regions and provide mechanisms for marking territory, controlling movement and for influencing the interactional properties of space. More specifically, boundaries can be thought of as having four kinds of effects: effects on aura, effects on focus, effects on nimbus and effects on traversal (i.e. movement). Furthermore, these effects can be of four sorts: obstructive, non-obstructive, conditionally obstructive and transforming [Bowers92]. These effects are also defined on a per medium basis and different boundaries may mix these effects in different ways. For example, a virtual door might conditionally obstruct traversal, aura, focus and nimbus (the condition being the possession of a key) whereas a virtual window might obstruct traversal but not obstruct aura, focus and nimbus. Of course, there may also be types of boundary which do not have any real-world counterpart like one way mirrors you can walk through.

2.1.4 Real time simulation

Real time 3D graphical simulation is a vast area of research 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.

One of the currently best environments for real time 3D graphics, and the one used in our work, is IRIS Performer [Rohlf94].

IRIS Performer is a toolkit for visual simulation, virtual reality and other real-time 3D graphics applications. The principal design goal is to allow application developers to more easily obtain maximal performance from 3D graphics workstations which feature multiple CPUs and support an immediate-mode rendering library. To this end, the toolkit combines a low-level library for high-performance rendering with a high-level library that implements pipelined, parallel traversals of a hierarchical scene graph Graphics optimizations focus on efficient data transfer to the graphics subsystem, reduction of mode setting, and restricting state inheritance. The toolkit's multiprocessing features solve the problems of how to partition work among multiple processes, how to synchronize these processes, and how to manage data in a pipelined multiprocessing environment. The toolkit also supports intersection detection, fixed-frame rates, run-time profiling and special effects such as geometric morphing.

2.1.5 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 (a more detailed discussion on user representation in NCVEs can be found in Chapter 3).

Although Networked Collaborative Virtual Environments have been around as a topic of research for quite some time, in most of the existing systems the embodiments are fairly simple, ranging 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, as described in subsection 2.2.6.

One of the main purposes of our own work is the introduction of high quality virtual humans for user representation to further enhance the quality of the NCVE experience.

2.1.6 Human communication

NCVE systems are basically communication systems so communication and collaboration between people are their primary purposes. 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 an important role 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 we propose further improvements in the area of facial communication.

2.1.7 Sense of presence

Despite knowledge to the contrary, users of virtual environments often report feeling as if they are actually in the computer-generated world to which they are being exposed. This subjective state is often referred to as *presence* or *being there* [Welch96]. Some investigators, e.g. Steuer, Slater, Usoh [Steuer92, Slater94-1], consider it to be a characteristic of virtual reality that most clearly distinguishes it from other forms of multimedia. Presence is considered to be a desirable outcome for VE participants. There has been much discussion on nature, genesis and modification of presence [Barfield93, Barfield95, Hendrix96, Fontaine92, Heeter92, Sheridan92, Slater94-1, Steuer92, Zeltzer92].

Slater and Usoh [Slater94-1] suggested that the principal determinants of presence were external and internal factors. External factors are those that result from the technology we use to create the virtual environment experience. Such factors include hardware components, peripherals, and the software and models used to create the virtual environment. In contrast, internal factors are those factors that are typically termed subject variables in psychological experiments. These include whether the person has the physical capability for binocular vision, and various psychological

aspects of the virtual environment participant, such as the person's preference for processing sensory data.

In the context of NCVE, the social aspect influences presence. It is expected that the ability to perceive other users and communicate with them increases the sense of presence. Also the quality of this perception and communication has a positive effect on presence. In NCVEs the notion of presence is extended by the presence of other participants to the notion of mutual presence, which is the sense of being together of other people which in reality are on different geographical locations. The graphical user representation and means of communication between the users play an important role for the sense of mutual presence.

2.2 NCVE systems

Networked Collaborative Virtual Environments (NCVE) have been an active area of research for several years now, and a number of working systems exist [Barrus96, Benford95, Carlsson93, Macedonia94, Ohya95, Singh95, Zyda93]. They differ largely in networking solutions, number of users supported, interaction capabilities and application scope, but share the same basic principle illustrated in Figure 1.

While presenting various systems in following subsections, we discuss in particular each system with respect to our main interest area: user representation and human communication.

2.2.1 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.

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). The DIS protocol attempts to provide a basis for communication between

different hardware and software platforms involved in a simulation. DIS is a group of standards being developed by the US Department of Defense and industry that addresses communications architecture, format and content of data, entity information and interaction, simulation management, performance measures, radio communication, emissions, field instrumentation, security, database formats, fidelity, exercise control and feedback. A second purpose is to provide specifications to be used by US government agencies and engineers that build simulation systems.

NPSNET uses uniform geometrical space structuring with hexagonal cells, as presented in subsection 2.1.3.2, combined with multicasting groups to efficiently partition the network traffic for communication between participants.

The user representation in NPSNET was originally the graphical representation of the vehicles they use, e.g. tanks or airplanes. In a later version, simulation of humans is included through integration of University of Pennsylvania Jack human simulation software [Badler93, Pratt97]. This allows for some forms of gestural communication through predefined behaviors like walking, running, crawling etc.. Facial communication is not supported. Figure 11 shows the user representation in NPSNET.

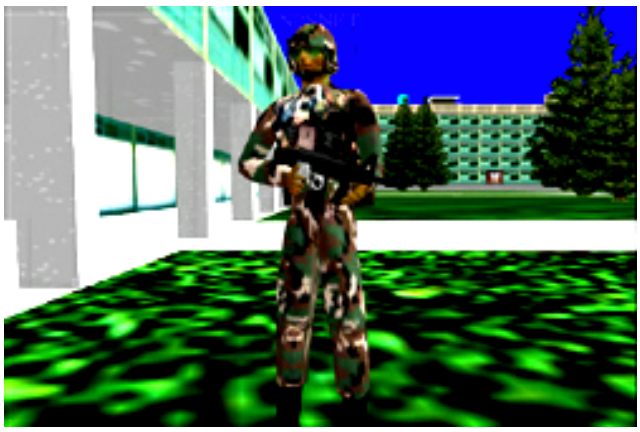


Figure 11: User representation in NPSNET (from NPS Web pages)

2.2.2 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.

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 abstracted descriptions of graphical objects that together constitute a virtual world. Associated with each world is a process group, consisting of all processes that are members of that world. Multicast protocols are used for the communication within such a process group [Birman91].

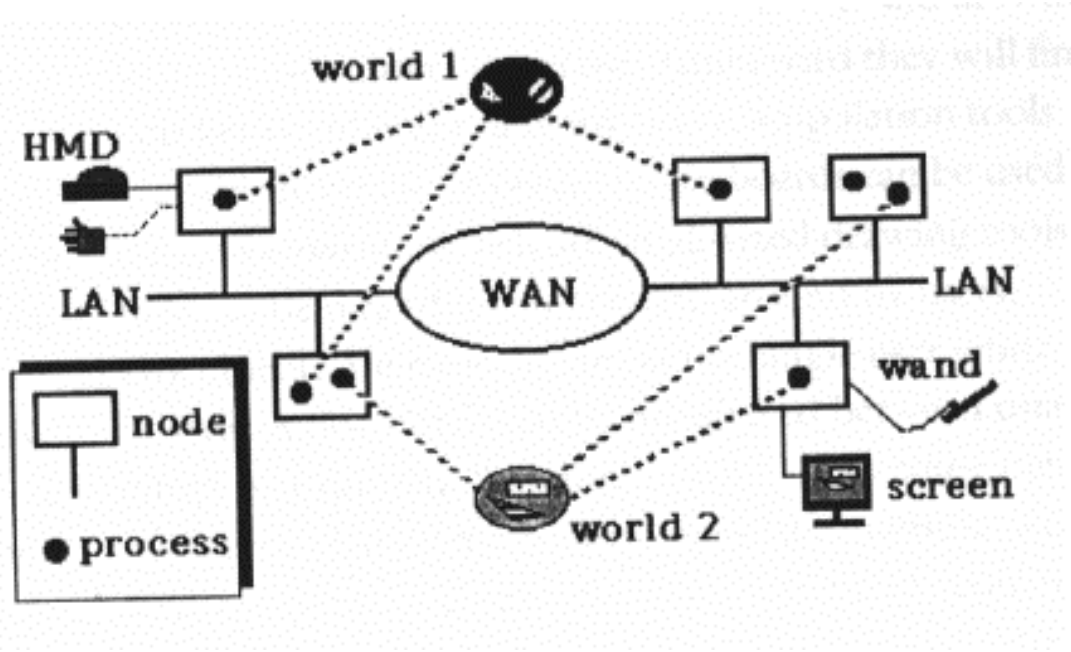


Figure 12: DIVE architecture (from [Benford95])

Figure 12 shows a typical DIVE architecture where several processes, running over a LAN, are members of process groups corresponding to DIVE virtual worlds. These processes are manipulating the world data and presenting it to users through a Head Mounted Display (HMD) or an ordinary computer screen. The LAN's are connected through a WAN.

Each member of a process group has a complete copy of the world database and when a process joins a certain group, it copies the world data from another member of that group. The information in this replicated database is kept consistent by way of distributed locking mechanisms and reliable multicast protocols [Hagsand91]. If there are no other members in the process group than the world state is read from a file.

Also, when the last group member exits a world the state is lost unless it has (explicitly) been saved to a file. A process may enter and leave groups dynamically, but at a given time it will be a member of only one process group. In VR terms this means that an object (e.g. user representation) can freely travel between different worlds, but can only be in one world at a time.

It is possible to distinguish between several different types of processes in the DIVE environment. We will now briefly present the most important ones.

A *user process* is a process that interfaces directly to a human user, and is therefore often responsible for managing the interaction between the user and the virtual environment.

The *visualizer* is a process that is responsible for a large part of what DIVE user encounters in the interface, being the manager of the display devices, different input devices, navigational aids etc. The visualizer supports so called *vehicles*. A vehicle is responsible for the mapping of input devices to actions inside the environment. Several different vehicles have been developed in DIVE, the most sophisticated being the mouse vehicle and the HMD vehicle.

The *auralizer* is a process responsible for generating audio output based on spatial localized sound sources.

Application processes are typically used for the introduction and subsequent management of objects in the environment. Example tasks are collision detection, animated object behavior, interfaces to external database services etc.

DIVE supports a simplified implementation of the dynamic space structuring described in section 0.

For the user representation, DIVE offers several techniques. First, simple "blockies" are composed from a few basic graphics objects (e.g. cubes). Blockies convey presence and location and the use of a line extending from the body to the point of manipulation in space represents the action point. In terms of identity, simple static cartoon-like facial features suggest that a blockie represents a human and the ability of users to tailor their own body images supports some differentiation between individuals (the specification of each individual is stored in a data file and the creation and subsequent management of these embodiments is the responsibility of the DIVE visualizer). A more advanced DIVE body for immersive use texture maps a static

photograph onto the face of the body, thus providing greater support for identity. The body itself is a simple articulated structure composed of rigid pieces for limbs and other body parts. Figure 13 shows one participant's view of a DIVE conference. Two of the participants have human-looking representations, a third is a blockie and an image of another participant is presented on a "virtual video monitor".

DIVE has very basic support for gestural communication by choosing predefined gestures to be reproduced on DIVE's body representation. Facial communication is partly supported through the possibility of streaming video into the scene on a "virtual video monitor". However, the video is not in any way attached to the user, it is just presented on the virtual monitor. Text and audio communication is supported.

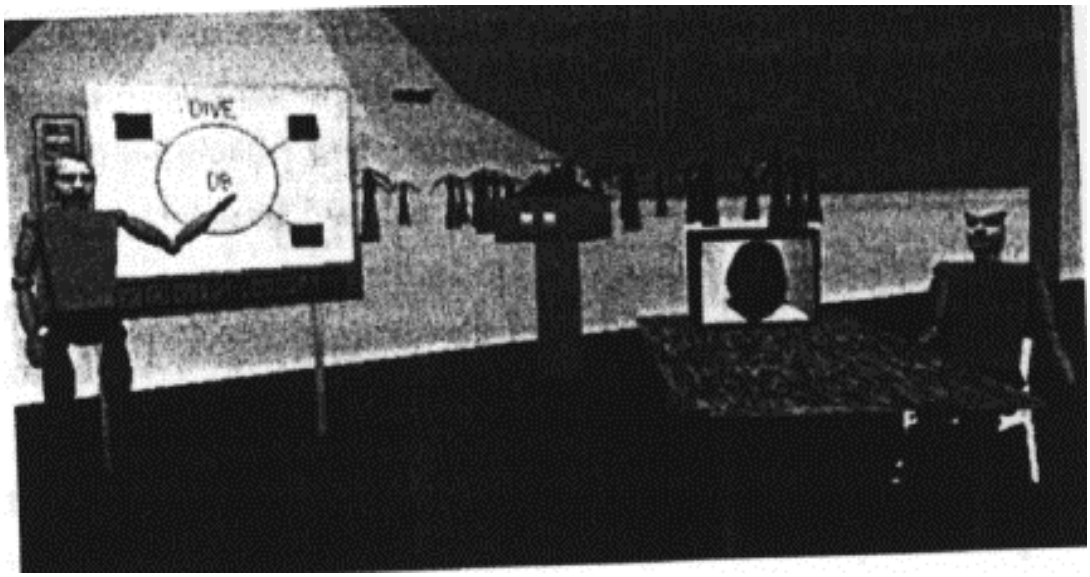


Figure 13: DIVE user representation (from [Benford95])

2.2.3 BrickNet

The BrickNet toolkit 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 as described in subsection 2.1.2.3. A BrickNet server acts as an object request broker and communication server for its clients. 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. The clients are unaware of the locale of the servers serving the client requests. Clients run asynchronously without having to wait for the data to arrive from the server.

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 his/her own virtual world with a set of objects of his/her choice. He/she 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. The authors of the system do not report on the support for text or audio communication. Facial or gestural communication are not supported.

2.2.4 MASSIVE

MASSIVE (Model, Architecture and System for Spatial Interaction in Virtual Environments) [Benford95, Greenhalgh95] is a prototype implementation of the dynamic space structure approach described in section 0. The main goals of MASSIVE are scaleability and heterogeneity, i.e. supporting interaction between users whose equipment has different capabilities and who therefore employ radically different styles of user interface, e.g. users on text terminals interacting with users wearing Head Mounted Displays and magnetic trackers.

MASSIVE supports multiple virtual worlds connected via portals. Each world may be inhabited by many concurrent users who can interact over ad-hoc combinations of graphics, audio and text interfaces. The graphics interface renders objects visible in a 3D space and allows users to navigate this space with six degrees of freedom. The audio interface allows users to hear objects and supports both real-time conversation and playback of preprogrammed sounds. The text interface provides a plan view of the

world via a window (or map) that looks down onto a 2D plane across which users move (similar to Multi-User Dungeons). Text terminal users may interact by typing messages to one another or by invoking emotions (e.g. smile, grimace).

These interfaces may be arbitrarily combined according to the capabilities of a user's terminal equipment. Thus, at one extreme, the user of a sophisticated graphics workstation may simultaneously run graphics, audio and text clients. At the other, the user of a dumb terminal (e.g., a VT-100) may run the text client alone. It is also possible to combine the text and audio clients without the graphics, and so on. This allows users of radically different equipment to interact, albeit in a limited way, within a common virtual environment.

All the above interfaces are driven by the spatial model as described in section 0. Thus, an object cannot be seen until graphics aurae collide and cannot be heard until audio aurae collide. The effects of focus and nimbus are most pronounced in the audio and text interfaces. Audio awareness levels are mapped to the volume with the net effect that audio interaction is sensitive to both the relative distances and orientations of objects involved. Text messages are also displayed according to the mutual level of awareness. In the current implementation, users may explicitly manipulate awareness by choosing between three settings for focus and nimbus - normal (general conversation), narrow (private conversation) and wide (intended for browsing). Two adapter objects are also provided: a podium that extends the aura and nimbus of its user (making him/her more noticeable) and a conference table that replaces the normal aurae, foci and nimbi with the new set that spans the table (invoking a private conversation around the table).

The user representation in MASSIVE is very simple, involving block-like characters shown in Figure 14. Each user may specify his/her or her own graphics embodiment via a configuration file. In addition, some default embodiments are provided that are intended to convey the capabilities of the user. Given MASSIVE's heterogeneity, a major goal of the embodiments is to convey users' capabilities. Thus, considering the graphics interface, and audio-capable user has ears, a desk-top graphics user (monoscopic) has a single eye, an immersed stereoscopic user has two eyes, and a text user has a letter T embossed on his/her head. In the text interface, users are embodied by a single character (usually the first character of their name) that shows position and may help identify the user in a limited way. An additional line (single

character) points in the direction the user is currently facing. Thus, using only two characters, MASIVE attempts to convey presence, location, orientation and identity.

Although it has good support for text and audio communication, MASSIVE does not support facial or gestural communication.

Figure 14 shows a typical view of a graphics user, with several other users in sight.

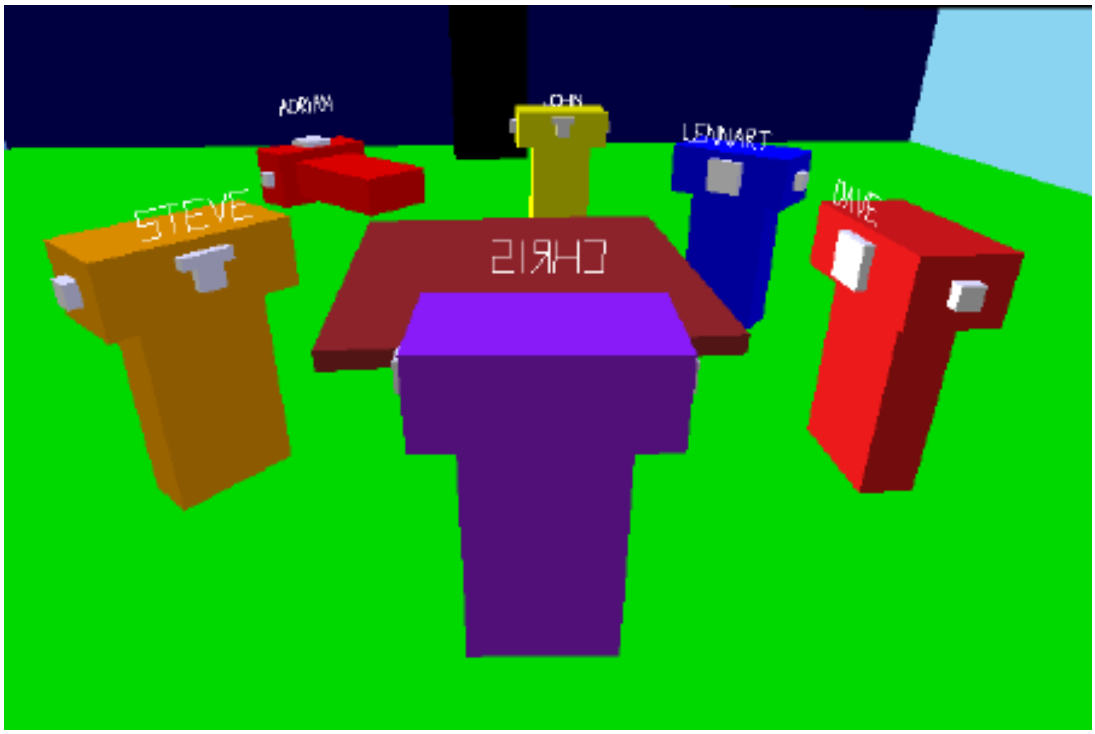


Figure 14 User representation in MASIVE (from [Benford95])

2.2.5 SPLINE

SPLINE, or Scaleable Platform for Interactive Environments [Barrus96] is best known for the introduction of the concept of locales, already discussed in subsection 2.1.3.2. On top of this non-uniform geometrical space structure, multicasting network topology is used.

Another important feature is the support for both pre-recorded and real-time audio. Volume attenuation is used to indicate distance of sound sources and differential attenuation of left and right channels to indicate direction (it is planned to incorporate better audio rendering algorithms to create a more detailed auditory environment).

SPLINE introduces a new synchronization algorithm for synchronizing sound with other events in the environment. Sound synchronization demands a millisecond precision for composing sound samples. On the other hand, virtual environments may persists for days and months. Unfortunately, time stamps with millisecond resolution and such long time span would require a lot of bits making them impractical to use. SPLINE introduces modular time stamps with 1 ms precision and 1 week modulus using Quotient-Normalized Modular Timestamps algorithm [Waters96] to avoid the complexity of the usual modular arithmetic.

SPLINE uses a simple articulated body representation composed of rigid body parts as shown in Figure 15. Audio communication is supported, but there is no support for facial or gestural communication.

Figure 15: User representation in SPLINE

2.2.6 Virtual Space Teleconferencing

Ohya et al. [Ohya95] propose VISTEL- Virtual Space teleconferencing system. As the name indicates, the purpose of this system is to extend teleconferencing functionality into a virtual space where the participants can not only talk to each other and see each other, but collaborate in a 3D environment, sharing 3D objects to enhance their collaboration possibilities.

The current system supports only two users and does not attempt to solve problems of network topology, space structuring or session management, so in a certain sense it is not a complete NCVE system. However, the work of Ohya et al. has similarities with our own, and especially bases itself on similar motives; therefore we give it a special place in this review.

In the VISTEL system most attention is concentrated on reproduction of human motion and facial expressions as means of natural communication in the virtual world.

The human body motion is extracted using a set of magnetic sensors placed on the user's body. Thus the limb movements can be captured and transmitted to the receiving end where they are visualized using an articulated 3D body representation.

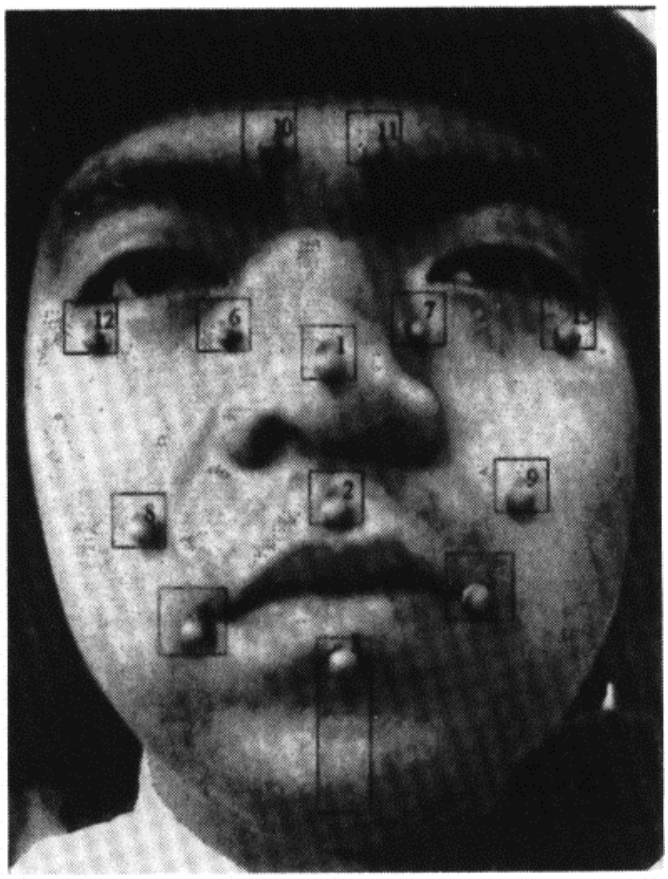


Figure 16: Markers taped on user's face for feature tracking (from [Ohya95])

The facial expressions are captured by tracking facial feature points in the video signal obtained from a camera. The algorithm of Ohya et al. requires colored markers to be placed on the user's face as shown in Figure 16 to facilitate tracking of features.

The movement of facial features is transmitted to the receiving end, where the features of an articulated 3D face model are moved in correspondence.

Figure 17 shows an example session of VISTEL, where a user embodiment can be seen.

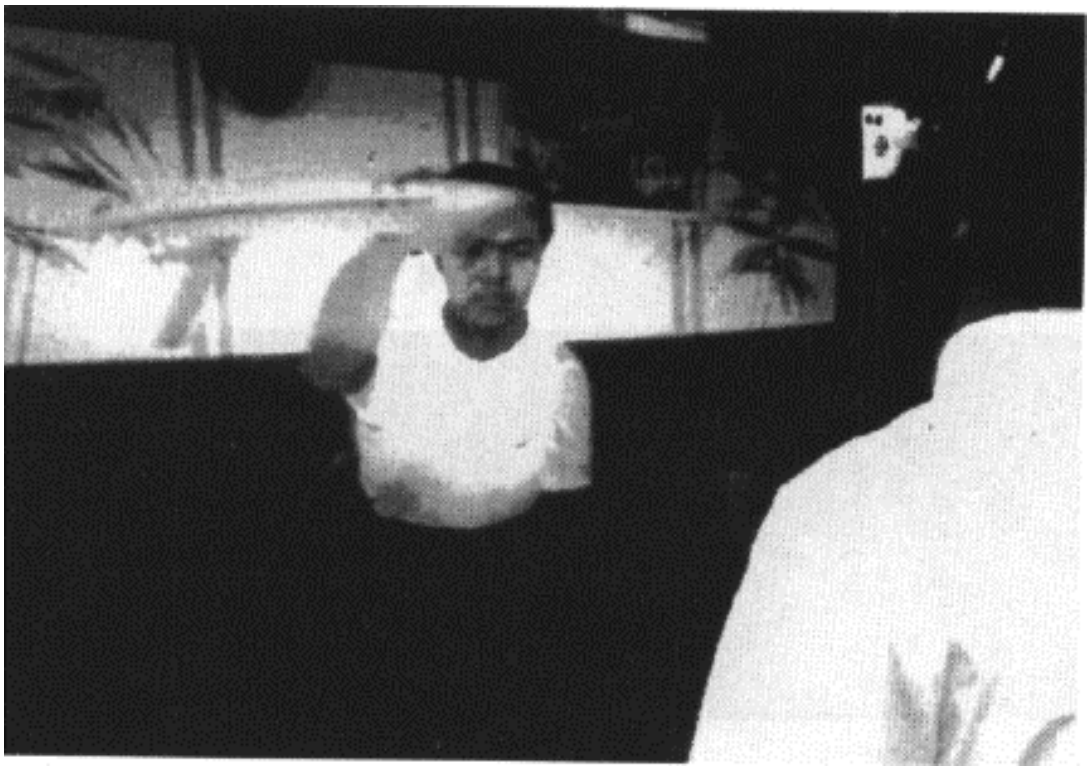


Figure 17: An example session of VISTEL (from [Ohya95])

2.2.7 Concluding remarks

Table 1 presents an overview of the analyzed systems with respect to the research challenges outlined in section 2.1. We have primarily concentrated on issues concerning user representation and human communication. The table shows that most of the systems use a body representation consisting of rigid segments, and some use more primitive representations. It is even more evident that gestural, and in particular facial communication are poorly supported in the current NCVE systems.

N/A = data not availabl	**User repre-sentation**	**Audio comm-unica-tion**	**Text comm-unica-tion**	**Gestural commu-nication**	**Facial commu-nication**	**Network topology**	**Space structu-ring**	**Origin**
NPSNET	Articulated rigid-segment body	Yes	Yes	Predefined gestures/ behaviors	No	Multicast	Uniform	Naval Postgraduate School, Monterey, CA, USA
DIVE	Articulated rigid-segment body	Yes	Yes	Predefined gestures	No	Multicast	Dynamic	Swedish Institute of Computer Science, Stockholm Sweden
BrickNet	None	N/A	N/A	No	No	Multiple servers	Separate servers	University of Singapore
MASSIVE	Simple block-like structure	Yes	Yes	No	No	Multicast	Dynamic	University of Nottingham, UK
SPLINE	Articulated rigid-segment body	Yes	N/A	No	No	Multicast	Free	Mitsubishi Electric Research Labs, Cambridge, MA, USA
VISTEL	Articulated rigid-segment body	Yes	No	User wears magnetic trackers	User wears markers on the face	Peer to peer	None	ATR Research Labs, Kyoto, Japan

Table 1: Comparison of current NCVE systems

3. Introducing Virtual Humans in NCVEs

The goal of this chapter is to establish the importance of integrating Virtual Humans in NCVE systems, analyze the encountered problems and outline the possible solutions as a basis for the work to be developed within this Ph.D. thesis. We give a brief introduction to Virtual Humans (section 3.1), explain why it is important to include them in NCVE systems (section 3.2), analyze the problems involved in such integration and outline the solutions (sections 3.3, 3.4, 3.5, 3.6, 3.7).

In a Virtual Environment (VE) with multiple users it is necessary to represent each participant in the environment using some kind of graphical embodiment. This graphical embodiment has multiple important functions which we discuss in section 3.2, showing that high level Virtual Humans can fulfill those functions most optimally.

However, Virtual Humans involve rather complex algorithms, and integrating them in an already complex NCVE framework requires a careful planing of a software architecture to be used. We analyze these problems and outline possible solutions in section 3.3.

Navigation is a common task in VEs, but in NCVEs involving Virtual Humans the task is largely extended. We analyze this problem in section 3.4.

Furthermore, Virtual Humans integration poses new burdens on the networking in an NCVE system, in particular for personal data exchange and transmission of human-specific data updates (postures, facial expressions). We deal with these problems in section 3.5.

A particularly important function of Virtual Humans in NCVE systems is the support for more natural means of communication through facial expressions and gestures. We analyze the requirements on these types of communication and outline possible solutions in section 3.6.

Section 3.7 analyzes the requirements for the optimal support of autonomous virtual actors in NCVEs.

The final section summarizes the main points brought up in this chapter as basic considerations for our work in this thesis.

3.1 Virtual Humans

The research domain that we call Virtual Humans is concerned with the simulation of human beings on computers. It involves representation, movement and behavior. The range of applications is huge: film and TV productions, ergonomic and usability studies in various industries (aerospace, automobile, machinery, furniture etc.), production line simulations for efficiency studies, clothing industry, games telecommunications (clones for human representation), medicine, etc. These applications have various needs. A medical application might require an exact simulation of certain internal organs; film industry requires highest esthetic standards, natural movements and facial expressions; ergonomic studies require faithful body proportions for a particular population segment and realistic locomotion with constraints; etc.

Sheer complexity of the simulated subject, i.e. the human body, combined with the multitude of applications and requirements, make Virtual Humans a vast research domain comprising numerous research topics:

- Anatomy and geometry, dealing with creation of human shape in 3D graphics, with methods ranging from point-to-point digitizing from clay models [Blum79, Smith83] through various software tools for geometry deformation and modeling [Barr84, Sederberg86, Allan89, MagnenatThalmann89] to laser 3D scanners.
- Hair and skin representation and rendering [Pearce86, Watanabe89, Leblanc90]
- Skeleton animation, or animation of joint angles of the skeleton structure [Badler79] defining the articulated body and consisting of segments (representing limbs) and joints (representing degrees of freedom). Main methods of skeleton animation are parametric keyframe animation [Lee89, Brotman88], direct and inverse kinematics (possibly with constraints) [Badler85, Girard85, Girard87, Forsey88], direct and inverse dynamics [Arnaldi89, Wilhelms87].
- Body surface animation and deformation, trying to simulate natural-looking movement and deformation of visible body surface with respect to the movement of the underlying skeleton structure [MagnenatThalmann87, Chadwick89, Komatsu88, Thalmann96].

- Hand animation and deformation [Badler82, MagnenatThalmann88, Gourret89, Moccozet97].
- Facial animation, playing an essential role for human communication. Two main stream facial animation research exist: parametrized models [Parke82] and muscle models [Waters87, MagnenatThalmann89, Kalra93].
- Clothes simulation [Volino95].
- Walking, i.e. generating natural-looking walking motion based on a given trajectory and velocity [Calvert78, Zeltzer82, MagnenatThalmann85, Boulic95].
- Obstacle avoidance, finding optimal trajectory for walking while avoiding obstacles [Schroeder88, Breen89, Renault90].
- Grasping, i.e. producing appropriate arm and hand motion to reach for and grab an object [Korein82, Moccozet97].
- Behavioral animation, striving to give more character and personality to the animation, thus making it look more natural than mechanics-based animations [Cohen89, Reynolds87, Noser96].

3.2 Reasons for Virtual Humans in NCVE

The participant representation in a networked VE system has several functions:

- perception
- localization
- identification
- visualization of interest focus
- visualization of actions
- communication

Perception and localization are the very basic functions of participant representation in NCVEs. They allow us to perceive the presence of others in the environment and see where they are. Even a crude embodiment can fulfill these tasks.

Identification is an important function because we usually want to know who is in front of us. Means of identification can range from simple ones, like displaying the first letter of one's name [Benford95] to complex body and face models resembling a particular person.

Visualization of interest focus can be achieved by any embodiment that somehow represents the direction of gaze - usually this means a graphical model that has eyes or symbols representing eyes, so we can see in which direction it is looking.

Visualization of actions requires the embodiment to have some end-effectors that perform actions. In a low-end implementation this might be a simple line reaching to the manipulated object, or it might be a virtual hand grasping the object.

Communication in real life is in many ways tied to our body - gestures and facial expressions (including lip movement that improves speech understanding) are natural part of our daily communication. If such communication is to be supported in NCVEs, the participant representation needs to be fairly sophisticated.

Although many of these functions can be fulfilled with very simple embodiments, it is obvious that most can be fulfilled better using more sophisticated Virtual Humans, and some functions can absolutely not be fulfilled without them.

Virtual Humans can fulfill these functions in an intuitive, natural way resembling the way we achieve these tasks in real life. Even with limited sensor information, a virtual human frame can be constructed in the virtual world, reflecting the activities of the real user. Slater and Usoh [Slater94] indicate that such a body, even if crude, already increases the sense of presence that the participants feel. Therefore it is expected that a better and more realistic embodiment will have a further positive effect on the sense of presence and mutual presence (for discussion on presence see subsection 2.1.7).

3.3 Architecture for Virtual Humans in NCVE

Introducing virtual humans in the distributed virtual environment is a complex task combining several fields of expertise [Pandzic97]. The principal components are:

- virtual human simulation, involving real time animation/deformation of bodies and faces and some of the other challenges outlined in section 3.1.
- virtual environment simulation, involving visual data base management and rendering techniques with real time optimizations
- networking, involving communication of various types of data with varying requirements in terms of bitrate, error resilience and latency
- interaction, involving support of different devices and paradigms
- artificial intelligence (in case autonomous virtual humans are involved), involving decision making processes and autonomous behaviors

Each of the involved components represents in itself an area of research and most of them are very complex. When combining them together the interaction between components and their impact on each other have to be considered. For example, using virtual humans sets new requirements on interaction which has to allow not only simple interaction with the environment, but at the same time the visualization of the actions through the body and face representing the user; the necessity to communicate data through the network forces more compact representation of face and body animation data.

Considering the total complexity of the above components, a divide-and-conquer approach is a logical choice. By splitting the complex task into modules, each with a precise function and with well defined interfaces between them, several advantages are achieved:

- high flexibility
- easier software management, especially in a team work environment
- higher performance

- leveraging the power of multiprocessor hardware or distributed environments when available

Flexibility is particularly important, because of the multitude of emerging hardware and software technologies that can potentially be linked with NCVE systems (various input devices and techniques, AI algorithms, real-time data sources driving multi-user applications). This is especially interesting in a research environment where a NCVE system can be used as a testbed for research in fields of AI, psychology, medical information systems etc. In general, a good NCVE system must allow implementation of different applications while transparently performing its basic tasks (networking, user representation, interaction, rendering...) and letting the application programmer concentrate on the application-specific problems.

From the software management point of view, a monolithic system of this complexity would be extremely difficult to manage, in particular by a team of programmers.

By carefully assigning tasks to processes and synchronizing them intelligently, higher performance can be achieved [Rohlf94].

Finally, a multi-process system will naturally harness the power of multi-processor hardware if it is available. It is also possible to distribute modules on several hosts.

Once it is decided to split the system into modules, roughly corresponding to the above listed components, it is necessary to define in detail the task of each module and the means of communication between them.

3.4 Navigation with Virtual Humans

Basic navigation involves using some input device to control walk-through or fly-through motion. In the context of NCVEs, this notion is vastly extended, especially when they involve human-like embodiments for the users [Pandzic97-1]. In such context, navigation involves (at least) the following problems:

- walking or flying
- basic object manipulation
- mapping of actions on embodiments
- general input device support
- implementing constraints

Walking and flying represent navigation in its basic sense, allowing the user to explore the environment from any point of view.

Basic object manipulation capabilities allow the user to pick up and displace objects in the scene. This may be extended by object behaviors that can make objects react in some other way to being picked up.

Mapping of actions on the embodiment is important for two reasons. First, it allows the local user to see what he/she is doing, e.g. by seeing his/her hand grab an object. Second, in a multi-user session it allows users to intuitively understand what the others are doing. Mapping of actions on the embodiment involves generation of walking motion while moving, as well as generation of natural arm motion while manipulating objects.

General device support means that it should be straightforward to connect any device to the system. This implies general solutions that will accommodate different kinds of devices, e.g. incremental devices like SpaceBall vs. absolute devices like magnetic trackers, devices that generate events like a button generating a "grab" event vs. devices generating states like a data glove generating a "grab" state while the fist is tightened.

Constraints are an extremely important component of any navigation. They avoid user getting lost, turning upside-down or coming into all sorts of impossible situations. They allow to tailor the navigation paradigm in a precise manner. We divide them in two groups:

- global motion constraints
- body posture constraints

The global motion constraints involve some global knowledge of the virtual world (e.g. up direction) and/or collision detection. They determine if the user can walk or fly, where it is possible to go, what are the possible orientations. A typical set of constraints for walking might include an inclination constraint keeping the user upright, a vertical collision constraint keeping him/her on the floor (and at the same time making it easy to climb/descend stairs or ramps) and a horizontal collision constraint keeping the user from going through the walls.

Body posture constraints keep the user's embodiment in natural-looking postures, e.g. the head can't wander from its position on the shoulders, the arm can reach only that far.

3.5 Networking for Virtual Humans

Virtual Humans put some additional burdens on the networking strategy of a NCVE system. First, there is a need for each participant to provide his/her personal data to all other participants in the session. Second, the movement/behavior of the user needs to be transmitted through the network.

The transmission of personal data typically happens when a new user joins the NCVE session. The personal data minimally includes some kind of identity (e.g. name) and appearance data. The appearance information might be simply a choice between a number of predefined appearances provided by the system. However, it is desirable for a user to be able to fully customize his/her appearance in the virtual world: size and shape of the body, face, textures etc. Depending on the way the system handles user actions, personal data might include behavior information, e.g. definition of personalized gestures or facial expressions. When the user joins a NCVE session, his/her personal data needs to be distributed to all users already in the session, or at least to those that are currently in the vicinity of the newcomer. Obviously, the newcomer must receive the personal data from other participants.

The transmission of user movement and/or behavior happens all the time during the session as the users move, gesture or emote through the facial expressions. A system using primitive user representation without Virtual Humans needs to update only user positions. When Virtual Humans are used, body posture and facial expression need to be updated.

One way of achieving this is through implicit or explicit behaviors. Implicit behavior means that the body postures are determined from the global motion, e.g. if the user is standing the global motion is interpreted as walking and appropriate walking motion is generated; if the user is lying on the ground, global motion is interpreted as crawling and crawling motion is generated [Pratt97]. In this strategy no additional data has to be transmitted as the postures are generated locally at each site. For explicit behaviors, high level behavior data is transmitted (e.g. jump, wave, smile) and this data is interpreted locally to provide actions (i.e. jump is translated into a jumping motion of the body, smile into appropriate facial expression).

The use of implicit and explicit behaviors provides means to control the behavior on a high level without the need to send a lot of data through the network. However, this approach lacks flexibility because it is constrained to the behaviors already defined in the system. To allow any kind of free movement and facial expression, body postures and facial expressions need to be transmitted through the network in a more general form.

For the body, the usual representation of postures is in terms of joint angles, where each degree of freedom of the body is represented by an angle. Typically, Virtual Humans are modeled with less degrees of freedom then the real human body.

The facial expressions need to be transmitted in terms of basic facial movements (e.g. raise left eyebrow, stretch lips etc.), allowing the reproduction of any facial expression as a combination of the basic movements.

3.6 Facial and gestural communication

In section 2.1.5 we have stressed the importance of facial and gestural communication in NCVEs. Here we outline the technical challenges involved in supporting them, concentrating more on facial communication.

The problem of communication in a networked environment can be split in three parts: data acquisition, transmission and reproduction. To illustrate this on a simple example, in a telephone conversation the data acquisition is done by a microphone, audio signal is transmitted through the network and reproduced by a loudspeaker in the handset on the other side.

For both facial and gestural communication in a NCVE system the reproduction is done by means of a Virtual Human model, where body and face can interpret postures gestures and facial expressions.

The transmission involves expressing gestures and expressions in a form that can be transmitted digitally, as we have already discussed in section 3.5.

The hardest problem is in fact data acquisition: how does the user input gestures and expressions? Ideally, this would be done by performing them in the natural way without any constraints. The problem is how to capture the movement. The "obvious" solution is to use one or more cameras to capture the user and analyze the video frames to extract the movement. However, this is quite a difficult computer vision task, especially for the tracking of body postures. The results can be greatly improved by using colored markers on the body/face. However, this method can not be classified as non-intrusive. It is unrealistic to expect the users to paint their faces in order to use a NCVE system.

For the acquisition of gestures, it is quite common to use magnetic trackers placed on the body. They can track body motion in real time. However, current tracker models are connected by wires and are cumbersome to put on and wear.

A simple solution for data acquisition is letting the user choose from a set of predefined gestures/postures/facial expressions. This approach matches well with explicit

behavior transmission as outlined in section 3.5, although the movements can also be generated locally and transmitted.

When talking about facial and gestural communication in a networked setting, it is difficult to avoid comparison with common video conferencing systems. Obviously, as people can see each other, facial expressions and gestures are perceived naturally. However, video conferencing lacks the capabilities of the NCVE systems to simulate a 3D environment and allow users to interact with 3D objects as well as with each other. Also, if more than two users participate in a video conferencing session, each user sees a number of windows on his/her screen, each window showing one participant, making the spatial relationship between participants confusing. On the other hand, video does provide a very good solution for acquisition/transmission/reproduction of facial expressions. Therefore, for the situations where additional bandwidth required for video can be afforded, it is worth considering a hybrid approach. The hybrid approach might transmit the facial expressions in form of video and integrate the video in the 3D environment instead of showing it simply in a window. In this approach the video may be placed in a logical place in the virtual environment (i.e. on the face of the Virtual Human representation), keeping the spatial relationship between the users.

3.7 Autonomous Virtual Humans

Introducing seemingly autonomous virtual beings into virtual environments to co-habit and collaborate with us is a continuous challenge and source of interest. Latest proof of human excitement for virtual life is the current world-wide craze for electronic pets that must be fed and cared for lest they develop a bad character or die. Even more interesting is the inclusion of autonomous actors in NCVEs. They provide a meeting place for people from different geographical locations and virtual beings. In NCVEs we do not see our correspondents, only their graphical representations in the virtual world, same as for the virtual ones - therefore the communication with virtual beings can come naturally. These computer-controlled, seemingly autonomous creatures would inhabit the virtual worlds, make them more interesting, help users to find their way or perform a particular task. For example, a virtual shop might have an autonomous virtual shopkeeper to help the customer to find the needed wares; complex virtual environments might have guides to lead visitors and answer questions; in a game, an opponent or a referee might be an autonomous virtual actor.

Simulation of Autonomous Behavior (AB) is a big research topic. There are different approaches and different implementations. Examples include the work of [Zeltzer82] on task level animation, [Reynolds87] on behavioral group animation, [Blumberg95] on autonomous creatures for interactive virtual environments, [Badler93] and [MagnenatThalmann93-1] on autonomous humanoids and [Noser96] on behavioral L-systems. Systems that implement such behaviors are typically rather complex. As NCVE systems are already very complex themselves, our approach to Autonomous Behaviors in NCVEs is interfacing the two systems externally rather than trying to integrate them completely in a single, large system. Such an approach also facilitates the development of various AB systems to be used with a single NCVE system, making it a testbed for various algorithms.

This interfacing leads to a kind of symbiosis between an NCVE system and an AB system, where the AB system provides the brains and the NCVE system the body to move around, be seen and act upon objects, but also to see, hear and get the external information to the brain (see illustration in Figure 18). In order to implement this

strategy, the NCVE system must provide an external interface to which AB systems can be hooked. This interface should be as simple as possible to allow easy connection of various autonomous behaviors. At the same time it should satisfy the basic requirements for a successful symbiosis of a NCVE system and an AB system: allow the AB system to control its embodiment and act upon the environment, as well as gather information from the environment upon which to act.

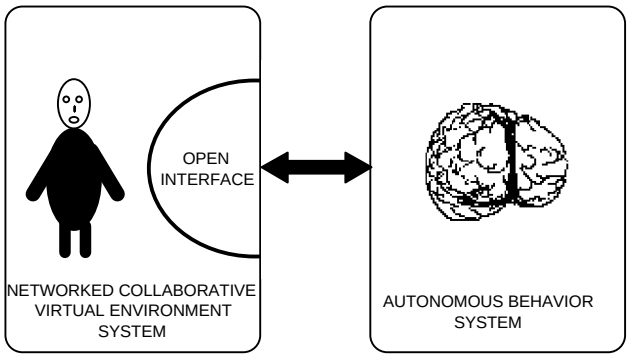


Figure 18: Symbiosis between the AB and NCVE systems

We study the functionalities that the NCVE system must provide to the AB system through the open interface. We have identified following important functionalities that must be provided for a successful symbiosis:

- embodiment
- locomotion
- capacity to act upon objects
- feedback from the environment
- verbal communication
- facial communication
- gestural communication

Embodiment, or a graphical representation, is a fundamental requirement to allow presence of the virtual actor in the environment. Though this can be a very simple

graphical representation (e.g. a textured cube), some of the more advanced functionalities (e.g. facial and gestural communication) require a more human-like structure as support.

Locomotion is necessary for getting from one place to another, and might involve generation of walking motion or simple sliding around. Essentially, the AB system must be able to control the position of the embodiment in the environment.

Capacity to act upon objects in the environment is important because it allows the AB system to interact with the environment. The interface should provide at least the possibility to grab and move objects.

Without some feedback from the environment our virtual actors might be autonomous, but blind and deaf. It is essential for them to be able to gather information from the environment about objects and other users.

If the AB system is to simulate virtual humans, it is necessary for real humans to be able to communicate with them through our most common communication channel - the verbal communication. Because the audio signal might in most cases be meaningless to the AB system, ASCII text seems to be the most convenient way to interface verbal communication to and from the AB system. This does not exclude the possibility of a speech recognition/synthesis interface on the human end, allowing us to actually talk to the virtual actors.

Finally, it is desirable for a virtual actor to have the capacity of facial and gestural communication and therefore the ability to have a more natural behavior by showing emotions on the face or passing messages through gestures.

3.8 Concluding remarks

In this section we summarize the main points brought up in this chapter, which are also the main considerations for the development of our own NCVE system.

The research domain of Virtual Humans deals with simulation of humans on a computer, involving their graphical representation, movement and behavior. Virtual Humans are of great importance for NCVE systems, providing the means for perception, localization and identification of users, as well as visualization of their interest focus and actions. Another important function of Virtual Humans is to allow natural communication through gestures and facial expressions.

However, the inclusion of Virtual Humans in NCVE systems introduces twofold complexities: the inherent complexity of the Virtual Human simulation and the additional complexity imposed on the other parts of the system by the inclusion of Virtual Humans. A modular system architecture with intelligent communication between modules is recommended in order to manage the total complexity of the system.

Navigation in the virtual environment is particularly influenced by the introduction of Virtual Humans. The user's movements and interactions have to be mapped on the motion of the user's representation. At the same time, constraints on global motion, as well as body movement have to be respected.

New demands on networking emerge because of the need to exchange personal data (identity, appearance, possibly behaviors) between the users participating in a session, and transmit updates of users' postures and facial expressions through the network.

This is particularly important for facial and gestural communication. For these types of communication the most difficult problem is the acquisition of data (expressions, gestures) for the user in a practical and preferably non-intrusive way. Several possibilities exist using computer vision, magnetic tracking, simple keyboard/mouse input or some hybrid approach involving video communication. These approaches have to be analyzed.

Introduction of Autonomous behaviors (AB) for Virtual Humans is a topic deserving special attention. Existence of various strategies for AB simulation leads to a symbiosis approach where the AB system acts as the brain and the NCVE system as the body, with suitable interfaces allowing easy connection of the AB system to the NCVE system.

4. Virtual Life Network

Based on the considerations from the previous chapter we have developed a flexible framework for the integration of virtual humans in the Networked Collaborative Virtual Environments. It is based on a modular architecture that allows flexible representation and control of the virtual humans, whether they are controlled by a physical user using all sorts of tracking and other devices, or by an intelligent control program turning them into autonomous actors. The modularity of the system allows for fairly easy extensions and integration with new techniques making it interesting also as a testbed for various domains from "classic" VR to psychological experiments.

In the approach we adopted when developing the Virtual Life Network (VLNET) system [Capin97, Pandzic97, Pandzic97-2], sophisticated virtual humans are simulated, including full anatomically based body articulation, skin deformations and facial animation [Boulic95]. Managing multiple instances of such complex representations and the involved data flow in the context of a Networked Collaborative Virtual Environment, while maintaining versatile input and interaction options, requires careful consideration of the software architecture to use. In view of the complexity of the task and versatility we wanted to achieve, a highly modular, multiprocess architecture was a logical choice. Some of the modules are fixed as part of the system core, while the others are replaceable external processes allowing greater flexibility in controlling the events in the environment, in particular the movement and facial expressions of the virtual humans.

VLNET is based on a client/server network topology as described in section 4.1. VLNET server takes care of the session management, message distribution and AOIM. We describe the server in the following section 4.1. Section 4.2 deals with the VLNET client. Most of the system functions - Virtual Humans management, visual data base management, rendering, networking, navigation, object manipulation - are concentrated in the client. It provides interfaces for external applications or drivers that extend VLNET functionality. In section 4.3 we explain how VLNET deals with navigation. Finally, section 4.4 discusses how support for autonomous actors is handled in VLNET.

4.1 VLNET Server

A VLNET server site consists of a HTTP server and a VLNET Connection Server. They can serve several worlds, which can be either VLNET world description files or VRML 1.0 files. VLNET world description file is a metafile containing pointers to files describing graphical objects which can be in various formats (VRML, Inventor, Alias/Wavefront, 3D Studio etc.). The VLNET file manages a scene hierarchy tree consisting of these graphical objects. Behaviors, sounds, light emitting properties or links to other worlds can be attached to objects.

For each world, a World Server is spawned as necessary, i.e. when a client requests a connection to that particular world. The life of a World Server ends when all clients are disconnected.

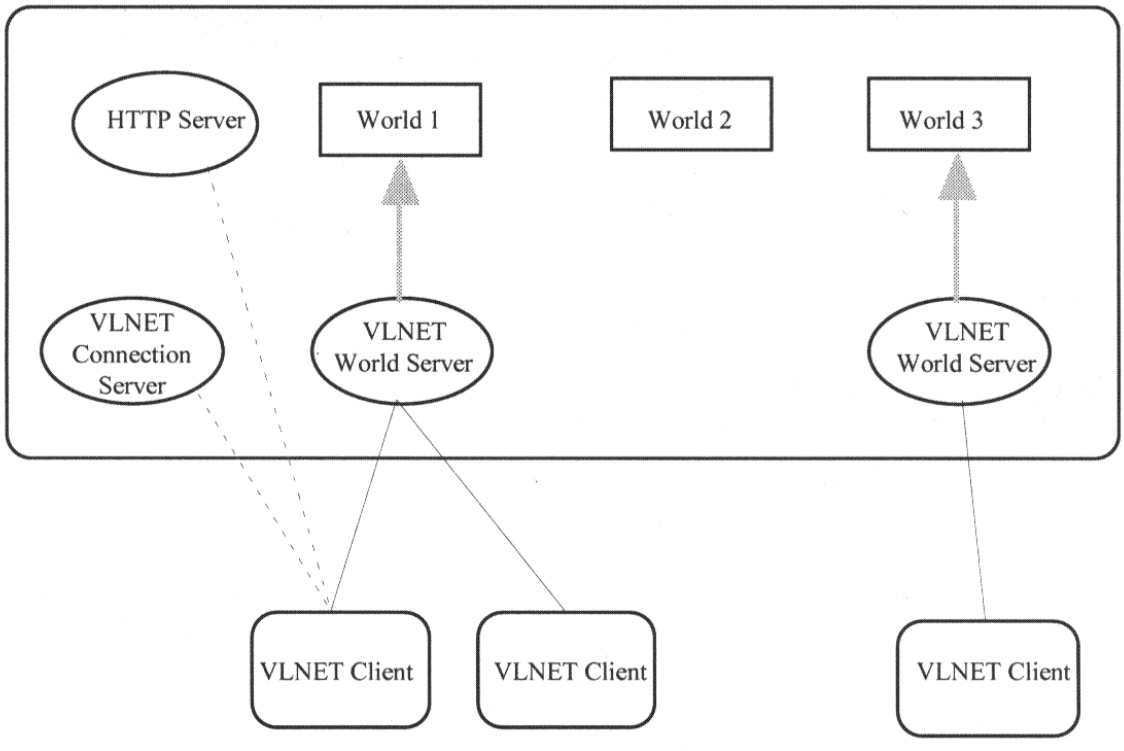


Figure 19: Connection of several clients to a VLNET server site

Figure 19 schematically depicts a VLNET server site with several connected clients. A VLNET session is initiated by a Client connecting to a particular world

designated by a URL. The Client first fetches the world database from the HTTP server using the URL. After that it extracts the host name from the URL and connects to the VLNET Connection Server on the same host. The Connection Server spawns the World Server for the requested world if one is not already running and sends to the Client the port address of the World Server. Once the connection is established, all communication between the clients in a particular world passes through the World Server. The user provides his/her personal data by distributing a URL from which all participants can fetch the data.

In order to reduce the total network load, the World Server performs the filtering of messages (AOIM) by checking the users' viewing frusta in the virtual world and distributing messages only on as-needed basis.

During a session, the World Server keeps the world data up to date with any changes happening in the world, so new users can get the up to date version of the world. It can save the state of the world in a file at the end of the session, keeping the world persistent in between sessions.

4.2 VLNET Client

The design of the VLNET Client is highly modular, with functionalities split into a number of processes. Figure 20 presents an overview of the modules and their connections. VLNET has an open architecture, with a set of interfaces allowing a user with some programming knowledge to access the system core and extend the system by plugging custom-made modules into the VLNET interfaces. In the next subsections we explain in some detail the VLNET Core with its various processes, as well as the interfaces and the possibilities for system extension they offer.

4.2.1 VLNET Core

The VLNET core is a set of processes, interconnected through shared memory, that perform basic VLNET functions. The Main Process performs higher level tasks, like object manipulation, navigation, body representation, while the other processes provide services for networking (Communication Process), database loading and maintenance (Database Process) and rendering (Cull Process and Draw Process).

4.2.1.1 The Main Process

The Main Process consists of seven logical entities, called engines, covering different aspects of VLNET. It also initializes the session and spawns all other internal and external processes. Each engine is equipped with an interface for the connection of external processes.

4.2.1.1.a) Object Behavior Engine

The Object Behavior Engine takes care of the predefined object behaviors, like rotation or falling, and has an interface enabling the external processes to control object behaviors.

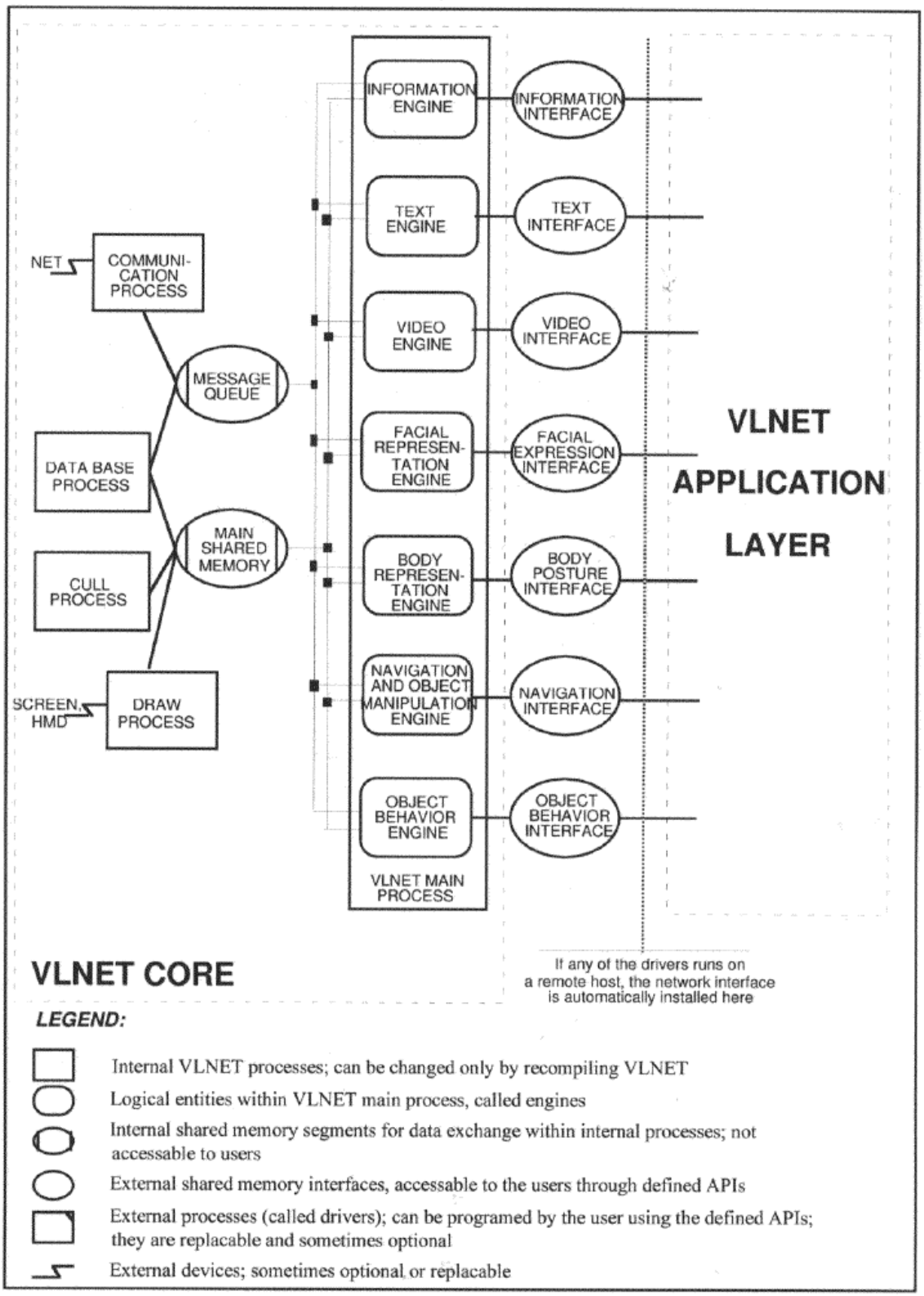


Figure 20: Virtual Life Network system overview

4.2.1.1.b) Navigation and Object Manipulation Engine

The Navigation and Object Manipulation Engine takes care of the basic user input: navigation, picking and displacement of objects. It provides basic mouse-based navigation as well as an interface that allows external processes to control navigation and extend the system with support of different devices and/or navigation paradigms. We discuss navigation in more detail in section 4.3.

4.2.1.1.c) Body Representation Engine

The Body Representation Engine is responsible for the deformation of the body. In any given body posture (defined by a set of joint angles) this engine will provide a deformed body ready to be rendered. The body representation is based on the Humanoid body model [Boulic95], adapted for real time operation [Thalmann96]. This engine provides the interface for changing the body posture.

4.2.1.1.d) Facial Representation Engine

The Facial Representation Engine provides the synthetic faces with a possibility to change expressions or the facial texture. The Facial Expression Interface is used for this task. It can be used to animate the face using a set of parameters defining the facial expression. These parameters are listed in Table 2. The facial representation is a polygon mesh model with Free Form Deformations simulating muscle actions [Kalra92]. Each animation parameter is interpreted as a Free Form Deformation of a particular region of the facial polygon mesh. This level of representation of expressions/animation is very general because the set of parameters is based on a study of muscle actions, and the parameters are therefore sufficient to express any facial expression. At the same time, with 63 parameters the representation is compact which is important for communication in a networked environment.

For changing/animating the facial texture, the Video Engine is used to receive and decompress the texture images, and the Facial Representation Engine maps the texture on the face

4.2.1.1.e) Video Engine

The Video Engine manages the streaming of dynamic textures to the objects in the environment and their correct mapping. Its interface provides the possibility to stream video textures on any object(s) in the environment. The engine compresses the outgoing video images and decompresses the incoming images, then maps the images on the objects in the environment. In the special case of mapping the texture on a face, the images are passed to the Facial Representation Engine as explained in the previous subsection. SGI Compression Library is used for compression and decompression.

4.2.1.1.f) Text Engine

The Text Engine does not make any visual changes in the Virtual Environment, but just serves to pass text messages between the users, providing the text interface. More details about the text interface are provided in subsection 4.2.2.1, and about different usage of text in VLNET in section 4.4 on autonomous actors in VLNET.

4.2.1.1.g) Information Engine

The Information Engine does not make any visual changes in the Virtual Environment, but just serves to pass information about the virtual environment to the external applications through the information interface, as described in detail in subsection 4.2.2.1.

All the engines in the VLNET core process are coupled to the main shared memory and to the message queue. They use the data from the culling process in order to perform the "computation culling". This means that operations are performed only for user embodiments and other objects when they are within the field of view, e.g. there is no need to perform facial expressions if the face is not visible at the moment. Substantial speedup is achieved using this technique.

4.2.1.2 Cull and Draw Processes

Cull and Draw processes access the main shared memory and perform the functions of culling and drawing as their names suggest. These processes are standard SGI Performer [Rohlf94] processes.

4.2.1.3 The Communication Process

The Communication Process handles all network connections and communication in VLNET. All other processes and engines are connected with it through the Message Queue. They fill the queue with outgoing messages for the Communication Process to send. The messages are sent to the server, distributed and received by Communication Processes of other clients. The Communication Process puts these incoming messages into the Message Queue from where the other processes and engines can read them and react. All messages in VLNET use the standard message packet. The packet has a standard header determining the sender and the message type, and the message body. The message body content depends on the message type but is always of the same size (80 bytes), satisfying all message types in VLNET.

For certain types of messages (positions, body postures) a dead-reckoning algorithm is implemented within the Communication Process [Capin97-1]. The dead-reckoning technique is a way to decrease the amount of messages communicated among the participants, and is used for simple non-articulated objects in popular systems such as DIS [IEEE93], NPSNET [Macedonia94].

To describe the dead-reckoning algorithm, similar to [Gossweiler94], we can give an example of space dogfight game with n players. Each player is represented by, and can control, a different ship. When a player X moves its own ship, it sends a message to all n-1 players, containing the new position. When all players move once, a total of n*(n-1) messages are communicated. To reduce the communication overhead, the player X sends the ship's position and velocity to other participants. The other participants will use the velocity information to extrapolate the next position of the participant X. This extrapolation operation is named dead-reckoning.

In this approach, each participant also stores another copy of its own model, called ghost model, to which it applies dead-reckoning algorithm. If the difference between the real position and this additional copy is greater than a predefined maximum, then player X sends the real position and velocity to other participants, so that they can correct their copy of participant X's object. Note that player X sends messages only if there is a big difference between the real position and the extrapolated one.

The performance of the dead-reckoning algorithm is dependent on how it correctly predicts the next frames. Therefore, the characteristics of the simulation, and the underlying object should be taken into account for developing the algorithm. The dead-reckoning technique for non-articulated rigid objects is straightforward. Message transmission occurs if the Euclidean distance between the object and its ghost is greater than a threshold distance, or the 3D angle between the object and its ghost is greater than a threshold degree. The extrapolation computation is also straightforward: the ghost object is transformed (i.e. translated and rotated) using the current translational and rotational speed.

The video data from the Video Engine is a special case and is handled using a separate communication channel. It is given lower priority than the other data.

By isolating the communications in this separate process, and by keeping the VLNET Server relatively simple, we leave the possibility to switch easily to a completely different network topology, e.g. multicasting instead of client/server.

4.2.1.4 The Data Base Process

The Data Base Process takes care of the off-line fetching and loading of objects and user representations. By keeping these time consuming operations in the separate process non-blocking operation of the system is assured.

4.2.2 The Application Layer

VLNET interfaces provide the simple and flexible means to access and control all the functionalities of VLNET internal processes. Each engine in VLNET provides a simple interface to its particular functions (for description of engines see subsection 4.2.1.1), and it is possible to connect one or more external processes to all needed interfaces depending on the wanted functions.

The interfaces are implemented as shared memory segments, each with a simple API (see subsection 4.2.2.2) allowing access to all functions of the engine. A transparent network connection is automatically installed by VLNET if an external process connects to VLNET interfaces from a remote host. This allows distributed computing within a single VLNET client.

The configuration of external processes and their connection to interfaces is done directly in the VLNET command line as explained in subsection 4.2.2.3.

From the functional point of view, two types of processes may be connected to VLNET: drivers and applications.

Drivers are small and simple processes that usually connect to only one interface and fulfill a well defined and isolated function. Examples include navigation drivers that are used to support different navigation devices and paradigms, drivers generating walking motion, etc. Many drivers are already provided as accessories to VLNET and these can be readily used and combined with user-developed drivers and applications.

Application programs of any kind and complexity can be connected to VLNET and use it as a motor to give them a 3D networked front-end. This makes development of various specific applications using VLNET straightforward.

4.2.2.1 Interfacetypes

The Facial Expression Interface is used to control expressions of the user's face. The expressions are defined using the Minimal Perceptible Actions (MPAs) [Kalra93]. The MPAs provide a complete set of basic facial actions, and using them it is possible to define any facial expression. The list of MPAs is provided in Table 2.

The API of the Facial Expression Interface allows to set the MPAs and activate the expression.

MPA name	MPA description	Bi-directional / Unidirectional	Direction of movement for positive intensities*
"move_h_l_eyeball"	Horizontal movement of the left eyeball	B	Right
"move_h_r_eyeball"	Horizontal movement of the right eyeball	B	Right
"move_h_eyeballs"	Horizontal movement of both eyeballs	B	Right
"move_v_l_eyeball"	Vertical movement of the left eyeball	B	Downward
"move_v_r_eyeball"	Vertical movement of the right eyeball	B	Downward
"move_v_eyeballs"	Vertical movement of both eyeballs	B	Downward

"close_upper_l_eyelid"	Vertical movement of the upper left eyelid	B	Downward
"close_upper_r_eyelid"	Vertical movement of the upper right eyelid	B	Downward
"close_upper_eyelids"	Vertical movement of both upper eyelids	B	Downward
"close_lower_l_eyelid"	Vertical movement of the lower left eyelid	B	Upward
"close_lower_r_eyelid"	Vertical movement of the lower right eyelid	B	Upward
"close_lower_eyelids"	Vertical movement of both lower eyelids	B	Upward
"raise_l_eyebrow"	Vertical movement of the left eyebrow	B	Upward
"raise_r_eyebrow"	Vertical movement of the right eyebrow	B	Upward
"raise_eyebrows"	Vertical movement of both eyebrows	B	Upward
"raise_l_eyebrow_l"	Vertical movement of left part of left eyebrow	B	Upward
"raise_r_eyebrow_l"	Vertical movement of left part of right eyebrow	B	Upward
"raise_eyebrows_l"	Vertical movement of left part of both eyebrows	B	Upward
"raise_l_eyebrow_m"	Vertical movement of middle part of left eyebrow	B	Upward
"raise_r_eyebrow_m"	Vert. movement of middle part of right eyebrow	B	Upward
"raise_eyebrows_m"	Vert. movement of middle part of both eyebrows	B	Upward
"raise_l_eyebrow_r"	Vertical movement of right part of left eyebrow	B	Upward
"raise_r_eyebrow_r"	Vertical movement of right part of right eyebrow	B	Upward
"raise_eyebrows_r"	Vertical movement of right part of both eyebrows	B	Upward
"squeeze_l_eyebrow"	Horizontal movement of the left eyebrow	U	Toward face center
"squeeze_r_eyebrow"	Horizontal movement of the right eyebrow	U	Left
"squeeze_eyebrows"	Horizontal movement of both eyebrows	U	Right
"move_o_l_eyeball"	Outside/inside movement of the left eyeball	B	Forward
"move_o_r_eyeball"	Outside/inside movement of the right eyeball	B	Forward
"move_o_eyeballs"	Outside/inside movement of both eyeballs	B	Forward
"open_jaw"	Vertical movement of the jaw	U	Downward
"move_hori_jaw"	Horizontal movement of the jaw	B	Right
"depress_chin"	Upward and compressing movement of the chin (like in sadness)	U	Upward

"raise_l_cornerlip"	Vertical movement of the left corner of the lips	U	Upward
"raise_r_cornerlip"	Vertical movement the right corner of the lips	U	Upward
"raise_cornerlips"	Vertical movement the corners of the lips	U	Upward
"puff_l_cheek"	Puffing movement of the left cheek	B	Left
"puff_r_cheek"	Puffing movement of the right cheek	B	Right
"puff_cheeks"	Puffing movement of both cheeks	B	Towards face edges
"lift_l_cheek"	Lifting movement of the left cheek	U	Upward
"lift_r_cheek"	Lifting movement of the right cheek	U	Upward
"lift_cheeks"	Lifting movement of both cheeks	U	Upward
"lower_l_cornerlip"	Vertical movement of the left corner of the lips	U	Downward
"lower_r_cornerlip"	Vertical movement of the right corner of the lips	U	Downward
"lower_cornerlips"	Vertical movement of the corners of the lips	U	Downward
"raise_upperlip"	Vertical movement of the upper lip	U	Upward
"lower_lowerlip"	Vertical movement of the lower lip	U	Downward
"raise_u_midlip"	Vertical movement of middle part of the upper lip	U	Upward
"raise_l_midlip"	Vertical movement of middle part of the lower lip	U	Upward
"raise_midlips"	Vertical movement of the middle part of the lips	U	Upward
"pull_midlips"	Protruding movement of the mouth (like when producing "ou" sound)	U	Forward
"stretch_cornerlips"	Stretch the corners of the lips	U	Toward face edges
"stretch_l_cornerlip"	Stretch the left corner of the lips	U	Left
"stretch_r_cornerlip"	Stretch the right corner of the lips	U	Right
"suck_lips"	Inward movement of the lips (like when producing the "m" sound)	U	Backward
"squeeze_cornerlips"	Squeeze the corners of the lips	U	Toward face center
"squeeze_l_cornerlip"	Squeeze the left corner of the lips	U	Right
"squeeze_r_cornerlip"	Squeeze the right corner of the lips	U	Left
"stretch_nose"	Stretch/squeeze movement of the nose	B	Toward face edges

"raise_nose"	Vertical movement of the nose	U	Upward
"turn_head"	Turning movement of the head	B	Right
"nod_head"	Nodding movement of the head	B	Down
"roll_head"	Rolling movement of the head	B	Clockwise
* The directions are expressed with respect to the gaze direction of the face			

Table 2: Minimal Perceptible Actions

The Body Posture Interface controls the motion of the user's body. The postures are defined using a set of joint angles corresponding to 72 degrees of freedom of the skeleton model used in VLNET. An obvious example of using this interface is direct motion control using magnetic trackers [Molet96]. A more complex body posture driver is connected to the interface to control body motion in a general case when trackers are not used. This driver connects also to the Navigation Interface and uses the navigation trajectory to generate the walking motion and arm motion. It also imposes constraints on navigation, e.g. not allowing the hand to move further than arm length or take an unnatural posture.

The API for the Body Posture Interface allows to set the joint angles of the body and activate the deformation of the body into the given posture.

The Navigation Interface is used for navigation, hand movement, head movement, basic object manipulation and basic system control. The basic manipulation includes picking objects up, carrying them and letting them go, as well as grouping and ungrouping of objects. The system control provides access to some system functions that are usually accessed by keystrokes, e.g. changing drawing modes, toggling texturing, displaying statistics. Typical examples of using this interface are a SpaceBall driver, tracker+glove driver, extended mouse driver (with GUI console). There is also an experimental facial navigation driver letting the user navigate using his/her head movements and facial expressions tracked by a camera [Pandzic94]. If no navigation driver is connected to the navigation interface, internal mouse navigation is activated within the Navigation Engine.

The API for the Navigation Interface allows to change the global position, viewpoint and hand position matrices, as well as to request picking up/letting go of objects and to activate keystroke-commands. Navigation in VLNET is explained in more detail in section 4.3.

The Object Behavior Interface is used for controlling the behavior of objects. Currently it is limited to controlling motion and scaling. Examples include the control of a ball in a tennis game and the control of graphical representation of stock values in a virtual stock exchange.

The API for the Object Behavior Interface allows to set and get transformation matrices of objects in the environment.

The Video Interface is used to stream video texture (but possibly also static textures) onto any object in the environment. Alpha channel can be used for blending and achieving effects of mixing real and virtual objects/persons. This interface can also be used to stream facial video on the user's face representation for facial communication [Pandzic96-1]. This will be presented in more detail in chapter 5.

The API for the Video Interface allows to set images to be mapped on any object in the environment. The images are passed as simple RGB-format pictures in shared memory and the object to map the image on is designated by an object ID. Changing images in time produces video effect.

The Text Interface is used to send and receive text messages to and from other users. An inquiry can be made through the text interface to check if there are any messages, and the messages can be read. The interface gives the ID of the sender for each received message. A message sent through the text interface is passed to all other users in a VLNET session.

The API for the text interface allows to set a message to be sent, check if there are any incoming messages, read them and find out who is the sender.

The Information Interface is used by external applications to gather information about the environment from VLNET. Because of its particular importance for implementation of autonomous actors and other complex external applications we will

present this interface in somewhat more detail. It provides high-level information while isolating the external application from the VLNET implementation details. It offers two ways of obtaining information, namely the request-and-reply mechanism and the event mechanism.

In the request-and-reply mechanism, a request is described and submitted to the VLNET system. Then, the request will be processed by the information interface engine in the VLNET system and a reply will be generated.

In the event mechanism, an event registration is described and submitted to the VLNET system. The event registration will be processed by the information interface engine and be stored in an event register. At each rendering frame, the VLNET system will process the event register and generate events accordingly. These event registrations will remain registered and be processed in each rendering frame until a removal from the event register is requested.

Figure 21 illustrates the information flow between the program controlling the autonomous actors and the VLNET system.

There are two message queues linking the program and the VLNET system. One message queue is used to submit requests, event registrations and event removals to the VLNET system. The other message queue is used to obtain replies and events from the VLNET system.

Within the VLNET system, the information engine is responsible for processing the requests, event registrations and event removals. The event registrations and event removals are sent to the event register, so that the event register can be updated accordingly. After processing the requests by the information interface engine, replies and events are generated and placed in the outgoing message queue from the VLNET system.

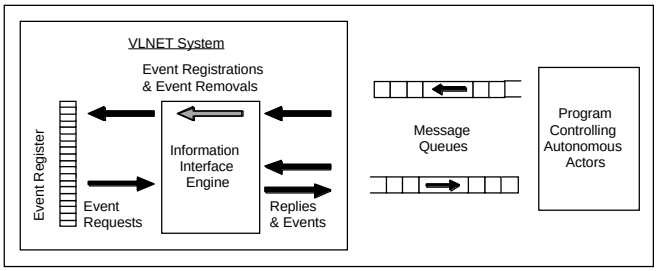


Figure 21: Data flow through the Information Interface

Following information can be requested from VLNET through the Information Interface API:

- Description (name) of an object in the virtual world, given the object ID
- List of objects (object IDs and corresponding descriptions) whose description contains the given keywords
- List of users (user IDs and corresponding descriptions) whose description contains the given keywords
- Transformation matrix of an object in world coordinates, given the object ID
- Transformation matrix of a user in world coordinates, given the user ID
- Number of objects in the virtual world that are picked up by a user, given the user ID
- Number of users in the virtual world
- Collisions between users/objects

4.2.2.2 APIs for external processes

For each type of interface there is an API that lets the external process connect to VLNET. These APIs are very simple. All of them include function calls to open and close the shared memory interface, to update the data (e.g. a position matrix for the navigation driver, a facial expression parameter set for the face driver) and to activate the data, i.e. to give the signal to VLNET that the data is updated and action needed. Only

the Information Interface API is a little more complex, as explained in the previous subsection, due to the need for a more general data exchange.

Here is an example of a valid facial expression driver that connects to the Facial Expression Interface and rhythmically opens and closes the mouth.

```
#include "vlnet_face.h"
void main(int argc, char *argv[])
  {
     float exp[MAX_MPAS];
     int shmKey = atoi(argv[2]) /* Get the shared memory key
                                        from the arguments */
     vrf_init(shmKey);          /* Connect to the interface */
     while(1)
       {
           if(exp[MPA_OPEN_JAW] >= 1.0) /* Control the
              exp[MPA_OPEN_JAW] = 0.0;     mouth opening */
           else
              exp[MPA_OPEN_JAW] += 0.05;
           vrf_set_expression(exp); /* Set the expression */
           vrf_active(); /* Activate */
       }
  }
```

4.2.2.3 Configuring the application layer

A simple, yet complete command line convention allows to specify any configuration of drivers and applications running on local or remote hosts. The external process to be spawned is defined by giving the name of the executable, and the interface(s) to which it should be connected, indicated by the option letter: F for face, B for body, N for navigation and O for object behaviors, V for video, T for text, I for information interface. The external process can be spawned on a remote host by specifying the executable in the form `executable@host`. Here are some command line examples:

`vlnet ... -F faced -N spaceballd` : spawns the `faced` driver for the face control and "spaceballd" driver for navigation

`vlnet ... -F faced@some.host` : spawns the `faced` driver at host some.host and connects it to the facial expression interface

`vlnet ... -NFB my_app` : spawns the `my_app` application and connects it to the face, body and navigation engines to control the complete body behavior

When spawning an external process, VLNET creates the shared memory interface(s) and passes the shared memory key(s) to the process. The process will receive as arguments the shared memory key for each engine interface it has to connect to. So, the "faced" driver in the first example would receive the arguments `-F <shmkey>`, -F meaning it has to connect to the facial expression interface and <shmkey> being the shared memory key to use for that connection. The `my_app` application in the last example has to connect to three interfaces so its arguments would be `-N <shmkey1> -F <shmkey2> -B <shmkey3>`.

In the second example a driver is spawned on a remote host. In this case the network interface processes are spawned on the local host and on the remote host. These processes are transparent to VLNET and to the driver. VLNET connects to the network interface process as it would to the driver, and the driver connects to the network interface process as it would connect to VLNET. The two network interfaces processes transfer the data between the two hosts.

It is possible to spawn an external process manually by specifying "man" in the command line instead of the driver name. In this case VLNET will create the interface for the driver and ask the user to start the driver manually, specifying the interface and shared memory key that should be passed to the driver.

4.3 Navigation in VLNET

In this section we describe in more detail how navigation works in VLNET, based on the requirements on navigation outlined in section 3.4. We isolate from the big picture of the VLNET client (Figure 20) the parts of VLNET involved in navigation in order to analyze the solutions offered in VLNET to the problems posed in the section 3.4. Figure 22 shows the modules involved with navigation, indicating their functions and the logical data flow between them.

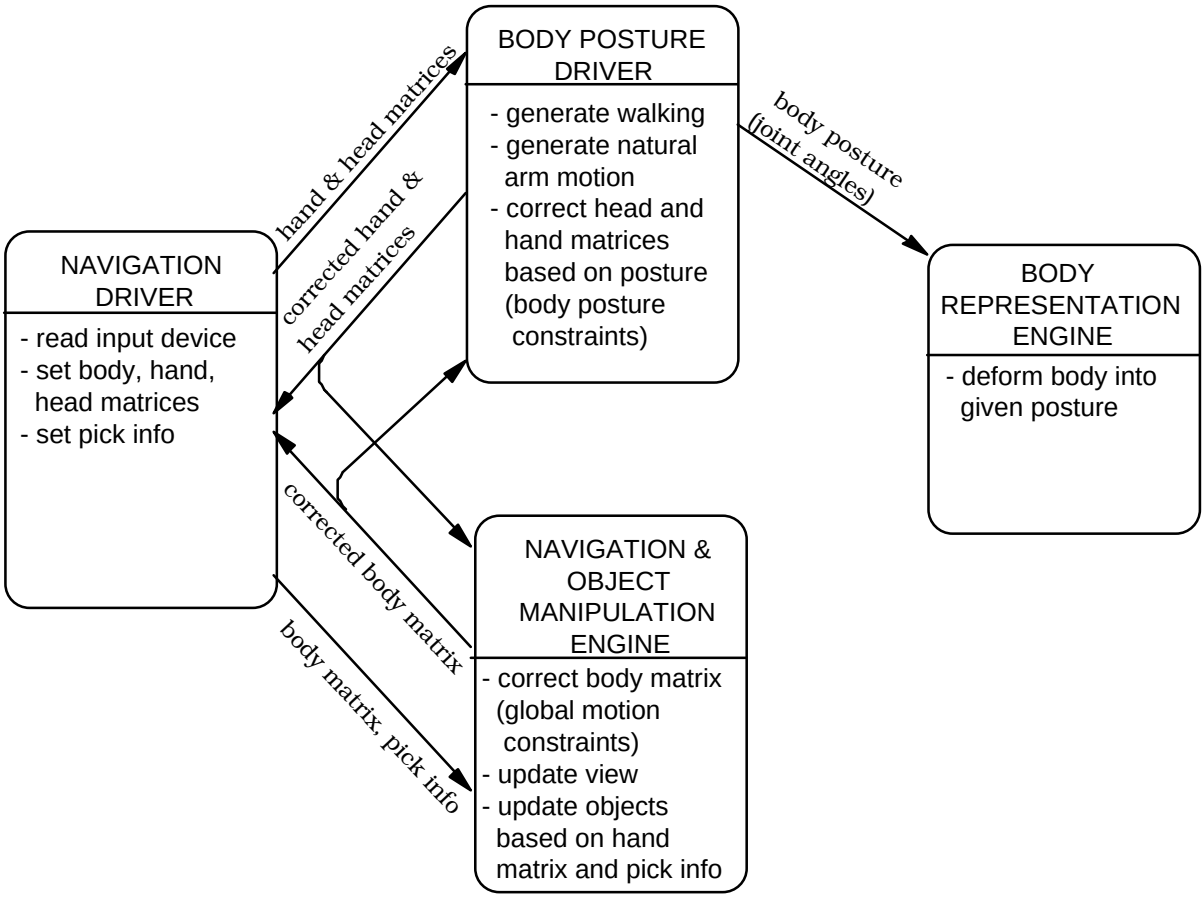


Figure 22: VLNET modules involved in navigation and corresponding data flow

It can be observed how the problems involved with navigation are split between modules of VLNET. The device support is handled by the Navigation Driver. The Body Posture Driver handles the mapping of actions on the embodiment and the body posture constraints. The basic navigation and object manipulation, as well as global motion constraints, are handled by the Navigation and Object Manipulation Engine.

This specialization of modules results in higher flexibility and efficiency.

4.3.1 The navigation data

The data involved in navigation includes the following:

• the body matrix

• the hand matrix

• the head matrix

• pick information

The body matrix determines the global position of the user's body origin in world coordinates. The hand matrix is relative to the body origin and defines the position of the end effector used for grabbing objects, usually the hand. The head matrix is also defined with respect to the body origin and determines the position and orientation of the user's head, i.e. the user's view into the world. The pick information contains the pick and unpick flags used to control the grabbing of objects.

4.3.2 The roles of modules

The basic role of the Navigation Driver is to support a particular input device and navigation paradigm. New input devices/paradigms can be added by programming new drivers - a simple interface API is provided for that purpose.

The general function of the Body Posture Driver in VLNET is to determine the body postures and pass them to the Body Representation Engine in order to put the body in the correct posture. The body postures are generated by walking and arm motors, generating appropriate motions [Boulic90, Pandzic96]. In the context of navigation, the function of this driver is to implement body posture constraints. It can be replaced by a user designed driver but within this thesis we will base the discussion on the standard driver provided in VLNET.

The Navigation and Object Manipulation Engine is the part of VLNET Core responsible for updating the view based on the incoming data, and for implementing basic object manipulation. It also implements the global motion constraints.

4.3.3 The data flow

The Navigation Driver reads from the input device and sets the matrices and pick information accordingly. In case of an incremental device, it uses the feedback of constraint-corrected matrices from the previous frame in order to prevent accumulation of error (at the beginning of a session the initial matrices are set by the Navigation Engine).

The Navigation Engine implements the global motion constraints using world global orientation and collision detection and corrects the body matrix to keep the body in correct upright orientation and keep it from colliding with obstacles.

Based on the corrected global motion expressed by the corrected body matrix, as well as hand and head positions, the Body Posture Driver generates the body posture reflecting the walking motion and arm movement. The resulting posture is proceeded in terms of joints to the Body Representation Engine for body deformation. At the same time, the resulting posture determines the constraints on the head and hand matrices based on which the new, corrected matrices are generated passed to the Navigation Engine.

The Navigation Engine updates the view matrix used by the rendering pipeline based on the body and head matrices. If object grabbing is requested by the pick information, it tries to grab an object in the vicinity of the user's hand and then moves the object accordingly.

4.4 Autonomous Actors in VLNET

In this subsection we show how all requirements for Autonomous Behaviors (AB) analyzed in section 3.7 are satisfied in VLNET.

VLNET provides *embodiment* through the use of articulated, deformable body representations with articulated faces [Boulic95, Capin95, Capin97, Pandzic97]. This highly realistic embodiment is also a good support for more advanced features like *facial and gestural communication*.

The VLNET interfaces explained in the section 4.2.2 are the simple connection between VLNET and an AB system.

Locomotion capacity is provided through the navigation interface allowing the AB system to move its embodiment through passing of simple matrices. The same interface, through its possibility of picking up and displacing objects, provides the *capacity to act upon objects.* This capacity is further extended by the object behavior interface which allows to access any object in the environment or many objects simultaneously.

The information interface provides the AB system with the *feedback from the environment* with the possibility to request various types of data about the users and objects in the environment.

Facial communication is provided by the facial expression interface which allows the AB system to set any expression on the face of its embodiment.

Gestural communication is possible through the body posture interface, allowing the AB system to change postures of its embodiment and thus perform gestures.

Verbal communication is supported by the text interface. It allows the AB system to get text messages from other users and send text to them. On the other hand, it allows easy connection of speech recognition/synthesis module providing the human user with the possibility to speak with the virtual actor.

4.5 VLNET Performance and Networking results

Considering the graphical and computational complexity of the human representations used in VLNET, we are currently not aiming for a system scaleable to large number of users, but rather trying to obtain a high quality experience for a small number of users. The graphs of performance and network traffic (Figure 24 and Figure 25) show that the system is indeed not scaleable to any larger number of participants. Nevertheless, the results are reasonable for a small number of users and, more importantly, their analysis permits to gain insight to the steps needed to insure a better scaleability.

4.5.1 Experiment design

In order to conveniently simulate a growing number of users we have designed simple drivers to generate some reasonable motion and facial expressions. The navigation driver we used generates a random motion within a room, and the facial expressions driver generates a facial animation sequence repeatedly. By launching several clients on different hosts using these drivers we can easily simulate a session with a number of persons walking in the room and emoting using their faces.

Although we looked for a way to simulate something close to a real multi-user session in a controlled way, there is a difference in the fact that simulated users move their bodies and faces all the time. In a real session, the real users would probably make pauses and thus generate less activity. Therefore we expect somewhat better results in a real session then the ones shown here, although for practical reasons we did not make such tests with real users.

In order to evaluate the overhead induced by the use of high level body representation in the NCVE, we have undertaken three series of measurements: with full body representation, simplified body representation and without a body representation. The full body representation involves complex graphical representation (approx. 10 000 polygons) and deformation algorithms. The simplified body representation consists of a body with reduced graphical complexity (approx. 1500 polygons), with facial deformations and with a simplified body animation based on displacement of rigid body elements (no deformation). The tests without the body representation were made for the

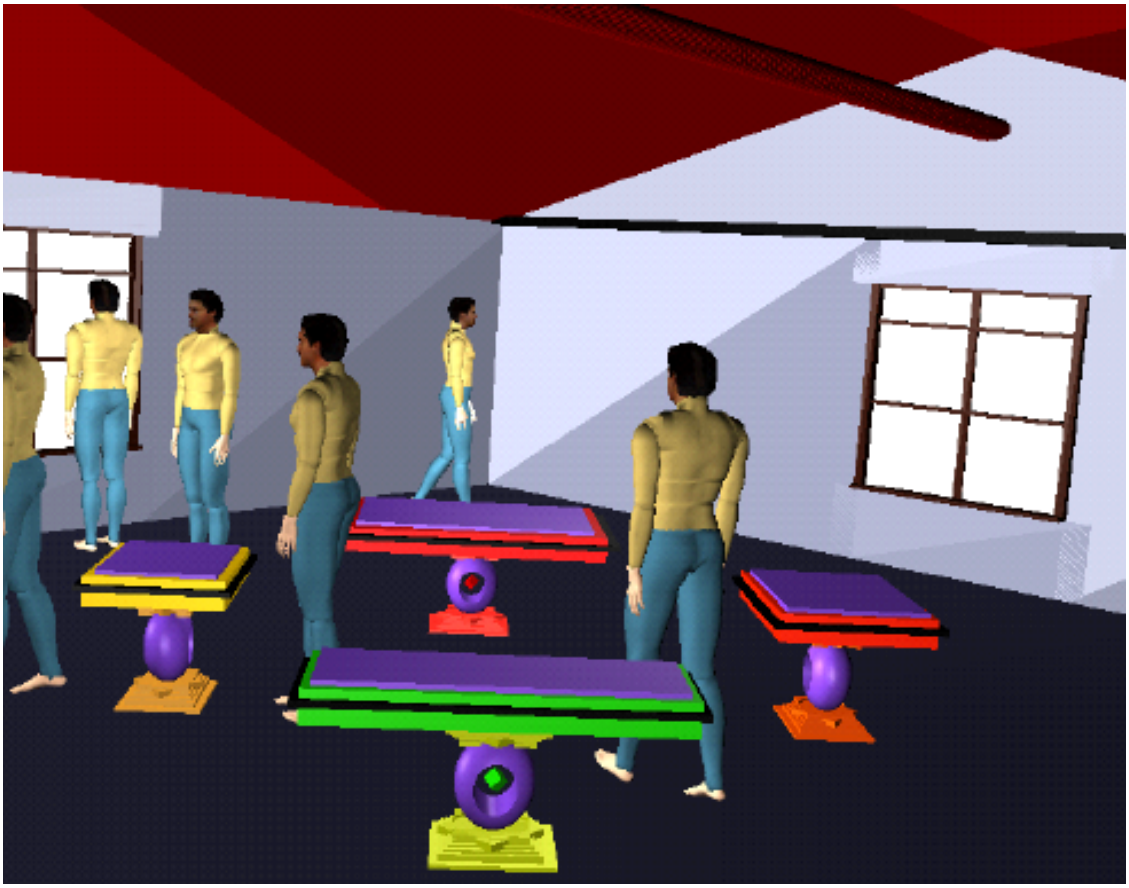
sake of comparison. To mark the positions of users we used simple geometric shapes. No facial or body animation is involved. Figure 23 shows some snapshots from the measurement sessions.

The network traffic (Figure 25) was measured on the server. We have measured incoming and outgoing traffic in kilobits per second.

For the performance measures (Figure 24), we connected one standard, user controlled client to be able to control the point of view. The rest of the clients were user simulations as described above. Performance is measured in number of frames per second. Since it varies depending on the point of view because of the culling (both rendering and computation is culled), we needed to insure consistent results. We have chosen to take measurements with a view where all the user embodiments are within the field of view, which represents the highest strain and the minimal performance for a given number of users in the scene. The performance was measured on a Silicon Graphics Indigo Maximum Impact workstation with 200 MHz R4400 processor and 128M RAM.

4.5.2 Analysis of performance results

Figure 24 shows the variation of performance with respect to the number of users in the simulation with different complexities of body representation as explained in the previous subsection. It is important to notice that this is the minimal performance, i.e. the one measured at the moment when all the users' embodiments are actually within the field of view. Rendering and computation culling boost the performance when the user is not looking at all the other embodiments because they are not rendered, and the deformation calculations are not performed for them. The embodiments that are out of the field of view do not decrease performance significantly, which means that the graph in Figure 24 can be also interpreted as the peak performance when looking at N users, regardless of the total number of users in the scene. This makes the system much more usable than the initial look at the graphs might suggest.



a)

b) c)

Figure 23: Snapshots from performance and network measurements: a) full bodies; b) simplified bodies; c) no body representation

We have also measured the percentage of time spent on two main tasks: rendering and application-specific computation. With any number of users, the rendering takes around 65 % of the time, and the rest is spent on the application-specific computation, i.e. mostly the deformation of faces and bodies. For the case of simplified body representation, the percentage is 58 %. On machines with less powerful graphics hardware, an even greater percentage of the total time is dedicated to rendering.

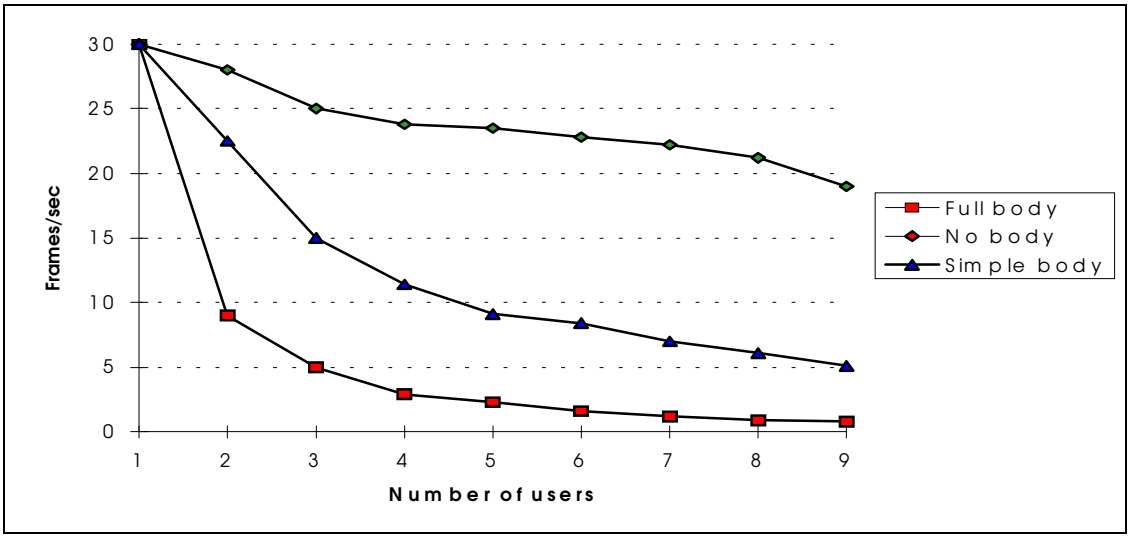


Figure 24: Minimal performance with respect to the number of users in the session

The results show that the use of complex body representation induces a considerable overhead on the rendering side and somewhat less on the computation side. The fact that the performance can be boosted significantly by simply changing the body representation proves that the system framework does not induce an overhead in itself. The system is scaleable in the sense that for a particular hardware and requirements a setup can be found to produce satisfying performance by varying the complexity of the used body and face representation. Most importantly, this proves that the system should lend itself to the implementation of extended Level of Detail [Rohlf94] techniques managing automatically the balance between performance and quality of the human representation.

4.5.3 Analysis of the network results

Figure 25 shows the bit rates measured on the VLNET server during a session with varying number of simulated users walking within a room as described in the subsection on experiment design. We have measured separately the incoming and outgoing traffic, then calculated the total.

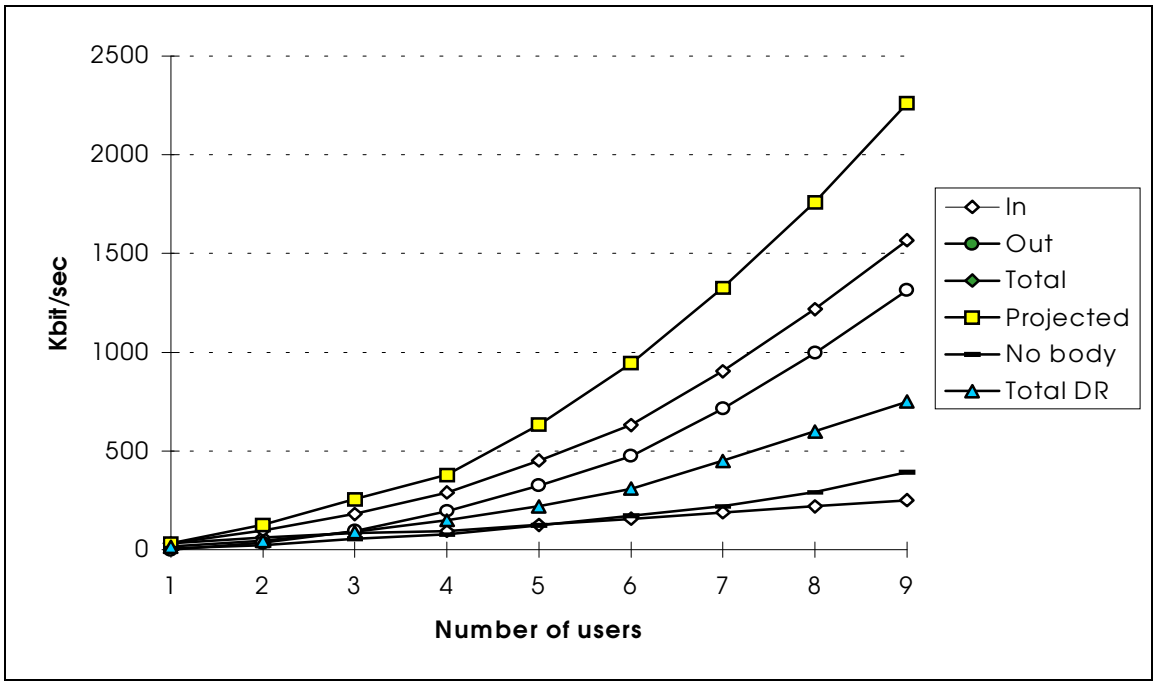


Figure 25: Network traffic measurements with respect to the number of users in the session

Obviously, the incoming traffic grows linearly with the number of users, each user generating the same bitrate. It can be remarked that the bitrate generated per user is roughly 30 Kbit/sec covering the transmission of body positions, body postures and facial expressions.

It is worthwhile remarking that the In traffic curve measured at the server corresponds also to the maximal incoming traffic on a client. The client will receive the incoming bitstream corresponding to the graph in the situations where all the embodiments of other users are in the viewing frustum of the local user, i.e. in the worst case. Otherwise, when looking at N users the incoming traffic will correspond to N users at the graph. This is similar to the situation with performance measurements.

The outgoing traffic represents the distribution of the data to all the clients that need it. The Total traffic is the sum of incoming and outgoing traffic. The Projected traffic curve represents the traffic calculated mathematically for the case of total distribution (all to all). The AOIM technique at the server (see subsection 4.1) insures that the messages are distributed only on as-needed basis and keeps the Total curve well below the Projected curve. Further reduction is achieved using the dead-reckoning technique [Capin 97-1], as illustrated by the curve labeled "Total DR".

Network traffic is the same when using full and simplified body representations, because same messages are transferred. In the case when no body representation is used, less network traffic is generated because there is no need to send messages about body postures and facial expressions. A user without a body representation sends only position updates, generating approximately 8 Kbit/sec. The total traffic curve without a body representation is shown in Figure 25 with label "No body".

The results show that a considerable overhead is introduced by sending face and body data. However, a great potential for improvement must be noticed, because the face and body animation parameters are sent in their raw form, i.e. without and coding/compression. The results shown within the MPEG-4 Ad Hoc Group on Face and Body Animation, where we are actively participating, show that full facial animation can be transmitted at approximately 2 Kbit/sec by using a relatively simple arithmetic coding algorithm. More information on MPEG-4 is given in chapter 6.

4.6 Concluding remarks

Virtual Life Network (VLNET) is an NCVE system based on client/server network topology and using multiple server space structuring strategy. The main feature distinguishing VLNET from other systems is its support for Virtual Humans with deformable, articulated bodies and faces. Another distinguishing feature is the system's open, modular architecture which permits to extend system functionality by adding modules on the application layer, or even to connect whole applications to VLNET, using the VLNET system as a graphical, networked front end for the application. Particular care has been taken to provide powerful support for the integration of autonomous actors in NCVE.

The performance and network measurements quantify the overhead induced on the CPU and the network by the introduction of complex virtual humans. They show that the overhead is considerable, but at the same time they show the potential of the system towards better scaleability using Levels of Detail technique for embodiments and compression techniques for data exchange.

VLNET is a result of a joint research effort including several directly or indirectly involved persons at MIRALab, University of Geneva and LIG, EPFL. While the general system architecture is a result of a long term collaboration between Igor Pandzic at MIRALab and Tolga Capin at LIG, it is possible to distinguish the personal contributions to particular parts of the system as shown in Figure 20. Thus, the Body Representation engine with its interface, as well as the biggest part of the Database process, dead-reckoning algorithm and part of the global constraints in the Navigation Engine were developed by Tolga Capin at LIG. Information and Object Behavior engines, as well as the global constraints in the Navigation Engine were developed by Elwin Lee at MIRALab. Cull and Draw processes are standard part of the IRIS Performer library. Text, Video, Navigation and Face Representation engines with their respective interfaces, Communication process with the Message Queue, as well as the VLNET Server, were predominantly developed by Igor Pandzic as part of this thesis. The developments were not done in an isolated way, therefore the authors made relatively minor updates on all parts the system as needed.

Next chapter shows how we have used VLNET's open architecture to support several methods of facial communication in NCVE by connecting specialized drivers in the VLNET Application Layer.

5. Facial Communication in VLNET

Facial expressions play an important role in human communication. They can express the speaker's emotions and subtly change the meaning of what was said. At the same time, lip movement is an important aid to the understanding of speech, especially if the audio conditions are not perfect or in the case of hearing-impaired listener.

We discuss four methods of integrating facial expressions in a Networked Collaborative Virtual Environment: video-texturing of the face, model-based coding of facial expressions, lip movement synthesis from speech and predefined expressions or animations. The methods vary in computational and bandwidth requirements, quality of the reproduced facial expressions and means of data acquisition. Therefore they are suitable for different situations/applications. We discuss merits, drawbacks and potential application of each method.

5.1 Video-texturing of the face

In this approach the video sequence of the user's face is continuously texture mapped on the face of the virtual human. The user must be in front of the camera, in such a position that the camera captures his/her head and shoulders. A simple and fast image analysis algorithm is used to find the bounding box of the user's face within the image. The algorithm requires that head & shoulder view is provided and that the background is static (though not necessarily uniform). Thus the algorithm primarily consists of comparing each image with the original image of the background. Since the background is static, any change in the image is caused by the presence of the user, so it is fairly easy to detect his/her position. This allows the user a reasonably free movement in front of the camera without the facial image being lost.

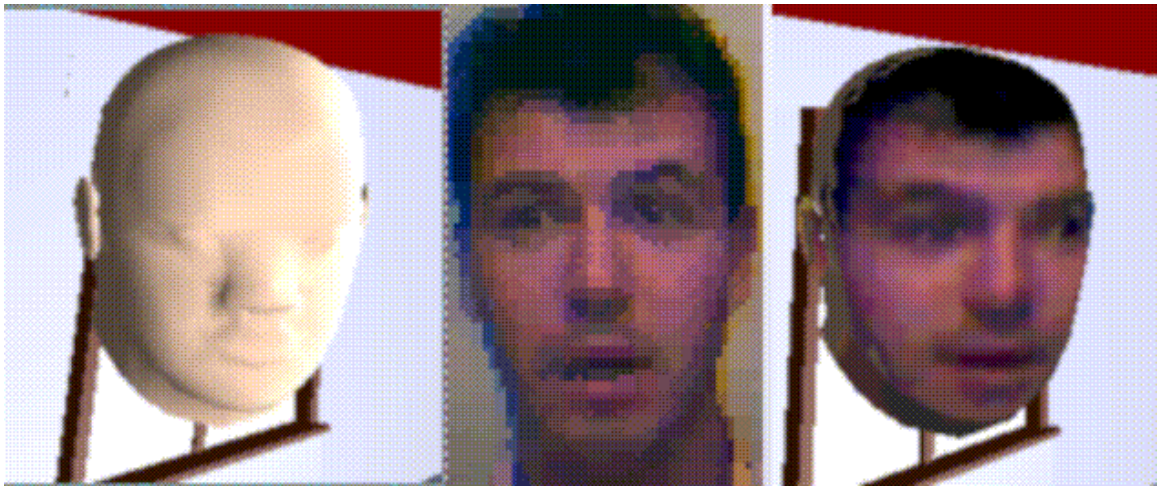


Figure 26: Texture mapping of the face

The VLNET Video Interface is used to pass textures to VLNET, and the Video Engines (see subsection 4.2.1.1) of all VLNET clients in the session receive the texture and map it on the face of the user's embodiment. The texture mapping is illustrated in Figure 26 which shows the face model without the texture, the image used as texture and the final texture-mapped result. We use a simple frontal projection for texture mapping. A simplified head model with attenuated features is used. This allows for less precise texture mapping: if the head model with all the facial features is used, any misalignment

of the topological features in the 3D model and the features in the texture produces quite unnatural artifacts.

Figure 27**:** Video texturing of the face -examples

Figure 27 illustrates the video texturing of the face, showing the original images of the user and the corresponding images of the Virtual Human representation.

The video texturing method achieves high quality of reproduced facial expressions. The bandwidth requirements are relatively high, though still lower then for classical video conferencing systems because of the small size of the used image. The computational complexity is increased because of the compression/decompression algorithms. It is interesting to notice that this method allows tradeoffs between bandwidth and CPU resources by changing the parameters of the compression algorithm. Potential application is the extension of video conferencing to 3D virtual spaces with the additional capability of collaborative work on 3D objects.

5.2 Model-based coding of facial expressions

Instead of transmitting whole facial images as in the previous approach, in this approach the images are analyzed and a set of parameters describing the facial expression is extracted [Pandzic94]. As in the previous approach, the user has to be in front of the camera that digitizes the video images of head-and-shoulders type.

Recognition of facial expressions is a very complex and interesting subject. There have been numerous research efforts in this area. Mase and Pentland [Mase90] apply optical flow and principal direction analysis for lip reading. Terzopoulos and Waters [Terzopoulos91] reported on techniques using deformable curves for estimating face muscle contraction parameters from video sequences. Waters and Terzopoulos [Waters91] modeled and animated faces using scanned data obtained from a radial laser scanner and used muscle contraction parameters estimated from video sequences. Saji et al. [Saji92] introduced a new method called "Lighting Switch Photometry" to extract 3D shapes from the moving face. Kato et al. [Kato92] use isodensity maps for the description and the synthesis of facial expressions. Most of these techniques do not perform the information extraction in real time. There have been some implementations of the facial expression recognition using colored markers painted on the face and/or lipstick [MagnoCaldognetto89, Patterson91, Kishino94]. However, the use of markers is not practical and the methods are needed to perform recognition without them. In another approach Azarbayejani et al. [Azarbayejani93] use extended Kalman filter formulation to recover motion parameters of an object. However, the motion parameters include only head position and orientation. Li et al. [Li93] use the Candid model for 3D motion estimation for model based image coding. The size of the geometric model is limited to only 100 triangles which is rather low for characterizing the shape of a particular model.

Magnenat-Thalmann et al. [MagnenatThalmann93] propose a real time recognition method based on "snakes" as introduced by Terzopoulos and Waters [Terzopoulos91]. The main drawback of this approach, is that the method relies on the information from the previous frame in order to extract the next one. This can lead to the accumulation of error and the "snake" may completely loose the contour it is supposed to follow. To

improve the robustness we adopt a different approach, where each frame can be processed independently from the previous one.

Accurate recognition and analysis of facial expressions from video sequence requires detailed measurements of facial features. Currently, it is computationally expensive to perform these measurements precisely. As our primary concern has been to extract the features in real time, we have focused our attention on recognition and analysis of only a few facial features.

The recognition method relies on the "soft mask", which is a set of points adjusted interactively by the user on the image of the face. Using the mask, various characteristic measures of the face are calculated at the time of initialization. Color samples of the skin, background, hair etc., are also registered. Recognition of the facial features is primarily based on color sample identification and edge detection. Based on the characteristics of human face, variations of these methods are used in order to find the optimal adaptation for the particular case of each facial feature. Special care is taken to make the recognition of one frame independent from the recognition of the previous one in order to avoid the accumulation of error. The data extracted from the previous frame is used only for the features that are relatively easy to track (e.g. the neck edges), making the risk of error accumulation low. A reliability test is performed and the data is reinitialized if necessary. This makes the recognition very robust. The set of extracted parameters includes:

- vertical head rotation (nod)
- horizontal head rotation (turn)
- head inclination (roll)
- aperture of the eyes
- horizontal position of the iris
- eyebrow elevation
- distance between the eyebrows (eyebrow squeeze)
- jaw rotation
- mouth aperture

- mouth stretch/squeeze

The following sections describe the initialization of the system and the details of the recognition method for each facial feature, as well as the verification of the extracted data. The recognition of the features and the data verification are presented in the order of execution, as also shown schematically in Figure 28.

5.2.1 Initialization

Initialization is done on a still image of the face grabbed with a neutral expression. The soft mask is placed over the image as shown in Figure 29. The points of the mask are interactively adjusted to the characteristic features of the face, such as mouth, eyes, eyebrows etc. These points determine the measures of the face with neutral expression and provide color samples of the background and the facial features. The process of setting the mask is straightforward and usually takes less than half a minute.

5.2.2 Head tracking

First step is to find the edges of the neck (blue circles in Figure 30, points N1 and N2 in Figure 31). During the initialization, color samples are taken at the points 1, 2 and 3 of the mask (Figure 29). Points 1 and 3 are aligned over background and skin respectively, and point 2 over the hair falling on the side of the face, if any. During recognition, a sample taken from the analyzed point of the image is compared with those three samples and identified as one of them. As each color sample consists of three values (red, green and blue), it can be regarded as a point in a three dimensional RGB space. The distance in this space between the sample being analyzed and each stored sample is calculated. The closest one is chosen to categorize the point. This method of sample identification works fine in the areas where the number of possible different colors is small and where there is sufficient difference between the colors.

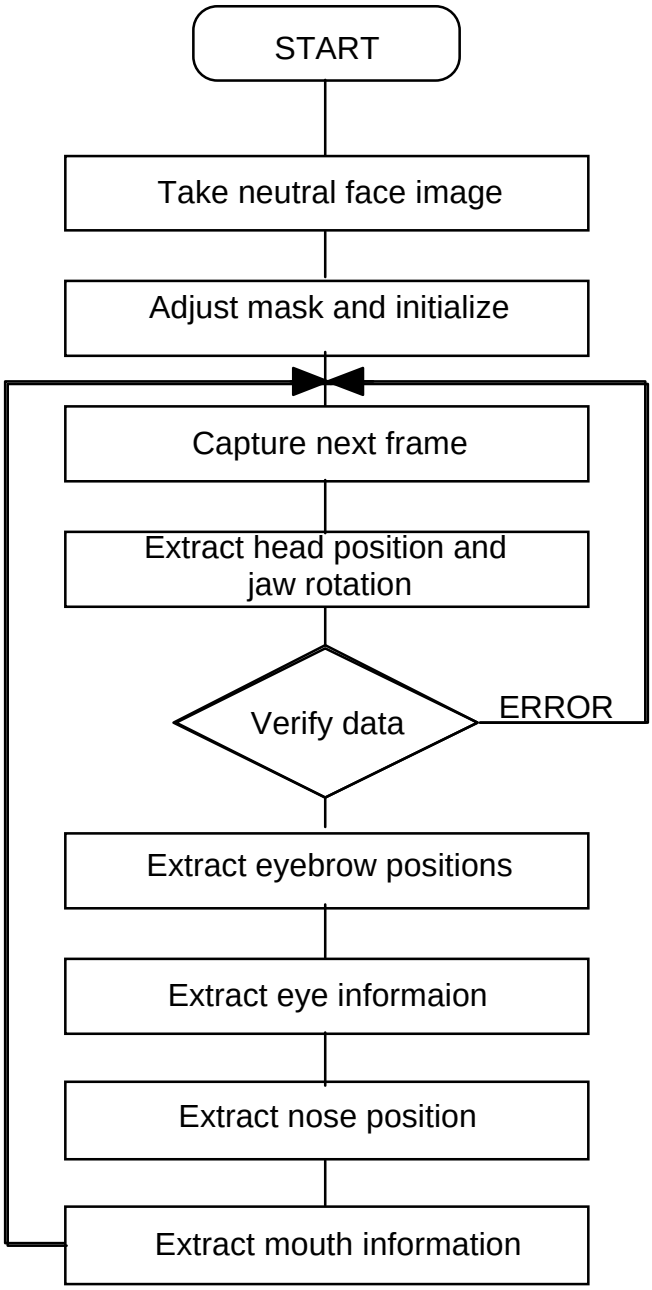


Figure 28: Flowchart of the facial recognition method

Next step is to find the hairline (marked with the red circle in Figure 30, point M in Figure 31). The samples of the hair and skin color are taken and edge between the two is detected. The horizontal position of the starting point is halfway between the neck edges, and the vertical position is taken from the previous frame. At a fixed distance below the hairline the edges of the hair seen on the sides of the forehead are detected (marked with

green and yellow circles in Figure 30, points L1, L2, R1, R2 in Figure 31) using the above described sample identification method.

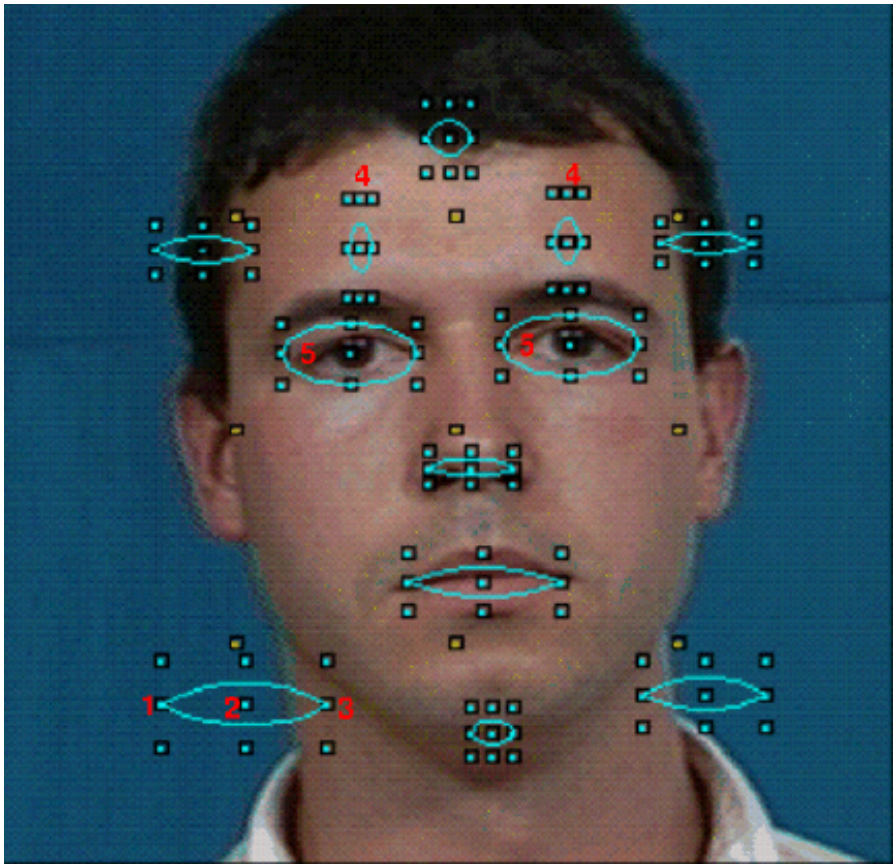


Figure 29: Recognition initialization - neutral face with the soft mask

Using the information from points L1, L2, R1, R2, N1, N2, and M (Figure 31) we estimate the head orientation for different movements. For example:

head turn = f(L1,L2,R1,R2)

head nod = f(M)

head roll = f(L1,L2,R1,R2,N1,N2)

5.2.3 Jaw rotation

To extract the rotation of the jaw the position of the chin has to be found. We exploit the fact that the chin casts a shadow on the neck, which gives a sharp color

change on the point of the chin. Once again the sample identification is used to track this edge.

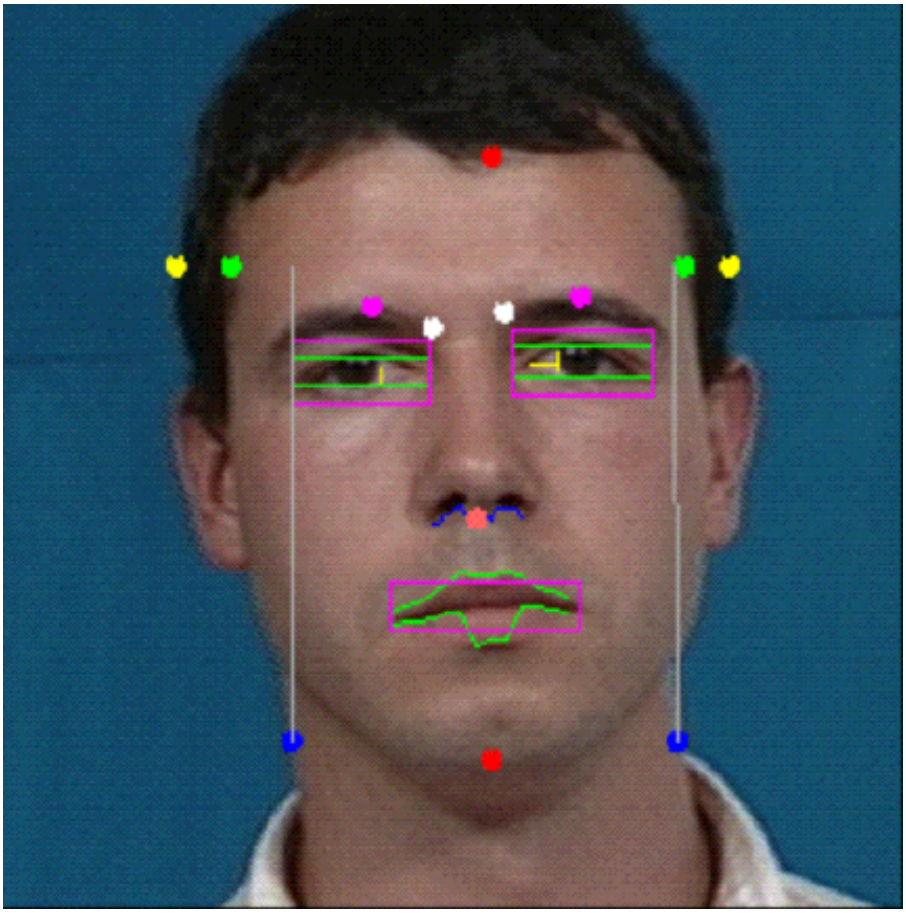


Figure 30: Face with recognition markers

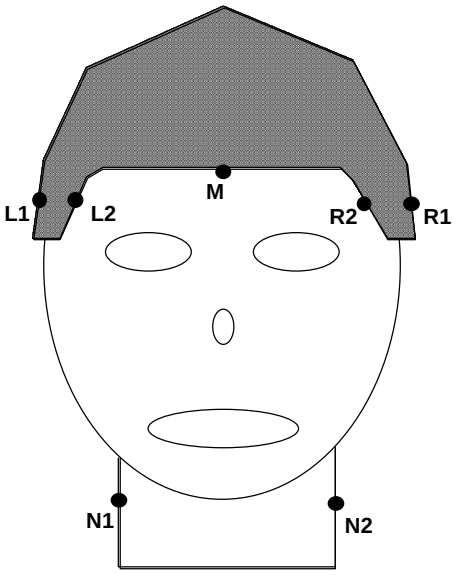


Figure 31: Points used in facial feature tracking

5.2.4 Data verification

At this point the data extracted so far is checked against the measurements of the face made during initialization. If serious discrepancies are observed the recognition of the frame is interrupted, the warning signal is issued and the data is reinitialized in order to recognize the next frame correctly. This may happen if the user partially or totally leaves the camera field of view or if he/she takes such a position that the recognition cannot proceed.

5.2.5 Eyebrows

The starting points for the eyebrow detection are above each eyebrow, sufficiently high that the eyebrows cannot be raised above them. They are adjusted interactively during initialization (points marked 4 in Figure 29) and kept at fixed position with respect to the center of the hairline. Also during initialization, the color samples of the skin and the eyebrows are taken. The search proceeds downwards from the starting point until the color is identified as eyebrow. To avoid wrinkles on the forehead being confused with the eyebrows, the search is continued downward after a potential eyebrow is found. If that is the real eyebrow (i.e. not just a wrinkle), the next sample resembling the eyebrow will be in the eye region, i.e. too low. The points on eyebrows are marked with magenta circles in Figure 30. The relative position of each eyebrow with respect to the hairline is compared with the eyebrow position in the neutral face to determine the eyebrow-raise. The eyebrow squeeze is calculated from the distance between the left and right eyebrow.

5.2.6 Eyes

During initialization, a rectangle (marked as 5 in Figure 29) is placed over each eye and its position relative to the center of the hairline is measured. During recognition the rectangles (outlined in magenta in Figure 30) are fixed with respect to the center of the hairline and stay around the eyes when the user moves.

To determine the aperture of the eye we exploit the fact that the sides of the iris make strong vertical edges in the eye region. The points lying on vertical edges are found as the local minima of a simplified color intensity gradient function. The edges are found

by searching for the groups of such points connected vertically. The largest vertical edge is a side of the iris. To find the aperture of the eye we search for the eyelid edges upwards and downwards from the extremes of the vertical edge found earlier. In Figure 30, the aperture of the eyes is marked with green lines, the vertical yellow line marking the side of the iris.

To determine the horizontal position of the iris we find the distance between the iris and the edge of the eye using simple edge detection. This distance is marked with a horizontal yellow line.

Figure 32**:** Model-based coding of the face - original and synthetic face

5.2.7 Nose and Mouth

The distance between the nose and the hairline is measured during initialization. Using this value the approximate position of the nose is determined. Edge detection is used for locating the nose. A point where the vertical color intensity gradient is above a certain threshold, is considered to lie on a horizontal edge. A 3x3 pixels gradient operator is used. The threshold value is determined during initialization by exploring the gradient

values in the area. The blue line in the Figure 30 connects the edge points and the orange circle marks the nose position.

For acquisition in the mouth region we search for a horizontal edge downward the nose point to find a point on the upper lip. At the same horizontal position the search is performed from the chin in upward direction to find a point on the lower lip. This process is repeated on the next horizontal position n pixels to the right, n being 1/10 of the mouth width. The search starts in the proximity of the found vertical positions. We continue to move to the right, each time storing in memory the points on the lips edges found, until the corner of the lips is passed. This is detected when no edge is found in the area. The corner of the lips is then tracked more precisely by decreasing the step to n/2, n/4, n/8,...,1. The same process is repeated for the left side. All the points found together thus form the mouth curve. It is shown in green in Figure 30. However, due to shadows, wrinkles, beard or insufficient lip-skin color contrast, the curve is not very precise. Therefore the average height of the points in the middle third of the curve is taken for the vertical position of the lip. The bounding rectangle of the mouth is outlined in magenta. This rectangle provides measures for the relative vertical positions of upper and lower lip and squeeze/stretch of the mouth.

The analysis is performed by a special Facial Expression Driver. The extracted parameters are easily translated into Minimal Perceptible Actions, which are passed to the Facial Representation Engine, then to the Communication process, where they are packed into a standard VLNET message packet and transmitted.

On the receiving end, the Facial Representation Engine receives messages containing facial expressions described by MPAs and performs the facial animation accordingly. Figure 32 illustrates this method with a sequence of original images of the user (with overlaid recognition indicators) and the corresponding images of the synthesized face.

This method can be used in combination with texture mapping. The model needs an initial image of the face together with a set of parameters describing the position of the facial features within the texture image in order to fit the texture to the face. Once this is done, the texture is fixed with respect to the face and does not change, but it is deformed

together with the face, in contrast with the previous approach where the face was static and the texture was changing. Some texture-mapped faces with expressions are shown in Figure 33.

The bandwidth requirements for this method are very low. However, considerable computing power is needed, and the complexities of facial expression extraction from video are not resolved in a satisfying way yet; therefore the method yields less quality in reproducing facial expressions than desirable. Providing that the extraction is improved, this method is promising for very low bitrate conferencing in virtual environments.

5.3 Lip movement synthesis from speech

It might not always be practical for the user to be in front of the camera (e.g. if he/she doesn't have one, or if he/she wants to use a HMD). Nevertheless, the facial communication does not have to be abandoned. Fabio Lavagetto [Lavagetto95] shows that it is possible to extract visual parameters of the lip movement by analyzing the audio signal of the speech. An application doing such recognition and generating MPAs for the control of the face can be connected to VLNET as the Facial Expression Driver, and the Facial Representation Engine will be able to synthesize the face with the appropriate lip movement corresponding to the pronounced speech. However, currently available software for the segmentation of the audio signal into phonemes does not provide real time performance, therefore we found it unsuitable for integration in VLNET and we have developed a simpler method that analyses the audio signal and produces a simple open/close mouth movement when speech is detected, allowing the participants in the NCVE session to know who is speaking.

This method in its current implementation has very low bandwidth and computing power requirements, but the functionality is limited to the indication of the active speaker. The full implementation of the method, with phoneme extraction from the audio signal and accurate synthesis of visual speech, would still be in very low bitrate domain, however the computing complexity is quite high. Nevertheless, the gap to real time implementation is not too big and it is realistic to expect this implementation to be possible in near future. This will allow potential enhancement of speech intelligibility in noisy environments or for hearing impaired persons.

5.4 Predefined expressions or animations

In this approach the user can simply choose between a set of predefined facial expressions or movements (animations). The choice can be done from a menu. The Facial Expression Driver in this case stores a set of defined expressions and animations and just feeds them to the Facial Representation Engine as the user selects them.

Figure 33 shows some examples of predefined facial expressions.

This method is relatively simple to implement and cheap in terms of both bandwidth and computing power. Its potential usage is in the virtual chat rooms on the network, where the user community already has a culture of using character-based “smileys” to express emotions.

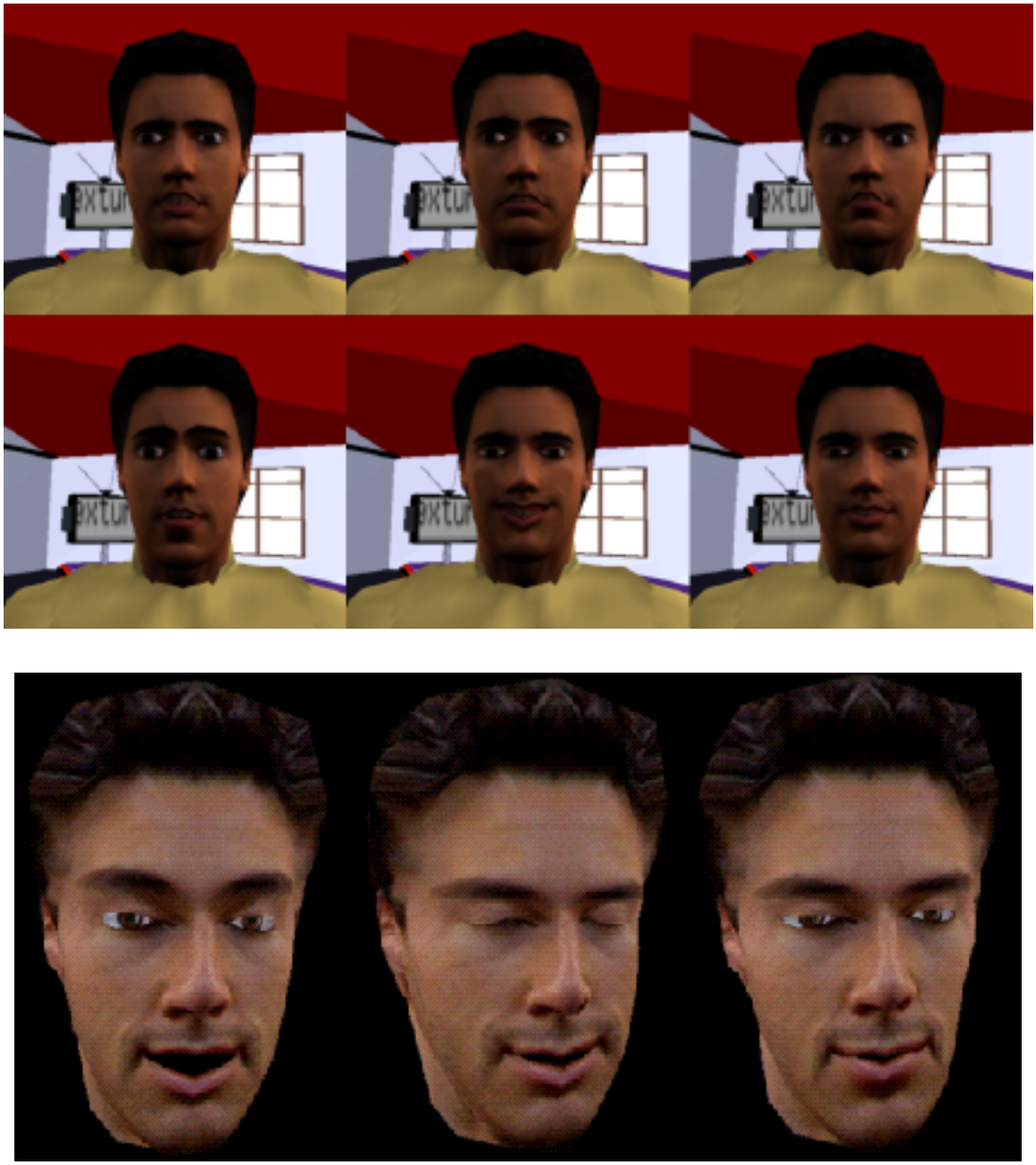


Figure 33**:** Predefined facial expressions - examples

6. Relations with the MPEG-4 standard

In parallel with our work on NCVEs, we participate in the work of ISO/IEC JTC1/SC29/WG11 - better known as MPEG. MPEG is traditionally committed to coding and compression of audio-visual data from natural sources. However, the emerging MPEG-4 standard aims not only at multiple natural audio-visual objects composing the scene, but also synthetic audio and video to be integrated with the natural. It will also allow more interaction with both synthetic and natural objects. Within the MPEG-4 Ad Hoc Group on Face and Body Animation we have provided a major contribution to the specification of Face Animation Parameters and Face Definition Parameters. This experience has lead us to believe that there is a strong potential relation of MPEG-4 standard to NCVEs and that it will be possible in near future to build rich multimedia 3D networked environments based on this standard.

In this chapter we analyze the potential usage of MPEG-4 for NCVE systems. In section 6.1 we briefly introduce the MPEG-4 standard, with more concentration on Face and Body Animation part of the standard. In section 6.2 we present in a systematic way the requirements on the bitstream contents encountered in NCVE applications, and in section 6.3 we analyze how these requirements can be met by the MPEG-4 standard. In the last section we bring the concluding remarks.

6.1 Introduction to MPEG-4

ISO/IEC JTC1/SC29/WG11 (Moving Pictures Expert Group - MPEG) is currently working on the new MPEG-4 standard, scheduled to become International Standard in February 1999. In a world where audio-visual data is increasingly stored, transferred and manipulated digitally, MPEG-4 sets its objectives beyond "plain" compression. Instead of regarding video as a sequence of frames with fixed shape and size and with attached audio information, the video scene is regarded as a set of dynamic objects. Thus the background of the scene might be one object, a moving car another, the sound of the engine the third etc. The objects are spatially and temporally independent and therefore can be stored, transferred and manipulated independently. The composition of the final scene is done at the decoder, potentially allowing great manipulation freedom to the consumer of the data.

Video and audio acquired by recording from the real world is called natural. In addition to the natural objects, synthetic, computer generated graphics and sounds are being produced and used in ever increasing quantities. MPEG-4 aims to enable integration of synthetic objects within the scene. It will provide support for 3D Graphics, synthetic sound, Text to Speech, as well as synthetic faces and bodies.

Currently there are four groups that work on producing MPEG-4 standards: Systems, Audio, Video and Synthetic/Natural Hybrid Coding (SNHC). The standard will finally consist of Systems, Audio and Video parts, and the specifications produced by SNHC will be integrated in either Audio or Video.

The Systems layer supports demultiplexing of multiple bitstreams, buffer management, time identification, scene composition and terminal configuration.

MPEG-4 video provides technologies for efficient storage, transmission and manipulation of video data in multimedia environments. The key areas addressed are efficient representation, error resilience over broad range of media, coding of arbitrarily shaped video objects, alpha map coding.

MPEG-4 Audio standardizes the coding of natural audio at bitrates ranging from 2 Kbit/sec to 64 bits/sec, addressing different bitrate ranges with appropriate coding technologies.

Synthetic/Natural Hybrid Coding (SNHC) deals with coding of synthetic audio and visual data. In particular, a subgroup of SNHC deals with the animation of human faces and bodies. We present in more detail the activities of this group and their current draft specification in the following subsection.

6.1.1 Face and Body Animation (FBA)

The Face and Body animation Ad Hoc Group (FBA) deals with coding of human faces and bodies, i.e. efficient representation of their shape and movement. This is important for a number of applications ranging from communication, entertainment to ergonomics and medicine. Therefore there exists quite a strong interest for standardization. The group has defined in detail the parameters for both definition and animation of human faces and bodies. This draft specification is based on proposals from several leading institutions in the field of virtual humans research. It is being updated within the current MPEG-4 Committee Draft [MPEG-N1901, MPEG-N1902].

Definition parameters allow detailed definition of body/face shape, size and texture. Animation parameters allow to define facial expressions and body postures. The parameters are designed to cover all naturally possible expressions and postures, as well as exaggerated expressions and motions to some extent (e.g. for cartoon characters). The animation parameters are precisely defined in order to allow accurate implementation on any facial/body model.

6.1.1.1 Facial Animation Parameter set

The FAPs are based on the study of minimal facial actions and are closely related to muscle actions. They represent a complete set of basic facial actions, and therefore allow the representation of most natural facial expressions. The lips are particularly well defined and it is possible to precisely define the inner and outer lip contour. Exaggerated values permit to define actions that are normally not possible for humans, but could be desirable for cartoon-like characters.

All the parameters involving translational movement are expressed in terms of the Facial Animation Parameter Units (FAPU). These units are defined in order to allow interpretation of the FAPs on any facial model in a consistent way, producing reasonable results in terms of expression and speech pronunciation. They correspond to fractions of distances between some key facial features (e.g. eye distance). The fractional units used are chosen to allow enough precision.

The parameter set contains two high level parameters. The viseme parameter allows to render visemes on the face without having to express them in terms of other parameters or to enhance the result of other parameters, insuring the correct rendering of visemes. The full list of visemes is not defined yet. Similarly, the expression parameter allows definition of high level facial expressions.

6.1.1.2 Facial Definition Parameter set

An MPEG-4 decoder supporting the Facial Animation must have a generic facial model capable of interpreting FAPs. This insures that it can reproduce facial expressions and speech pronunciation. When it is desired to modify the shape and appearance of the face and make it look like a particular person/character, FDPs are necessary.

The FDPs are used to personalize the generic face model to a particular face. The FDPs are normally transmitted once per session, followed by a stream of compressed FAPs. However, if the decoder does not receive the FDPs, the use of FAPUs insures that it can still interpret the FAP stream. This insures minimal operation in broadcast or teleconferencing applications.

The Facial Definition Parameter set can contain the following:

- 3D Feature Points
- Texture Coordinates for Feature Points (optional)
- Face Scene Graph (optional)
- Face Animation Table (FAT) (optional)

The Feature Points are characteristic points on the face allowing to locate salient facial features. They are illustrated in Figure 34. Feature Points must always be supplied, while the rest of the parameters are optional.

The Texture Coordinates can be supplied for each Feature Point.

The Face Scene Graph is a 3D polygon model of a face including potentially multiple surfaces and textures, as well as material properties.

The Face Animation Table (FAT) contains information that defines how the face will be animated by specifying the movement of vertices in the Face Scene Graph with respect to each FAP as a piece-wise linear function. We do not deal with FAT in this paper.

The Feature Points, Texture Coordinates and Face Scene Graph can be used in four ways:

- If only Feature Points are supplied, they are used on their own to deform the generic face model.
- If Texture Coordinates are supplied, they are used to map the texture image from the Face Scene Graph on the face deformed by Feature Points. Obviously, in this case the Face Scene Graph must contain exactly one texture image and this is the only information used from the Face Scene Graph.
- If Feature Points and Face Scene Graph are supplied, and the Face Scene Graph contains a non-textured face, the facial model in the Face Scene Graph is used as a Calibration Model. All vertices of the generic model must be aligned to the surface(s) of the Calibration Model.
- If Feature Points and Face Scene Graph are supplied, and the Face Scene Graph contains a textured face, the facial model in the Face Scene Graph is used as a Calibration Model. All vertices of the generic model must be aligned to the surface(s) of the Calibration Model. In addition, the texture from the Calibration Model is mapped on the deformed generic model.

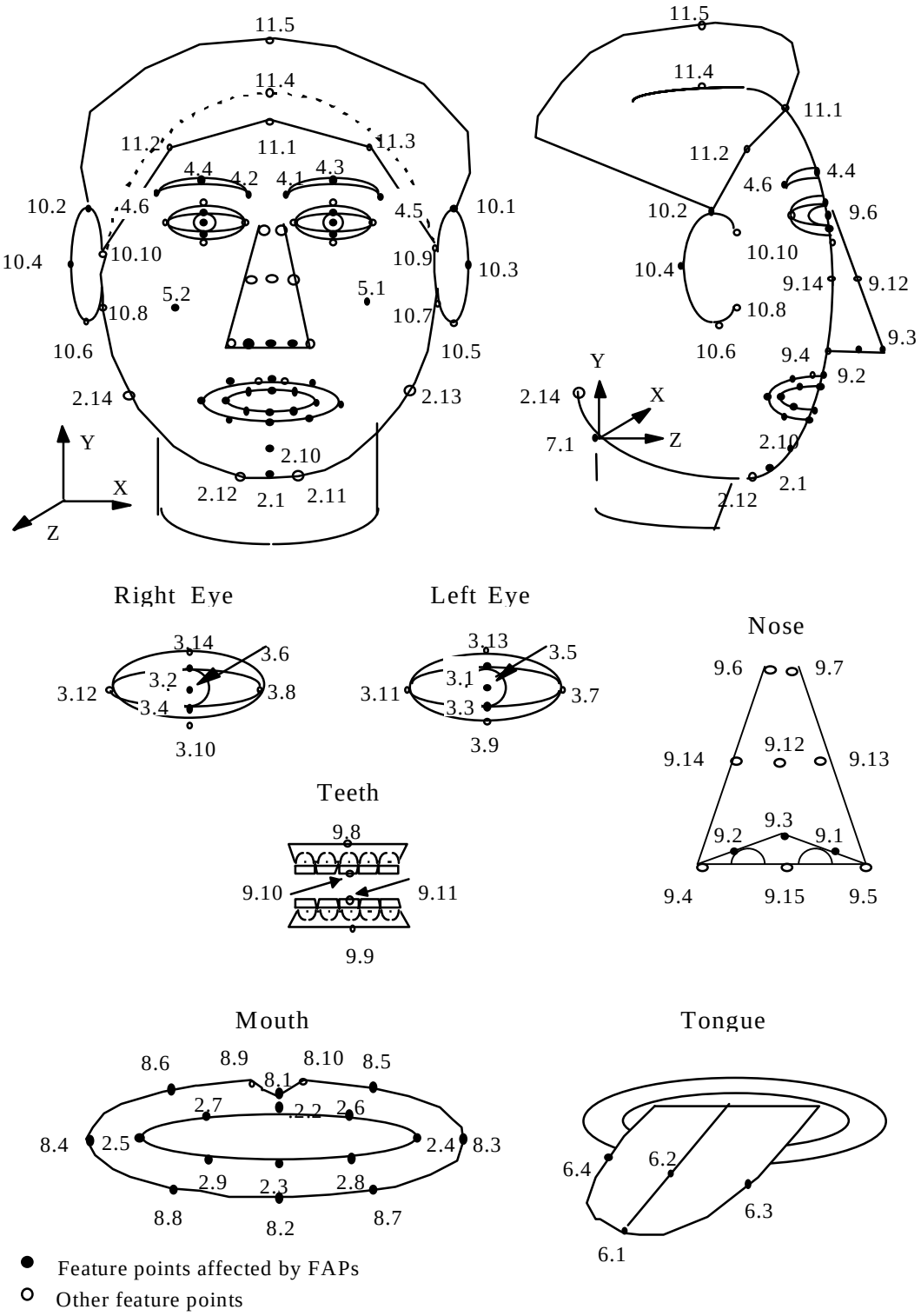


Figure 34: FDP Feature Points

6.2 Bitstream contents in NCVE applications

We analyze various data types that are transmitted through the network in NCVE systems. Figure 35 presents an overview of all data types usually encountered. Current systems usually support only a subset of the data types presented here.

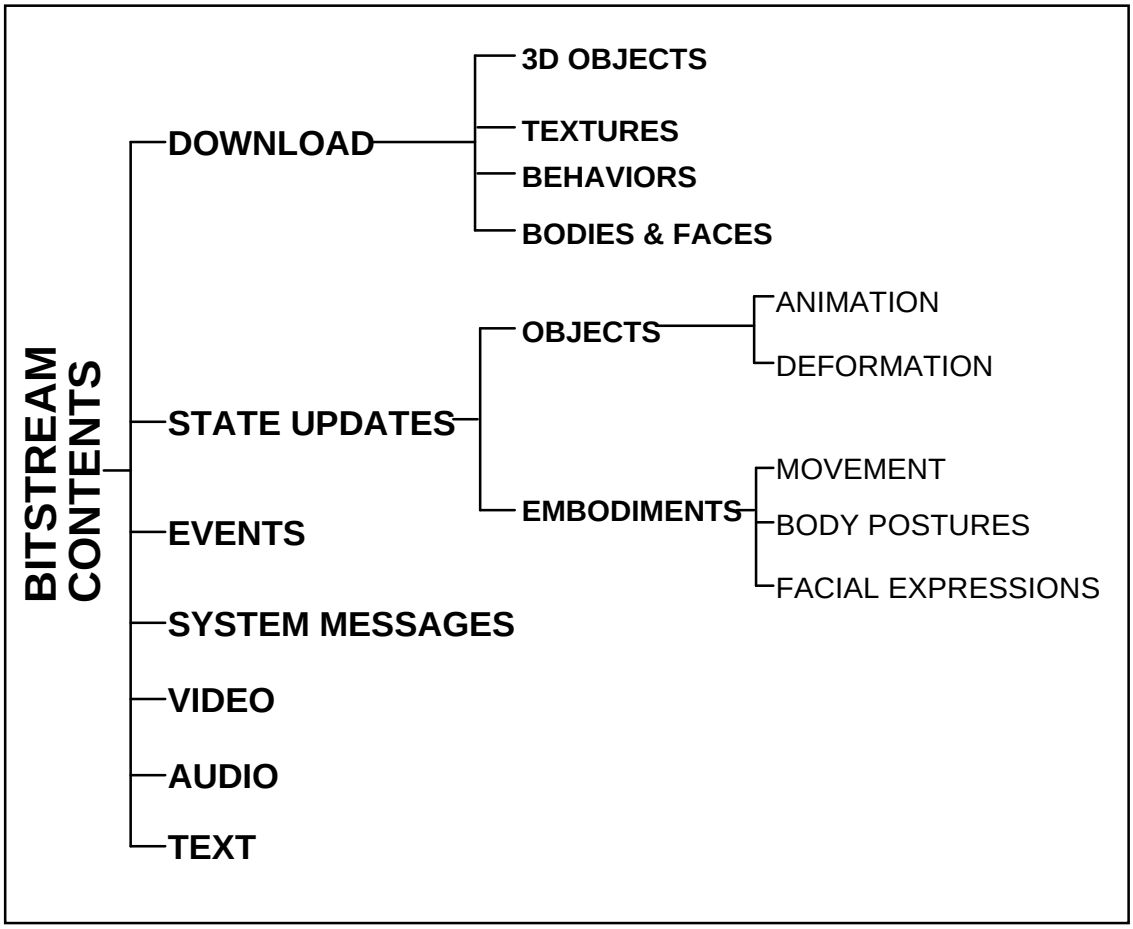


Figure 35: Bitstream contents breakdown for NCVE systems

6.2.1 Download

The main need for download arises when a new user joins a NCVE session. At this moment the complete description of the Virtual Environment has to be downloaded to the new user. This includes 3D objects structured in a scene hierarchy, textures and possibly behaviors in form of scripts or programs. The new user also has to download the embodiment descriptions of all users and send his own to everyone. The embodiment

might be a simple geometrical object, but in a more sophisticated system it should be a body and face description in a form that allows later animation of both body and face.

Downloads are not restricted to the session establishment phase, they can also occur anytime during the session if new objects are introduced in the scene - they have to be distributed.

6.2.2 State updates

All state changes for both the environment itself and the users' embodiments as special part of the environment have to be propagated through the network.

For the objects in the environment, this commonly involves the change of position/orientation, i.e. animation of rigid objects. In non-networked VEs objects are often deformed and not only animated rigidly. Free deformation of objects is not commonly supported in NCVE systems because of increased bandwidth needs - all displaced vertices have to be updated. It is however a very desirable feature to be included in NCVE systems.

For user embodiments, state updates also involve basic movement, and in the case of simple, rigid embodiments this is enough. For articulated, human-like embodiments means must be provided to communicate body postures and facial expressions to allow the simulation of natural body movements and actions.

6.2.3 Events

These are typically short messages about events happening in the environment. The basic difference from state updates is that a state message makes all previous state messages for the same object obsolete (e.g. a new position of the object makes all previous positions obsolete) [Kessler96], which is not the case with event messages. An event is sent only once and can influence the environment state potentially for a long time. Therefore the security of event messages must be higher then for state updates.

6.2.4 System Messages

These messages are used typically during session establishment and log-off. Their security is essential because errors can cause serious malfunction of the system.

6.2.5 Video

Video can be streamed and texture-mapped on any object in the environment producing real-time video textures. This can be used in different ways for various applications. Examples include virtual video presentations, facial texture mapping for facial communication as presented in chapter 5, as well as mixing of real and virtual worlds for augmented reality applications or virtual studios.

The requirements on the video quality may vary from application to application.

6.2.6 Audio

The most common use of audio in NCVEs is for speech communication. However, synthetic 3D sound can also be an important part of a Virtual Environment.

6.2.7 Text

The most common use of text is for text-based chat between users. There are however more interesting ways to use it. As explained in section 3.7, autonomous Virtual Humans with built-in AI decision-making algorithms can also be participants in NCVEs. The best way to communicate orders/questions/dialogues to them is through text, and that is also the easiest way for them to respond. Speech recognition and Text-to-Speech (TTS) systems (possibly coupled with facial animation for lip sync) can be used to interface naturally with the users.

6.3 How MPEG-4 can meet NCVE requirements

Based on the requirements laid out in the previous section and MPEG-related documentation [MPEG-N1886, MPEG-N1901, MPEG-N1902, Koenen97, Doenges97] we study how MPEG-4 tools can be deployed to solve the problems of NCVE.

As for the network topologies, the solutions are out of scope of MPEG which mostly concentrates on bitstream contents. It is however worthwhile noting that an MPEG-4 system will be capable of receiving objects from up to 8 different sources, a fact to be considered when planning network topologies for particular applications [MPEG-N1886].

In the following subsections we study various components of bitstream contents with respect to MPEG-4.

6.3.1 Download

The MPEG-4 standard shall provide the means to download and store audio, video and synthetic objects [MPEG-N1886]. Furthermore, progressive transmission based on scaleable coding techniques will be supported.

MPEG-4 will support a VRML-like 3D geometry hierarchy with all attributes, as well as behavior data. It will provide means for efficient compression of 3D meshes. Efficient still texture coding will be supported, as well as spatial- and quality- scaleable coding to fit available bandwidth and rendering capabilities.

MPEG-4 will support body and face objects. Using Body Definition Parameters (BDPs) and Face Definition Parameters (FDPs) [MPEG-N1901, MPEG-N1902] it is possible to define body and face representation. BAPs and FAPs are scaleable, offering a wide choice of tradeoffs between definition quality and bandwidth required. In case of absence of FDPs and/or BDPs generic bodies and faces can be used.

6.3.2 State updates

MPEG-4 supports scaling, rotation and translation of any video object (natural or synthetic, i.e. 3D objects included) about any axis in 2D or 3D space [MPEG-N1886]. Changes in audio objects localization are supported.

As for deformable 3D objects, at the time of writing this Ph.D. thesis the need was recognized in MPEG to support efficient coding of object deformations, though this specification was not yet included in the documents.

MPEG-4 will support efficient coding of face and body animation. The parameters are defined to express body postures and facial expressions in an efficient manner and independently of a particular face/body model. These parameters are compressed to obtain very small bitrates (e.g. approx. 2 Kbit/sec for facial expressions).

6.3.3 Events and system messages

These messages are specific to a particular NCVE system and as such are not explicitly covered by MPEG-4.

6.3.4 Video

Video is a traditional part of MPEG and the video tools are mature and extensive. MPEG-4 Video will support all types of pixel-based video with high compression efficiency. Tools will be provided to achieve error resilient video streams over a variety of networks with possibly severe error conditions, including low-bitrate networks [MPEG-N1886]. Scaleability in terms of content and spatial and temporal quality will be supported. Various delay modes, including low delay modes for real-time communication will be supported. MPEG-4 tools are optimized for the following bitrate ranges: < 64 Kbit/sec (low), 64 - 384 Kbit/sec (intermediate) and 384 Kbit/sec - 1.8 Mbit/sec (high). Various video formats will be supported.

6.3.5 Audio

MPEG-4 will support following types of audio content: high quality audio (> 15 KHz), intermediate quality audio (<15 KHz), wideband speech (50 Hz - 7 KHz), narrowband speech (50 Hz - 3.6 KHz) and intelligible speech (300 Hz - 3.4 KHz) [MPEG-N1886]. Tools will be provided to achieve error resilient audio streams, including support for low bitrate applications. In particular, speech coding compression will support intelligible speech at 2 Kbit/sec. A number of audio formats, as defined by sampling frequency, amplitude resolution, quantizer characteristics and number of channels will be supported.

6.3.6 Text

Simple text is not explicitly supported by MPEG-4. However, there will be tools for Text-to-Speech functionality [MPEG-N1886] which requires at least simple text, and possibly auxiliary information such as phoneme duration, amplitude of each phoneme etc. Capability to synchronize TTS output with facial animation system visualizing the pronunciation will be supported.

6.3.7 Integration

For NCVE systems it is not only important to support all the data types described in previous sections, but also to achieve an orderly integration of all data types with respect to relative priorities and synchronization.

MPEG-4 shall support dynamic multiplexing of all objects [MPEG-N1886]. Means will be provided to identify relative importance of parts of coded audio-visual information with at least 32 levels of priority. Synchronization between all objects is supported, with specified maximal differential delays (e.g. between two video objects or between an audio and a video object).

6.4 Concluding remarks

We have analyzed the networking requirements of Networked Collaborative Virtual Environments, and how MPEG-4 tools can be used to fulfill these requirements.

The building of network topology, session establishment/destruction and system-particular message passing are out of MPEG-4 scope and should be dealt with on another level. However, most of the data types that are important for NCVE systems will be very well supported by MPEG-4 tools (video, audio, 3D objects, textures, bodies, faces). On top of this, MPEG-4 will offer reliable multiplexing, mechanisms for establishing priorities among data, as well as synchronization. We believe that MPEG-4 tools should play an important role in building future Networked Collaborative Virtual Environment systems.

7. Conclusion

In this chapter we summarize our contribution, present potential applications of our work and discuss ideas for future work.

7.1 Contribution

To fulfill the defined objectives of our work, we have completed the following tasks:

- Development of a Networked Collaborative Virtual Environment framework integrating Virtual Humans
- Development of techniques for facial communication in NCVEs

An additional contribution of this thesis is to provide an analysis of the potential usability of the currently developed MPEG-4 standard in the field of NCVEs.

We summarize each of these contributions in the following subsections.

7.1.1 Development of a Networked Collaborative Virtual Environment framework integrating Virtual Humans

The survey of previous work and existing NCVE systems has shown that most existing systems use relatively simple graphical models to represent users in the VE. The survey of related research has also shown expectations of increased quality and usability of NCVE systems with the introduction of more sophisticated human-like representations. We have analyzed the challenges and requirements for introducing Virtual Humans (VH) as participant representation in NCVE, and developed the Virtual Life Network (VLNET) system. VLNET provides a modular, open system architecture with a set of extension interfaces. These interfaces provide easy access not only to functions related to VH support, but to all other functions of NCVE. Therefore VLNET offers flexible support for different functions and applications, and lends itself very well also as a research testbed.

VLNET is a joint research effort by MIRALab, University of Geneva and LIG, EPFL and the general system architecture is a result of this collaboration. Specific parts of the system that have been developed as part of this thesis work are: Text, Video, Face and Navigation engines with their respective interfaces, Communication process with the Message Queue as well as the VLNET Server. Illustration of these system components

can be found in Figure 20, and a more detailed discussion on individual contributions to VLNET in section 4.6.

7.1.2 Development of techniques for facial communication in NCVEs

The survey of previous work shows a lack of means for natural communication in NCVE systems, in particular facial communication. Using our VLNET system as the basic framework, we have built four different methods of facial communication in NCVE.

Video texturing method provides very good results in terms of reproduction of facial expression and the face itself; however, it is rather costly in terms of bandwidth required to stream the video texture through the network. By choosing different video compression algorithms, different tradeoffs between video quality, bandwidth and CPU load can be obtained.

Model based coding of facial expressions shows less impressive results in terms of expression reproduction, due mostly to the difficulties of expression extraction from real time video. It is also relatively expensive in terms of CPU load. However, the bandwidth required for this method is very low.

Lip movement synthesis from speech as promising in terms of generating realistic lip movement synchronized with speech based only on the audio signal, serving as additional visual clue for better speech understanding. In spite of impressive results of non-real time experiments, currently the method can’t be deployed for real-time usage because of the non-real time performance of the speech analysis software currently available. We have therefore adopted a simpler version of this method which provides simple mouth movement, just enough to indicate that a person is speaking.

The use of predefined expressions and emotions is a relatively simple, though quite effective method of conveying expressions and emotions. The user activates them by mouse or keystrokes. This method is very inexpensive in terms of both CPU and bandwidth.

7.1.3 MPEG-4 for NCVEs

We have analyzed the networking requirements of Networked Collaborative Virtual Environments, and how MPEG-4 tools can be used to fulfill these requirements. This analysis is based on our active participation in the development of the MPEG-4 standard which is going on in parallel with our work on NCVEs.

Most of the data types that are important for NCVE systems will be very well supported by MPEG-4 tools (video, audio, 3D objects, textures, bodies, faces). On top of this, MPEG-4 will offer reliable multiplexing, mechanisms for establishing priorities among data, as well as synchronization. We believe that MPEG-4 tools should play an important role in building future Networked Collaborative Virtual Environment systems.

7.2 Potential applications

Networked Collaborative Virtual Environment systems are suitable for numerous collaborative applications ranging from games to medicine [Doenges97], for example:

- Virtual teleconferencing with multimedia object exchange
- All sorts of collaborative work involving 3D design
- Multi-user game environments
- Teleshopping involving 3D models, images, sound (e.g. real estate, furniture, cars)
- Medical applications (distance diagnostics, virtual surgery for training)
- Distance learning/training
- Virtual Studio/Set with Networked Media Integration
- Virtual travel agency

Several experimental applications were developed using the VLNET system. Some snapshots are presented in Figure 36. We present the developed experimental applications in the following subsections.

7.2.1 Entertainment

NCVE systems lend themselves to development of all sorts of multi user games. We had successful demonstrations of chess and other games played between Switzerland and Singapore, as well as between Switzerland and several European countries.

A virtual tennis game has been developed [Noser 96-1] where the user plays against an opponent who is an autonomous virtual human. The referee is also an autonomous virtual human capable of refereeing the game and communicating the points, faults etc. by voice.

Currently a multi user adventure game is under development.

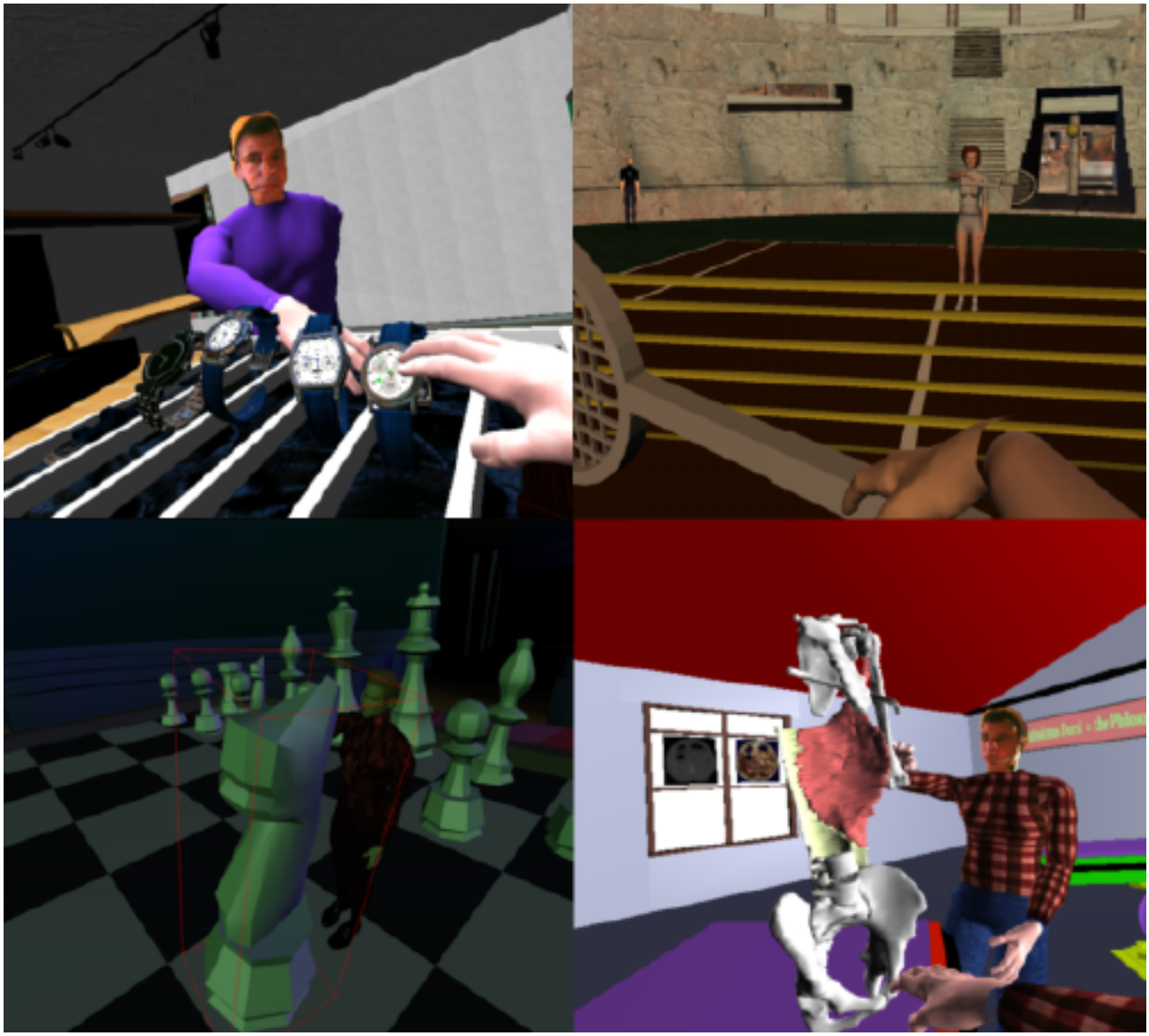


Figure 36: Snapshots from applications

7.2.2 Teleshopping

In collaboration with Chopard, a watch company in Geneva, a teleshopping application was successfully demonstrated between Geneva and Singapore. The users were able to communicate with each other within a rich virtual environment representing a watch gallery with several watches exposed. They could examine the watches together, exchange bracelets, and finally choose a watch.

7.2.3 Medical education

In collaboration with our colleagues working on a medical project we have used the 3D data of human organs, muscles and skeleton reconstructed from MRI images [Beylot96] to build a small application in the field of medical education. The goal is to teach a student the position, properties and function of a particular muscle - the latissimus dorsi. In the virtual classroom several tools are at the disposition of the professor. MRI, CT and anatomical slice images are on the walls showing the slices where the muscle is visible. A 3D reconstruction of the skeleton is available together with the muscle, allowing to examine the shape of the muscle and the points of attachment to the skeleton. Finally, an autonomous virtual human with a simple behavior is there to demonstrate the movements resulting from a contraction of the discussed muscle.

7.2.4 Stock exchange

An application has been developed to visualize in 3D the real time updates of stock exchange data and allow interactions of users with the data (buy, sell) and with each other.

7.3 Future research

We see four domains for future work:

- Improvement of real time VH simulation
- Scaleable VH
- Input techniques for natural communication
- Support of standards

7.3.1 Improvement of real time VH simulation

Not so long ago it was practically impossible to simulate Virtual Humans in real time on a computer. Now it is possible, however considerable computing power is still needed and realism of animation and skin deformation could be improved. For example, currently the hands can not be deformed in our real time model, only animated as a set of rigid finger segments. Work is ongoing in improving the real time animation.

7.3.2 Scaleable VH

No matter how much we improve the algorithms for real time human simulation, simulation of large numbers of humans will bring difficulties by sheer computation load. Therefore it is necessary to provide an extension of the method known as Levels of Detail (LOD) [Rohlf94] to Virtual Humans, allowing graceful degradation in representation quality as a particular VH moves further away from the viewpoint. Thus a crowd in a large distance would be a mere set of dots, and by approaching to a particular person we would see him/her in all detail. This method permits significant reduction of CPU and graphics system overhead, as well as the network bandwidth because we need less data transmissions for the VH we see with less detail. The work on LOD for Virtual Humans is ongoing and has already been partly integrated in the VLNET system.

7.3.3 Input techniques for natural communication

For simple and natural input of facial expressions and gestures, sophisticated non-intrusive input techniques are needed. Ongoing work mostly concentrates on video input and image analysis.

7.3.4 Support of standards

As outlined in chapter 6, currently developed MPEG-4 standard shows promise in terms of supporting NCVE requirements [Pandzic97-2], so one of the future work items might be development of a MPEG-4 compatible NCVE system.

8. References

[Allan89] Allan J.B., Wywill B., Witten I.A., “A Methodology for Direct Manipulation of Polygon Meshes”, *Proc. Computer Graphics International ’89*, Leeds, 1989., pp. 451-469.

[Arnaldi89] Arnaldi B., Dumont G., Hegron G., Magnenat Thalmann N., Thalmann D., “Animation Control with Dynamics”, in *State of the Art in Computer Animation*, Springer, Tokyo, pp. 113-124, 1989.

[Azarbayejani93] Azarbayejani A, Starner T, Horowitz B, Pentland A 'Visually Controlled Graphics' *IEEE Transaction on Pattern Analysis and Machine Intelligence*, June 1993, Vol. 15, No 6, 602-605.

[Badler79] Badler N.I., Smoliar S.W., “Digital Representation of Human Movement”, *ACM Computing Surveys*, March issue, pp. 19-38, 1979.

[Badler82] Badler N.I., Morris M.A., “Modelling Flexible Articulated Objects”, *Proc. ComputerGraphics’82*, Online Conf., pp.305-314, 1982.

[Badler85] Badler N.I., Korein J.D., Korein J.U., Radack G.M., Brotman L.S., “Positioning and Animating Figures in a Task-oriented Environment”, *The Visual Computer*, Vol. 1, No. 4, pp. 212-220, 1986.

[Badler93] Norman I. Badler, Cary B. Phillips, Bonnie Lynn Webber, “Simulating Humans”, *Computer Graphics Animation and Control*, Oxford University Press, 1993

[Barfield93] Barfield, W, Weghorst, S., “The sense of presence within virtual environments: a conceptual framework”, in G. Salvendy & M.J. Smith (Eds.), *Human-computer interaction: Software and hardware interfaces*, Amsterdam: Elsevier, 1993.

[Barfield95] Barfield, W, Zeltzer, D., Sheridan, T., Slater, M., “Presence and performance within virtual environments”, in W. Barfield & T. Furness (Eds.), *Virtual environments and advanced interface design*, New York, Oxford, 473-513, 1995.

[Barr84] Global and Local Deformations of Solid Primitives. *Proc. SIGGRAPH’84*, Computer Graphics 18(3):21-30

[Barrus96] Barrus J. W., Waters R. C., Anderson D. B., "Locales and Beacons: Efficient and Precise Support For Large Multi-User Virtual Environments", *Proceedings of IEEE VRAIS*, 1996.

[Benford95] Benford S., Bowers J., Fahlen L.E., Greenhalgh C., Mariani J., Rodden T., "Networked Virtual Reality and Cooperative Work", *Presence: Teleoperators and Virtual Environments*, Vol. 4, No.4, Fall 1995, pp. 364-386

[Beylot96] Beylot P., Gingins P., Kalra P., Magnenat-Thalmann N., Maurel W., Thalmann D., Fasel J. "3D Interactive Topological Modeling using Visible Human Dataset", *Proceedings of EUROGRAPHICS 96*, Poitiers, France, 1996.

[Birman91] Birman, K., Cooper, R., Gleeson, B., "Programming with process groups: Group and multicast semantics", *Technical Report TR-91-1185*, Deparment of Computer Sciences, University of Cornell

[Blum79] Blum R., "Representing Three-dimensional Objects in Your Computer", *Byte*, May1979, pp. 14-29

[Blumberg95] B.M. Blumberg, T.A. Galyean, "Multi-Level Direction of Autonomous Creatures for Real-Time Virtual Environments", *SIGGRAPH 95, Conference Proceedings*, August 6-11, 1995, ACM Press, pp. 47-54.

[Boulic90] Boulic R., Magnenat-Thalmann N. M.,Thalmann D. "A Global Human Walking Model with Real Time Kinematic Personification", *The Visual Computer*, Vol.6(6),1990.

[Boulic95] Boulic R., Capin T., Huang Z., Kalra P., Lintermann B., Magnenat-Thalmann N., Moccozet L., Molet T., Pandzic I., Saar K., Schmitt A., Shen J., Thalmann D., "The Humanoid Environment for Interactive Animation of Multiple Deformable Human Characters", *Proceedings of Eurographics '95*, 1995.

[Bowers92] Bowers, J.M., "Modelling Awareness and Interaction in Virtual Spaces", *Proc. 5th Multi-G Workshop*, Kista, Stockholm, Sweden, 1992.

[Breen89] Breen D.E., "Choreographing Goal-oriented Motion Using Cost Functions" in Magnenat Thalmann N., Thalmann D. (Eds.) *State of the Art in Computer Animation*, Springer, Tokyo, pp. 141-152, 1989.

[Brotman88] Brotman L.S., Netravali A.N., "Motion Interpolation by Optimal Control", *Proc. SIGGRAPH'88*, Computer Graphics, Voll 22., No. 4., pp. 179-188, 1988.

[Calvert78] Calvert T.W., Chapman J., “Notation of Movement with Computer Assistance”, *Proc. ACM Annual Conf.*, Vol. 2, pp. 731-736, 1978.

[Capin95] Capin T.K., Pandzic I.S., Magnenat-Thalmann N., Thalmann, D., "Virtual Humans for Representing Participants in Immersive Virtual Environments", *Proceedings of FIVE '95*, London, 1995.

[Capin97] Capin T.K., Pandzic I.S., Noser H., Magnenat Thalmann N., Thalmann D. "Virtual Human Representation and Communication in VLNET Networked Virtual Environments", *IEEE Computer Graphics and Applications*, Special Issue on Multimedia Highways, March-April 1997.

[Capin97-1] Capin T.K., Pandzic I.S., Thalmann D., Magnenat Thalmann N. "A Dead-Reckoning Algorithm for Virtual Human Figures", *Proc. VRAIS'97*, IEEE Press, 1997

[Carlsson93] Carlsson C., Hagsand O., "DIVE - a Multi-User Virtual Reality System", *Proceedings of IEEE VRAIS '93*, Seattle, Washington, 1993.

[Chadwick89] Chadwick J., Haumann D.R., Parent R.E., “Layered Construction for Deformable Animated Characters”, *Proc. SIGGRAPH’89*, Computer Graphics, Vol 23, No. 3, pp. 234-243, 1989.

[Cohen98] Cohen M.F., “Gracefulness and Style in Motion Control”, *Proc. Mechanics, Control and Animation of Articulated Figures*, MIT, 1998.

[Doenges97] Doenges P.K., Capin T.K., Lavagetto F., Ostermann J., Pandzic I.S., Petajan E.D. "MPEG-4: Audio/Video & Synthetic Graphics/Audio for Mixed Media", *Image Communication Journal, Special issue on MPEG-4*, 1997. (to appear)

[Durlach95] Durlach N.I., Mavor A.S., eds., “Virtual Reality: Scientific and Technological Challenges”, Committee on Virtual Reality Research and Development, National Research Council, National Academy of Sciences Press, ISBN 0-309-05135-5, 1995.

[Fahlen93] Fahlen, L.E., Brown C.G., Stahl, O., Carlsson, C., “A Space Based Model for User Interaction in Shared Synthetic Environments”, *Proceedings InterCHI’93*, Amsterdam, 1993.

[Fontaine92] Fontaine, G., “The experience of a sense of presence in intercultural and international encounters”, *Presence: Teleoperators and virtual environments*, 1, 482-490, 1992.

[Forsey88] Forsey D., Wilhelms J., “Techniques for Interactive Manipulation of Articulated Bodies using Dynamics Analysis”, *Proc. Graphics Interface’88*, pp. 8-15, 1988.

[Funkhouser95] Funkhouser T.A., "RING: A Client-Server System for Multi-User Virtual Environments", *ACM SIGGRAPH Special Issue on 1995 Symposium on Interactive 3D Graphics*, pp. 85-92, Monterey, CA, 1995

[Funkhouser96] Funkhouser T.A., "Network Topologies for Scaleable Multi-User Virtual Environments", *Proceedings of VRAIS ' 96*, IEEE 1996.

[Girard85] Girard M., Maciejewski A.A., “Computational Modeling for Computer Generation of Legged Figures”, *Proc. SIGGRAPH’85*, Computer Graphics, Vol. 19, No. 3, pp. 263-270, 1985.

[Girard87] Girard M, “Interactive Design of 3D Computer-animated Legged Animal Motion”, *IEEE Computer Graphics and Applications*, Vol. 7, No. 6, pp. 39-51, 1987.

[Gouret89] Gouret J.P., Magnenat Thalmann N., Thalmann D., “Simulation of Object and Human Skin Deformations in a Grasping Task”, *Proc. SIGGRAPH’89*, Computer Graphics, Vol. 23, No. 3, pp. 21-30, 1989.

[Greenhalgh95] Greenhalgh, C., Benford, S., “MASSIVE, A Distributed Virtual Reality System Incorporating Spatial Trading”, *Proceedings of the 15th International Conference on Distributed Computing Systems*, pp 27-34, Los Alamitos, CA, ACM, 1995.

[Gossweiler94] Rich Gossweiler, Robert J. Laferriere, Michael L. Keller, Pausch, "An Introductory Tutorial for Developing Multiuser Virtual Environments", *Presence: Teleoperators and Virtual Environments*, Vol. 3, No. 4, 1994.

[Hagsand91] Hagsand, O., “Consistency and concurrency control in virtual worlds”, *Proceedings of the Second MultiG Workshop*, 1991.

[Heeter92] Heeter, C., “Being there: The subjective experience of presence”, *Presence: Teleoperators and Virtual Environments*, 1, 262-271, 1992.

[Hendrix96] Hendrix C., Barfield W., "Presence within Virtual Environments as a function of Visual Display Parameters", *Presence: Teleoperators and Virtual Environments*, Vol. 5, No. 3, Summer 1996, pp. 274-289.

[IEEE93] Institute of Electrical and Electronics Engineers, International Standard, ANSI/IEEE Standard 1278-1993, Standard for Information Technology, Protocols for Distributed Interactive Simulation, March 1993.

[Kalra92] Kalra P., Mangili A., Magnenat Thalmann N., Thalmann D., "Simulation of Facial Muscle Actions Based on Rational Free Form Deformations", *Proc. Eurographics '92*, pp.59-69., 1992.

[Kalra93] Kalra P. "An Interactive Multimodal Facial Animation System", *PhD Thesis nr. 1183*, EPFL, 1993

[Kato92] Kato M, So I, Hishinuma Y, Nakamura O, Minami T 'Description and Synthesis of Facial Expressions based on Isodensity Maps' in Tosiyasu L (Ed) *Visual Computing*, Springer - Verlag Tokyo 1992, 39-56

[Kessler96] Kessler G.D., Hodges E.F., "A Networked Communication Protocol for Distributed Virtual Environment Systems", *Proceedings of IEEE VRAIS '96*, 1996

[Kishino94] Kishino F, 'Virtual Space Teleconferencing System - Real Time Detection and Reproduction of Human Images" *Proc. Imagina 94*, 109 - 118

[Koenen97] Koenen R., Pereira F., Chiariglione L., "MPEG-4: Context and Objectives", *Image Communication Journal, Special Issue on MPEG-4*, Vol. 9, No. 4, May 1997.

[Komatsu88] Komatsu K., “Human Skin Model Capable of Natural Shape variation”, *The Visual Computer*, Vol. 3, No. 5, pp. 265-271, 1988.

[Korein82] Korein J., Badler N.I., “Techniques for Generating the Goal-directed Motion of Articulated Structures”, *IEEE Computer Graphics and Applications*, Vol. 2, No. 9, pp. 71-81, 1982.

[Lavagetto95] F.Lavagetto, "Converting Speech into Lip Movements: A Multimedia Telephone for Hard of Hearing People", *IEEE Trans. on Rehabilitation Engineering*, Vol.3, N.1, pp.90-102, 1995.

[Lee89] Lee M.W., Kunii T.L., “Animation Design: A database-oriented Animation Design Method with a Video Image Analysis Capability”, in Magnenat-Thalmann N., Thalmann D. (Eds.) *State of the art in Computer Animation*, Springer, Tokyo, pp. 97-112, 1989.

[Li93] Li Haibo, Roivainen P, Forchheimer R '3-D Motion Estimation in Model Based Facial Image Coding' *IEEE Transaction on Pattern Analysis and Machine Intelligence*, June 1993, Vol. 15, No 6, 545-555.

[Leblanc90] Leblanc A., Thalmann D., “Rendering Antialiased Hair Using an Alpha-Blending Method”, *Technical Report, Computer Graphics Lab, Swiss Federal Institute of Technology, 1990.*

[Macedonia94] Macedonia M.R., Zyda M.J., Pratt D.R., Barham P.T., Zestwitz, "NPSNET: A Network Software Architecture for Large-Scale Virtual Environments", *Presence: Teleoperators and Virtual Environments*, Vol. 3, No. 4, 1994.

[Macedonia97] Macedonia M.R., Zyda M.J., “A Taxonomy for Networked Virtual Environments”, *IEEE Multimedia*, Vol. 4, No. 1, pp. 48-56, 1997.

[MagnenatThalmann87] Magnenat Thalmann N., Thalmann D., “The Direction of Synthetic Actors in the film Rendes-vouz a Montreal”, *IEEE Computer Graphics and Applications*, 7(12), pp.9-19, 1987.

[MagnenatThalmann88] Magnenat Thalmann N., Laperriere R., Thalmann D., “Joint-Dependent Local Deformations for Hand Animation and Object Grasping”, *Proc. Graphics Interface’88*, 1988.

[MagnenatThalmann89] Magnenat Thalmann N., Minh H.T., de Angelis M., Thalmann D., “Design, Transformation and Animation of Human Faces”, *The Visual Computer*, Vol. 5, No. 3, pp. 32-39, 1989.

[MagnenatThalmann93] Magnenat Thalmann N, Cazedevals A, Thalmann D 'Modeling Facial Communication Between an Animator and a Synthetic Actor in Real Time' *Proc Modeling in Computer Graphics*, Genova, Italy, June 1993 (Eds Falcidieno B and Kunii L), 387-396.

[MagnenatThalmann93-1] Magnenat Thalmann N., Thalmann D., “The Artificial Life of Synthetic Actors”, IEICE Transactions, J76-D-II, 8, pp1506-1514, 1993.

[MagnoCaldognetto89] Magno Caldognetto E, Vagges K, Borghese N A, Ferrigno G 'Automatic Analysis of Lips and Jaw Kinematics in VCV Sequences' *Proceedings of Eurospeech 89 Conference* vol. 2, 453 - 456

[Mase90] Mase K, Pentland A 'Automatic Lipreading by Computer' *Trans. Inst. Elec. Info. and Comm. Eng.* 1990, Vol. J73-D-II, No. 6, 796-803

[Moccozet97] Moccozet L., Magnenat Thalmann N., “Dirichlet Free Form Deformations and their Application to Hand Deformation”, *Proc. Computer Animation’97*, pp. 93-102, 1997.

[Moccozet97-1] Moccozet L., Huang Z., Magnenat Thalmann N., Thalmann D., “Virtual Hand Interactions with 3D World”, *Proc. Multimedia Modeling ’97*, Singapore, 1997.

[Molet96] Molet T., Boulic R., Thalmann D., "A Real Time Anatomical Converter for Human Motion Capture", *Proc. of Eurographics Workshop on Computer Animation and Simulation*, 1996.

[MPEG-N1886] "MPEG-4 Requirements version 5", ISO/IEC JTC1/SC29/WG11 N1886, MPEG97/November 1997.

[MPEG-N1901] "Text for CD 14496-1 Systems", ISO/IEC JTC1/SC29/WG11 N1886, MPEG97/November 1997.

[MPEG-N1902] "Text for CD 14496-2 Video", ISO/IEC JTC1/SC29/WG11 N1886, MPEG97/November 1997.

[Noser96] H. Noser, D. Thalmann, “The Animation of Autonomous Actors Based on Production Rules”, *Proceedings Computer Animation' 96*, June 3-4, 1996, Geneva Switzerland, IEEE Computer Society Press, Los Alamitos, California, pp 47-57

[Noser96] H. Noser, I. S. Pandzic, T. K. Capin, N. Magnenat Thalmann, D. Thalmann, "Playing Games through the Virtual Life Network", *Proceedings of Artificial Life V*, Nara, Japan, 1996.

[Ohya95] Ohya J., Kitamura Y., Kishino F., Terashima N., "Virtual Space Teleconferencing: Real-Time Reproduction of 3D Human Images", *Journal of Visual Communication and Image Representation*, Vol. 6, No. 1, pp. 1-25, 1995.

[Pandzic94] Pandzic I.S., Kalra P., Magnenat-Thalmann N., Thalmann D., "Real-Time Facial Interaction", *Displays*, Vol. 15, No 3, 1994.

[Pandzic96] I.S. Pandzic, T.K. Capin, N. Magnenat Thalmann, D. Thalmann, "Motor functions in the VLNET Body-Centered Networked Virtual Environment", *Proc. of 3rd Eurographics Workshop on Virtual Environments*, Monte Carlo, 1996.

[Pandzic96-1] Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D. "Towards Natural Communication in Networked Collaborative Virtual Environments", *Proc. FIVE 96*, Pisa, Italy, 1996.

[Pandzic97] Pandzic I.S., Capin T.K., Lee E., Magnenat Thalmann N., Thalmann D., "A flexible architecture for Virtual Humans in Networked Collaborative Virtual Environments", *Proceedings Eurographics 97*

[Pandzic97-1] Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D., "A Versatile Navigation Interface for Virtual Humans in Networked Collaborative Virtual Environments", *Proceedings ACM Symposium on Virtual Reality Software and Technology* , Lausanne, 1997

[Pandzic97-2] Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D., "MPEG-4 for Networked Collaborative Virtual Environments", *Proceedings International Conference on Virtual Systems and Multimedia*, IEEE, Geneva, 1997.

[Pandzic97-3] Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D., "VLNET: A Body-Centered Networked Virtual Environment", *Presence: Teleoperators and Virtual Environments*, Vol. 6, No. 6, 1997.

[Parke82] Parke F.I., "Parametrized Models for Facial Animation", *IEEE Computer Graphics and Applications*, Vol. 2, No. 9, pp. 61-68, 1982.

[Patterson91] Patterson E C, Litwinowich P C, Greene N 'Facial Animation by Spatial Mapping', *Proc. Computer Animation 91*, Magnenat Thalmann N, Thalmann D (Eds.), Springer-Verlag, 31 - 44

[Pearce86] Pearce A., Wywill B., Wywill G., Hill D., "Speech and Expression: a Computer Solution to Face Animation", *Proc. Graphics Interface '86*, pp. 136-140, 1986.

[Pratt97] Pratt D.R., Pratt S.M., Barham R.E., Waldrop M.S., Ehlert J.F., Chrislip C.A. "Humans in Large-scale, Networked Virtual Environment", Presence: Teleoperators and Virtual Environments, Vol. 6, No. 5, pp. 547-564, 1997.

[Renault90] Renault O., Magnenat Thalmann N., Thalmann D., "A Vision-based Approach to Behavioral Animation", *Visualization and Computer Animation Journal*, John Wiley, Vol. 1, No. 1, 1990.

[Reynolds87] Reynolds C (1987), *Flocks, Herds, and Schools: A Distributed Behavioral Model*,Proc. SIGGRAPH 1987, Computer Graphics, Vol.21, No4, pp.25-34

[Rohlf94] Rohlf J., Helman J., "IRIS Performer: A High Performance Multiprocessing Toolkit for Real-Time 3D Graphics", *Proc. SIGGRAPH' 94*,1994.

[Saji92] Saji H, Hioki H, Shinagawa Y, Yoshida K, Kunii T 'Extraction of 3D Shapes from the Moving Human Face using Lighting Switch Photometry' in Magnenat Thalmann N, Thalmann D (Eds) *Creating and Animating the Virtual World*, Springer-Verlag Tokyo 1992, 69-86

[Schroeder88] Schroeder P, Zeltzer D., "Pathplanning inside Bolio", in D. Thalmann (Ed.) *Synthetic Actors: The Impact of Artificial Intelligence and Robotics on Animation*, Course Notes SIGGRAPH'88, pp. 194-207, 1988.

[Sederberg86] Sederberg T.W., Parry S.R. , "Free Form Deformation of solid geometric models", *Proc. SIGGRAPH'86*, Computer Graphics, Vol. 20, No. 4, pp. 151-160, 1986.

[Shaw93] Shaw, C., Green, M., "The MR Toolkit Peers Package and Experiment", *Proc. IEEE Virtual Reality Annual International Symposium*, pp 463-469, 1993.

[Sheridan92] Sheridan, T.B., "Musings on telepresence and virtual presence", *Presence: Teleoperators and virtual environments*, 1, 120-126

[Singh95] Singh G., Serra L., Png W., Wong A., Ng H., "BrickNet: Sharing Object Behaviors on the Net", *Proceedings of IEEE VRAIS ' 95*,1995.

[Slater94] Slater M., Usoh M. "Body Centered Interaction in Immersive Virtual Environments", in *Artificial Life and Virtual Reality*, N. Magnenat Thalmann, D. Thalmann, eds., John Wiley, pp 1-10, 1994.

[Slater94-1] Slater M., Usoh M., Steed A. "Depth of presence in virtual environments", *Presence: Teleoperators and Virtual Environments*, 3 (2), 130-140, 1994.

[Smith83] Smith, A.R., "Digital Filmmaking, *Abacus*, Vol 1., No. 1., pp. 28-45

[Stansfield95] Stansfield S., Shawver D., Miner N., Rogers D., "An Application of Shared Virtual Reality to Situational Training", *Proceedings of IEEE VRAIS ' 95*,1995.

[Steuer92] Steuer, J., "Defining Virtual Reality: Dimensions determining telepresence", *Journal of Communication*,42, 73-93, 1992.

[Thalmann96] D. Thalmann, J. Shen, E. Chauvineau, "Fast Realistic Human Body Deformations for Animation and VR Applications", *Proc. Computer Graphics International '96*, Pohang, Korea,1996.

[Terzopolous91] Terzopoulos D, Waters K 'Techniques for Realistic Facial Modeling and Animation' *Proc. Computer Animation* 1991, Geneva, Switzerland, Springer - Verlag, Tokyo, 59 - 74

[Tromp95] Tromp J.G., "Presence, Telepresence and Immersion: The Cognitive Factors of Embodiments and Interaction in Virtual Environments", *Proceedings of FIVE '95*, London, 1995.

[Volino95] Volino P., Courchesne M., Magnenat Thalmann N., "Versatile Efficient Techniques of Simulating Cloth and other Deformable Objects", *Proc. SIGGRAPH'95*, pp. 137-144, 1995.

[VRML97] The Virtual Reality Modeling Language, ISO/IEC DIS 14772-1, April 1997.

[Watanabe89] Watanabe Y, Suenega Y, "Drawing Human Hair Using Wisp Model", *Proc. Computer Graphics International*, Springer, Tokyo, pp. 691-700, 1989.

[Waters87] Waters K., "A Muscle Model for Animating Three-Dimensional Facial Expression", *Proc. SIGGRAPH'87*, Vol. 21, No. 4, pp. 17-24, 1987.

[Waters 91] Waters K, Terzopoulos D 'Modeling and Animating Faces using Scanned Data' *Journal of Visualization and Computer Animation* 1991, Vol. 2, No. 4, 123-128

[Waters96] Waters R.C., "Time Synchronization In the Spline Scaleable Platform for Interactive Environments", *Proceedings MMM'96*, Singapore, 1996.

[Welch96] Welch R.B., Blackmon T.T., Liu A., Mellers B.A., Stark L.W., "The Effects of Pictorial Realism, Delay of Visual Feedback, and observer Interactivity on the Subjective Sense of Presence", *Presence: Teleoperators and Virtual Environments*, Vol. 5, No. 3, Summer 1996, pp. 263-273

[Wilhelms87] Wilhelms J., "Using Dynamic Analysis for Realistic Animation of Articulated Bodies", *IEEE Computer Graphics and Applications*, Vol. 7, No. 6, pp.12-27, 1987.

[Zeltzer82] D. Zeltzer, "Motor Control Techniques for Figure Animation", *IEEE Computer Graphics and Applications*, 2 (9), 53-59, 1982

[Zeltzer92] Zeltzer, D., "Autonomy, interaction and presence", *Presence: Teleoperators and virtual environments,1, 127-132, 1992.*

[Zyda93] Zyda, M. J., Pratt, D. R., Falby, J. S., Barham, P., Kelleher, K. M., "NPSNET and the Naval Postgraduate School Graphics and Video Laboratory," Presence, Vol. 2, No. 3., pp. 244-258, 1993.

[Zyda97] Zyda, M., Sheehan, J., eds., “Modeling and Simulation: Linking Entertainment and Defense”, ISBN 0-309-05842-2, National Academy Press, 1997.

9. Lists of figures and tables

List of figures

List of tables

10. Related publications by the author

Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D., "VLNET: A Body-Centered Networked Virtual Environment", *Presence: Teleoperators and Virtual Environments*, Vol. 6, No. 6, 1997.

Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D., "A Versatile Navigation Interface for Virtual Humans in Networked Collaborative Virtual Environments", *Proceedings ACM Symposium on Virtual Reality Software and Technology* , Lausanne, 1997.

Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D., "MPEG-4 for Networked Collaborative Virtual Environments", *Proceedings International Conference on Virtual Systems and Multimedia*, IEEE, Geneva, 1997.

Pandzic I.S., Capin T.K., Lee E., Magnenat Thalmann N., Thalmann D., "A flexible architecture for Virtual Humans in Networked Collaborative Virtual Environments", *Proceedings Eurographics'97*, 1997.

Capin T.K., Pandzic I.S., Noser H., Magnenat Thalmann N., Thalmann D. "Virtual Human Representation and Communication in VLNET Networked Virtual Environments", *IEEE Computer Graphics and Applications*, Special Issue on Multimedia Highways, March-April 1997.

Doenges P.K., Capin T.K., Lavagetto F., Ostermann J., Pandzic I.S., Petajan E.D. "MPEG-4: Audio/Video & Synthetic Graphics/Audio for Mixed Media", *Image Communication Journal, Special issue on MPEG-4*, 1997. (to appear)

Capin T.K., Pandzic I.S., Thalmann D., Magnenat Thalmann N. "A Dead-Reckoning Algorithm for Virtual Human Figures", *Proc. VRAIS' 97*, IEEE Press, 1997

Thalmann D., Babski C., Capin T.K., Magnenat Thalmann N., Pandzic I.S., "Sharing VLNET worlds on the WEB", *Proceedings of Compugraphics' 96*, 1996.

Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D. "Towards Natural Communication in Networked Collaborative Virtual Environments", *Proc. FIVE 96*, Pisa, Italy, 1996.

Noser H., Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D. "Playing Games through the Virtual Life Network", *Proceedings of Artificial Life V*, Nara, Japan, 1996.

Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D., "Motor functions in the VLNET Body-Centered Networked Virtual Environment", *Proc. of 3rd Eurographics Workshop on Virtual Environments*, Monte Carlo, 1996.

Magnenat Thalmann N., Kalra P., Pandzic I.S., "Direct Face To Face Communication Between Real And Virtual Humans", *International Journal of Information Technology*, Vol. 1, No. 2, pp. 145-157 (World Scientific Publishing), 1995.

Thalmann D., Capin T.K., Pandzic I.S., Magnenat Thalmann N., "Participant, User-Guided and Autonomous Actors in the Virtual Life Network VLNET", *Proc. of International Conference on Artificial Reality and Tele-Existence '95, Conference on Virtual Reality Software and Technology '95 ICAT/VRST'95* (ACM SIGCHI press), Chiba, Japan, 1995.

Pandzic I.S., Capin T.K., Magnenat Thalmann N., Thalmann D., "VLNET: A Networked Multimedia 3D Environment with Virtual Humans", *Proc. Multi-Media Modeling MMM`95* (World Scientific Press)(ISBN 981-02-2502-4), Singapore, 1995.

Capin T.K., Pandzic I.S., Magnenat-Thalmann N., Thalmann, D., "Virtual Humans for Representing Participants in Immersive Virtual Environments", *Proceedings of FIVE '95*, London, 1995.

Kalra P., Magnenat-Thalmann N., Pandzic I.S., "Facial Interaction for Human Machine Interface", in *Human Comfort and Security of Information Systems*, K. Varghese, S. Pfleger (Eds.) (Springer)(ISBN 3-540-62067-2), 1997 - Proceedings of Human Comfort and Security Workshop, Brussels, Belgium, 1995.

Pandzic I.S., Magnenat-Thalmann N., Roethlisberger M., "Parallel Raytracing on the IBM SP2 and CRAY T3D", *EPFL Supercomputing Review*, No 7, 1995.

Boulic R., Capin T., Huang Z., Kalra P., Lintermann B., Magnenat-Thalmann N., Moccozet L., Molet T., Pandzic I., Saar K., Schmitt A., Shen J., Thalmann D., "The Humanoid Environment for Interactive Animation of Multiple Deformable Human Characters", *Proceedings of Eurographics '95*, 1995.

Pandzic I.S., Kalra P., Magnenat-Thalmann N., Thalmann D., "Real-Time Facial Interaction", *Displays*, Vol. 15, No 3, 1994.