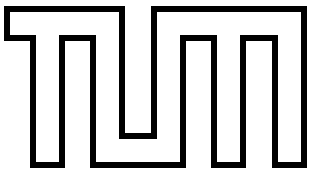
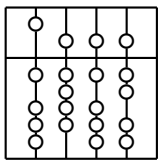


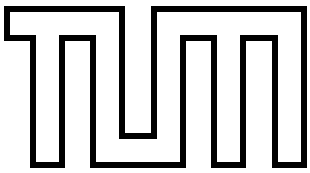
Technische Universität
München
Fakultät für Informatik



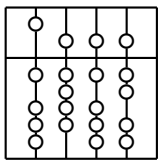
Diplomarbeit

Importance of Gaze Awareness in Augmented Reality Teleconferencing

Michael Siggelkow



Technische Universität
München
Fakultät für Informatik



Diplomarbeit

Importance of Gaze Awareness in Augmented Reality Teleconferencing

Michael Siggelkow

Aufgabensteller: Univ-Prof. Gudrun Klinker, Ph.D.

Betreuer: Dr. Martin Wagner

Abgabedatum: 11. Mai 2005

Erklärung

Ich versichere, dass ich diese Ausarbeitung der Diplomarbeit selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 11. Mai 2005 Michael Siggelkow

Zusammenfassung

Kommunikation ist heutzutage erforderlich und erwünscht. Da das Telefon keine Blickrichtung oder Gestik von Personen übertragen kann und somit auch keine soziale Nähe des Gesprächspartners vermittelt, wird die Videotelefonie oft als eine mögliche Lösung dieses Problems angesehen. Sie ist jedoch mit einigen Nachteilen verbunden. Für eine flüssige Unterhaltung in einer Konferenzschaltung ist es wichtig, dass die Teilnehmer wissen, wer in welche Richtung blickt und worüber jeder Einzelne spricht.

Ich beschreibe in dieser Arbeit eine neue Technik, die Benutzern bewusst machen soll, wer wen anblickt und was die Benutzer auf einem gemeinsamen Arbeitsbereich betrachten. Augmented Reality (AR, Erweiterte Realität) ermöglicht es, virtuelle Objekte in die reale Welt einzublenden und kann somit die Gesprächspartner als virtuelle Avatare in der wirklichen Welt darstellen. Gleichzeitig wird die Blickrichtung der Benutzer erfasst und durch Drehungen der Avatare zueinander sowie Pfeile auf dem gemeinsamen Arbeitsbereich visualisiert. Jeder Benutzer kann seinen Arbeitsplatz und die Positionierung der Avatare frei gestalten. So können die Benutzer in ihrer gewohnten Umgebung weiterarbeiten und müssen sich nicht an eine neue Umgebung anpassen, wie dies bei bisherigen Ausführungen zu diesem Thema der Fall ist.

Mit Hilfe von 30 Probanden wurde eine Benutzerstudie durchgeführt, um die Akzeptanz und Bedienbarkeit der Anwendung zu untersuchen. Die Ergebnisse zeigen eine erhöhte Zufriedenheit der Benutzer mit dem Ablauf der Zusammenarbeit und eine gesteigerte Wahrnehmung der Partner bei dieser neuen Form der Kommunikationsunterstützung. Besonders die Pfeile auf dem gemeinsamen Arbeitsplatz waren sehr hilfreich. Die Avatare ermöglichten die Zuordnung der Pfeile zur jeweiligen Person.

Abstract

Today communication is required and desired. Since telephony cannot convey spatial cues such as gaze or gestures and so provide a kind of social presence, video conferencing is often seen as a possible solution for this problem. However it suffers from several disadvantages. It is important for a fluent conversation in teleconferences that users are aware of where the remote participants are looking at and about what they are talking.

In this thesis I describe a new approach to maintain gaze awareness between users and on a shared workspace, that means every user knows about the gaze direction and the point the other participants are looking at. Augmented Reality (AR) enables users to see virtual objects placed in the real world and so used to display remote persons as virtual avatars in the genuine world and track their gaze at the same time. The captured gaze is reflected in the rotations of the avatars towards each other and with help of arrows on the shared workspace. Users can set up their workspace independently including the free positioning of the avatars. So they do not have to adopt to a new environment - as it is required in other systems - but can go on working in their common workspace.

To evaluate the usability and acceptance of the application a user study with thirty subjects was conducted. The results show an increased satisfaction of the users with the collaboration session and improved awareness. Especially the arrows on the shared workspace helped the users a lot. The avatars were very important to get a mapping which arrow on the shared workspace belongs to which person.

Acknowledgements

This work would have never been possible without the help and assistance of many people who supported me with my work or made the stay in New Zealand so enjoyable.

First I have to thank Felix Löw. He encouraged me during hard times of the work, advised me to concentrate on the real important things to not get lost in some details and just being there whenever I needed someone to talk.

Secondly I want to thank Martin Wagner for his help, support and criticism from the beginning of the work until the end. Without you the work would be absolutely diffuse.

I also thank Gudrun Klinker for giving me the chance to work for my thesis in New Zealand.

A very special thank you goes to Mark Billinghurst. It was a pleasure to work with you and be inspired by your ideas. I also want to thank all the people at the HitLabNZ. It was so great to work with you and to experience the spirit of the HitLabNZ. Especially thank you to Raphaël Grasset that you were always available to me when there were unsolvable problems and your great ideas imbuing my work. Thanks as well to Jörg Hauber, Thomas Zurbrügg, Anna-Lee Mason, Philip Lamb, Claudia Nelles, Nathan Gardiner and all the others.

Thank you Michael Herchel for your first level support and all the fun we had. You really made work more joyful.

Thanks to the Canterbury University Tramping Club members who brought me to great experiences in the beautiful New Zealand nature, especially Steve Pawson. Greetings to all the soccer guys and thank you to Jörg Hauber for introducing me to both groups.

Two guys I will never forget are Johan Karlsson and Mikael Selegård. Thank you guys for such a great time. It was just perfect.

Thanks as well to my sister and friends who brought a bit of home to me by their visit in New Zealand: Katrin Siggelkow, Kerstin Kunoth, Christian Göbel, Carola Kühn, Claudia Kühn and Christoph Maurer.

A huge thanks goes to my sister Katrin for her self-sacrificing dedication prove-reading this thesis. Nobody could understand this work without your precious review!

Almost last but the most I want to thank my parents who enabled me to study and always stood behind me the last years. I would have never succeeded without you. *Danke!*

My stay in New Zealand was financially supported by the Deutscher Akademischer Austausch Dienst (DAAD).

Contents

CHAPTER 1

Introduction

In this first chapter I motivate the need of awareness in remote conferencing and introduce the fields of research in computer science *Computer Supported Cooperative Work*, *Human-Computer Interaction* and *Augmented Reality* , how they are in relationship with this work and how they can contribute to the idea of awareness.

1.1 Motivation

Todays technology allows people to communicate to everyone at almost every time and every place. This is not only a trend to satisfy human wishes it is also requested by the industry.

People like to communicate, they want to tell their families and friends about their feelings and experiences. The telephone was one of the greatest if not the greatest invention of the 19th century. It allowed people to talk about great distances. Of course the telephone became really popular in the last century since almost every household got its own telephone mainline and prices got down. So it was easy for everyone to reach family and friends. With the wide spreading of mobile telephony the need to keep in contact at every time even increased.

Another factor that requires more and more communication is the globalisation of today's industry. Parts of the development or manufacturing are either outsourced to other countries to save production costs or to customise products better to needs on site. This development requires extra effort to coordinate work. This coordination can be supported by communication systems and world-wide networking infrastructure. There are systems out that cover the range from offering support during face-to-face meetings via project management tools to teleconferences. All these systems claim to support users and bring them together over the distance. Especially video conferencing lives through a real hype even in private life.

But of course such systems cannot supersede real conversation since certain social cues get lost. Short, Williams and Christie [SWC76] introduced the theory of *Social Presence* as a "quality of the medium" that users use to communicate. So it should be the aim of researchers and developers to focus on these aspects and find new ways to increase this *Social*

Presence.

Human-Computer Interaction focuses on the user-friendly interaction techniques between the humans and the computer. Whereas *Computer Supported Cooperative Work* concentrates on possibilities to support groups of users in their work life. In this context *Augmented Reality* can provide new ways of interaction. How this fields can come together and what they actually mean I will introduce in the rest of this chapter and chapter 3.

1.2 Computer Supported Cooperative Work

Computer Supported Cooperative Work (CSCW) covers a huge field of different research. As the name implies CSCW searches for possibilities and ways how computer systems can support humans in their day-to-day professional life. This covers as well research investigating workflows and activities in companies and how colleagues work together. Wilson [Wil91] defines:

> CSCW [is] a generic term, which combines the understanding of the way people work in groups with the enabling technologies of computer networking, and associated hardware, software, services and techniques.

Sometimes you also find the term *Computer Supported Collaborative Work* for the acronym CSCW but there is still disagreement if collaboration and cooperation have the same meaning. Dillenbourg et al. [DBBO96] define the difference as follows:

> Cooperation and collaboration do not differ in terms of whether or not the task is distributed, but by virtue of the way in which it is divided: in cooperation, the task is split (hierarchically) into independent subtasks; in collaboration, cognitive processes may be (heterarchically) divided into intertwined layers. In cooperation, coordination is only required when assembling partial results, while collaboration is [...] a coordinated, synchronous activity that is the result of a continued attempt to construct and maintain a shared conception of a problem

Often *groupware* is used as a synonym for CSCW. But groupware describes more the sum of applications and tools that support groups to work together. Whereas CSCW also includes the research on psychological, social, and organisational effects.

In the rest of this work I inherit the definition of Wilson from above for CSCW as *Computer Supported Cooperative Work* .

CSCW systems can be classified in different ways. An intuitive way is the separation in the two dimensions *time* and *space*. Dix et al. devide in [DFAB97] non computer supported work as you can see in table 1.1. Baecker [BGBG95] uses the time/space taxonomy as well to distinguish the different types of support by CSCW as you can see in table 1.2.

	Same place	Different place
Same time	Face-to-face conversation	Telephone
Different time	Post-it note	Letter

Table 1.1: Basic time/space matrix

	One meeting site (same places)	Multiple meeting sites (different places)
Synchronous communication (same time)	Face-to-Face Interactions	Remote Interactions
Asynchronous communication (different time)	Ongoing Tasks	Communication and Coordination

Table 1.2: CSCW time/space matrix

Public computer displays, electronic meeting rooms or group decision support systems are examples for *face-to-face interactions*. Imagine a meeting room where every user has her own computer with a private display and there is one big public display for the whole room. In a brainstorming session every user can collect ideas privately and then post them anonymously to the public display. So no one has to be afraid of not being respected. In a next step a group decision system could help to lead to a structured and fair way to find a final product idea, for example.

Possible *remote interactions* are shared view desktop conferencing systems, desktop conferencing with collaborative editors, video conferencing or media spaces. Since this is the main scope of this work more detailed information on this area can be found in section 1.2.1.

To the category *Ongoing Tasks* you can add team rooms, group displays, shift work groupware and project management. Team rooms are a kind of repository where user can store documents and various data. They are mostly web-based so they are accessible all the time from everywhere. In [RG96] Rosemann and Greenberg present an application that combines conferencing and repositories and so brings groups co-located or at a distance closer together.

In the *Communication and Coordination* class all systems are combined that support groups located at different places and at different time. These are email, asynchronous conferencing bulletin boards, structured messaging systems, workflow management, version control, meeting schedulers, cooperative hypertext and organisational memory. Most of these techniques as email and bulletin boards are commonly used today not only any more in business but in private life as well. Almost every company uses some kind of meeting scheduler like Microsoft Outlook [1] or Lotus Notes[2] today. In development departments version control systems are a naturally used service.

1.2.1 Remote Conferencing

As mentioned above remote conferencing is part of the synchronous collaborations and commonly at different locations. Typically this includes tele- and video conferences. But techniques as group editors and application sharing belong to this category as well. Group editors facilitate the concurrent working of multiple persons on one document whereas application sharing often just allows users to watch a master user working with an application.

You have to separate three different kinds of video conferencing.

Media spaces or informal video conferences: Media spaces allow users at different locations

[1] http://office.microsoft.com/outlook
[2] http://www.lotus.com/notes

to get in contact informally. This means that always-on video and audio connections are available. The first attempt was at Xerox where a developer group was split up to Palo Alto and Portland. On every site a video wall was installed next to the coffee machine and both where connected with a video/audio link.

Special video conference rooms: In companies certain rooms are equipped with multiple cameras and big video screens to allow groups to take part in video conferences.

Desktop video conferencing: All participants have their own web cam at their workspace and see the other users' video streams in windows on their screen. This technique becomes more and more popular even for the private use, because cameras are cheap and many instant messenger as Microsofts MSN Messenger [3] or ICQ [4] support video conferencing.

Advantages and Disadvantages

Video conferences benefit on the possibility to convey facial expressions and gestures of the user [IT93]. That helps because in face-to-face a lot of non-verbal cues are supporting the conversation. This can be nodding or shaking one's head.

The easy set-up and the cheap equipment of desktop video conference is a virtue but also introduces problems. First the video quality is not very good and the size of the video picture is so small that you can't actually recognise details such as nodding with the head for example. Often jitter and latency in the network connection worsen the comfort in a video conference as well.

Another problem is the positioning of the camera. Since the camera is normally placed on top or below the screen you never get direct eye contact (see figure 1.1). In more complex set-ups this problem is solved by special half-transparent mirrors as you can see in figure 1.2. Tsai et al. instead present in [TKHS04] how they can preserve eye contact by using two precalibrated cameras and morphing one image out of the two.

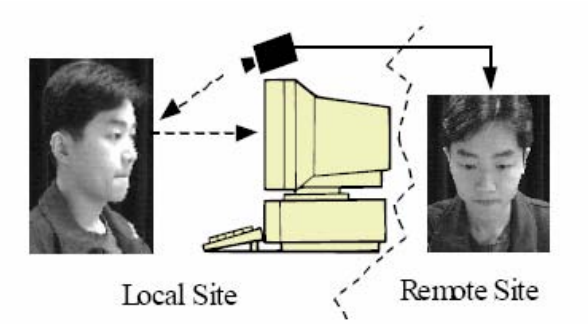


Figure 1.1: Problem of camera positioning and lost eye contact in desktop video conferences (taken from [TKHS04])

In special designed video conference rooms the problem with the small video images is solved (see figure 1.3). Such rooms have a better network connection or sometimes even rely on network technology that supports quality of service (QoS) . Better cameras are used, too,

[3] http://messenger.msn.com/
[4] http://www.icq.com

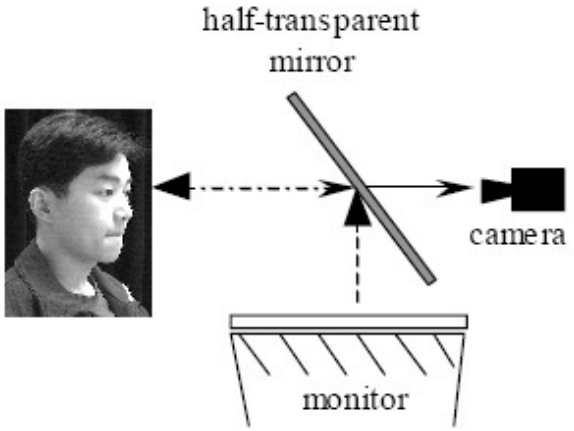


Figure 1.2: Example setup with a half-transparent mirror to preserve eye contact (taken from [TKHS04])

what introduces new problems. Because of the high costs to equip such a room companies can probably only afford one room. More effort to manage the access to this room is required. So you need schedules and the room is not always free if needed. Due to the high network load the ongoing costs are high.

Figure 1.3: Example of a video conferencing room (taken from [Int])

As the distance from the video display and camera to the users is higher than at desktop video conferences you can probably place the camera better but you still cannot establish a real eye contact in such a set-up. So a single user cannot recognise at whom the others a looking or if someone is gazing at her. But this is especially very important in conferences with many participants because this eye contact and *awareness* where everyone is looking normally controls the floor passing and turn taking in face-to-face meetings.

Since cameras have a much smaller field of view compared with the human eye you always just see a small clipping of the remote room. To overcome that problem and enable a better impression of the remote side there can be multiple cameras installed in one room so

you get a better feeling but therefore another display is needed and network load increases again.

1.2.2 Awareness

Dourish and Bellotti give following definition for awareness [DB92]:

> Awareness is an understanding of the activities of others to provide a context for your own activities.

That means, with the knowledge what is happening around you, your work improves and fits better in the whole group. In the context of CSCW this is often called *group awareness*. Examples can be found in group editors where paragraphs are highlighted with a special colour to signal the user that someone else is working on this part of the document. In the earlier mentioned team rooms and similar systems a separate module could provide awareness. Imagine multiple user are working from time to time on the same document. When a user logs onto the system a message appears telling which other user have changed the document since the last access. Other notification ways could be email and SMS. Often user can apply different filter where they can define certain threshold value for distinct notification levels.

For synchronous conferencing the users need certain awareness information in real-time that get lost through the application computer systems. Gaze is an important cue that conveys several information as

Direction: At whom or what is someone paying attention or listening?

Attention: How attentively is someone listening to a conversation?

Gestures are suppressed by media too. Especially pointing devices are missing in standard video conferences. In application sharing systems telepointers are common tools today but they bring new problems, too. Either all users have to share one telepointer so they need additional conversation to control the floor passing this is regulating who is allowed to use the telepointer. Or every user has his own telepointer. This can lead to complexity because you need a mapping between the telepointers and the users behind.

The absence of these important interpersonal cues obstructs the acceptance and usability of conferencing systems.

In [GG02] Gutwin and Greenberg define the term *workspace awareness* as the *"understanding of another person's interaction with a shared workspace"*. The essential questions for workspace awareness among others are *Who is working?, Where are they working?* and *What are they doing?*. There will be a more detailed discussion in section 2.1.

As you can see by means of these few examples the term *awareness* is used in different meanings. Schmidt summarises in [Sch02] several used terms and draws the conclusion that you have to provide the specific information of what you are aware: *"awareness of what?"*.

In this work I focus on two kinds of gaze awareness. The first is between the users: *"Who is looking at whom?"*. The later is on the shared workspace: *"Who is looking where?"*

1.3 Human Computer Interaction

1.3.1 General

In a Curricula for Human-Computer Interaction of the ACM SIGCHI [HBC+92] the following definition is proposed:

> Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.

This definition makes clear that *Human-Computer Interaction* is not just a research area of computer science but an interdisciplinary topic touching computer science, psychology, sociology, anthropology and industrial design. It does not just mean the way persons are working with a personal computer it is more the way how they get in contact with any kind of electronic device - from the video recorder to the washing machine. But it includes as well the interaction between humans mediated by computers and that is exactly what this work will target. So the focus is more on computer mediated communication than real human computer interaction. But pointing with the gaze and not with a mouse is not a commonly used interaction technique.

1.3.2 Past - Present - Future

From a historical view the development of the text based towards the graphical user interfaces (GUI) controlled by devices other than the keyboard - the mouse for example - is one of the first steps of ergonomic aspects in computer interaction. Xerox Palo Alto Research Center (PARC) developed the Star as an experimental personal computer providing a GUI with icons and a dynamic menu [SIKH86]. Apple's LISA was the first commercial personal computer with a GUI that was inspired by Xerox's Star.

The concept of WYSIWYG ("What you see is what you get") is also a step towards user friendly applications. Users can see how the result will look like printed while they are working, exactly as we are working everyday with almost all our applications in office use.

Hypertext and the web is part of *Human-Computer Interaction* as well. With help of a browser you can easily and user-friendly navigate through worldwide spread information resources just by clicking.

Other more advanced interaction techniques available today are gesture or speech recognition. Still there is a lot of development in these areas but there are already commercial products like any handheld computer or tablet PC on which you can write with a pen. New mobile phones offer you the possibility to dial contacts from your phone book or even execute commands by speech input.

More still visionary research is going on at the Tangible Media Group at the MIT Media Lab. Under the leadership of Hiroshi Ishii new techniques to interact with the computer are developed. This is the area of Tangible User Interfaces (TUI).

> Tangible User Interface can be defined as an interface, which places a greater emphasis on touch and physical environment or its element between human and digital information. It is a broad aspect Human-Computer Interaction that includes haptic interface or tactile interface.[5]

[5] http://en.wikipedia.org

One example is the sensetable [PIHP01] where user can interact by just moving small pucks on a table (see figure 1.4). In [BID98] even a remote conferencing device is presented where user can move bricks on the local table and the same movements are done on the remote table.

Figure 1.4: Example of the sensetable with a chemistry application (taken from [PIHP01])

1.4 Augmented Reality

Augmented Reality (AR) enhances the reality with additional - in the majority of all cases visual - information. Audio output would be another alternative. In contrast to *Virtual Reality* (VR) where users immerse in a totally computer generated world in *Augmented Reality* only virtual objects are added to the real world. Azuma [Azu95] claims three criteria for *Augmented Reality* :

- Combines real and virtual
- Interactive in real time
- Registered in three dimensional (3D) space

Since in this work only the visual part of *Augmented Reality* is relevant we discuss in the following only topics that are relevant for visual output. Whereas some concepts may be valid for other output technologies as well.

AR-Applications

Azuma [Azu95] also presentes several possible application fields for *Augmented Reality* . In medical AR applications surgeons are supported by AR. They can see for example earlier Computed Tomography (CT) scans overlaid on the real patient and so look into the body of a person without really opening it. That allows a smaller transection and less harmed tissue that leads to a shorter healing process (see figure 1.5).

Another possible application scenario is the area of maintenance, repair and assembly. For example hairline cracks in airplanes can be found by X-Ray scans and with help of

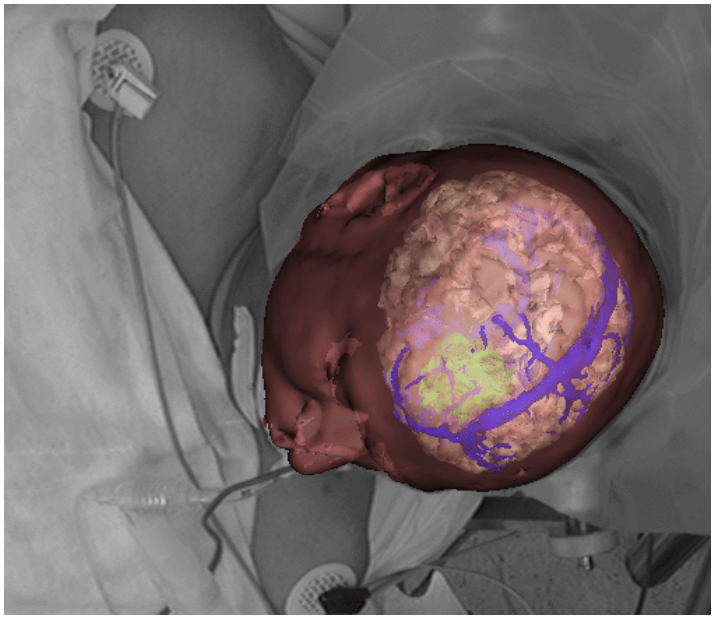


Figure 1.5: Example of *Augmented Reality* in medical applications (taken from [LTJ])

Augmented Reality they are made visible to mechanics. You can also think of *Augmented Reality* manuals that tell you interactively how to assemble components by highlighting the required components and combine them virtually.

As military applications you can think of soldiers who get enemies highlighted in their glasses or navigation information in unknown terrain.

In a further paper Azuma et al. [ABB+01] present more recent applications and classify them in three categories:

Commercial applications can be found primarily in broadcast video where either distances or virtual finish lines are shown or commercials are projected for example in the middle of a soccer field.

Mobile applications are enabled by more high-performance mobile hardware and offer outdoor navigation or even gaming.

Collaborative applications allow the user to work together in a totally new way as Billinghurst et al. show with the Magic Book [BKP01].

I will concentrate here especially on the aspects of mobility and collaboration. As mentioned before communication is desired at every time and every place. This work can be a first step towards a mobile conferencing application with multiple users.

Augmented Reality can also lead to new interaction devices for 3D applications. The FingARTips project [BVBC04] at the HITLabNZ showed how you can model and plan environments just by hand movements (see figure 1.6).

Graphical Output Devices

The most commonly used device in *Augmented Reality* is a *Head Mounted Display* (HMD) . There are two different kinds:

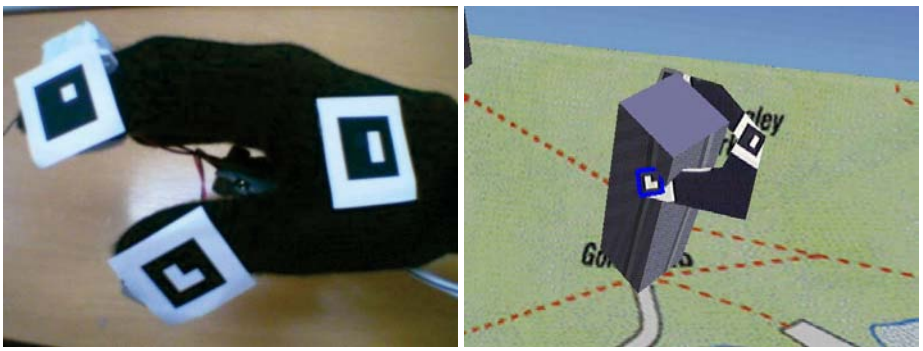


Figure 1.6: New way of interaction with *Augmented Reality* : left a glove with markers, right the virtual hand grasping a virtual building (taken from [BVBC04])

Optical see-through HMD has a semi-transparent mirror, where you can see both, the real world and the virtual objects in one display

Video see-through HMD show a video image of the real world overlaid by the virtual objects

Both device types have different pros and cons. On the one hand with optical see-through you see the real world but virtual objects often lag behind the reality. On the other hand with video see-through you only have a video image of the reality that lags because it is only refreshed when the virtual objects are rendered. A detailed discussion can be found in [RF00].

But *Augmented Reality* is not restricted to HMDs only. Projectors are used for table-top applications as used for example in SHEEP [MSW+03]. Head-Up Displays (HUD) are known in military aircrafts for a long time and now also get realised in the next generation of cars. Head-Up Displays project information directly in the field-of-view of users for example in the windscreen of a car (see figure 1.7).

Figure 1.7: Example of a Head-Up Display in the next generation of cars (taken from [CT03])

Registration

Since the overlay and alignment of virtual objects in the real world should be as realistic as possible you need to track the reality. You call that registering virtual objects in the real world.

For this registration you need tracking systems that provide the position and orientation. You distinguish tracking devices by their degree-of-freedom (DOF). That means for the position in a three dimensional room you have three degree-of-freedom and for the orientation you have three axis around which you can rotate the objects that result in another three degree-of-freedom. To get hold of the position and orientation for totally free moving objects in the three dimensional space you need a six degree-of-freedom (6-DOF) tracker.

Further on in this work the term *Pose* will refer to the combination of position and orientation, so the 6-DOF information of objects.

There are different tracking techniques known for *Augmented Reality*. You can classify them roughly in vision-based technology and systems with extra hardware except for a camera.

Vision-based tracking captures the real world with cameras - mostly ordinary web cams. You separate between marker and markerless tracking. Marker tracking needs fiducial markers such as paper cards with special patterns placed in the scene. The ARTOOLKIT [BK99] is a well-known exponent of this technology and will be presented in detail in section 4.2.1. Markerless tracking approaches either use motion flow in the video images [HK99] or use natural-feature tracking [NY99].

Other tracking systems use special hardware like magnetic, inertial, or acoustic sensors. The drawback of these systems is that you either do not obtain the full 6-DOF (inertial).

Especially in outdoor applications a hybrid approach is followed (an example can be found in [AHNS99]). That means that the fusion of different tracking systems like the Global Positioning System (GPS) and an inertial sensor leads to wide range and accuracy at the same time.

1.5 Problem Statement

The aim of the underlying work was to build a teleconferencing system simulating a meeting where the participants are sitting around a table and can work collaboratively on objects that can be models, documents or presentations for example. The focus was to investigate possibilities to maintain gaze awareness between the users and on the objects on the table.

The user should be able to work in her known workspace without a costly hardware set-up or longsome preparations.

In a final user study the results leading to a first prototype have been evaluated. Therefore a suiting task had to be developed and a set-up designed.

1.6 Thesis Contribution

In the work described in this thesis a first prototype of a conferencing application is developed and evaluated. The approach is to visualise users as avatars placed directly in the real

world with help of *Augmented Reality*. The gaze is expressed by the rotation of the avatars and small arrows on the shared workspace.

Results of a final user study showed an increased awareness and high user acceptance. This approach of abstracting the gaze allows to reduce the network load and so can lead to even mobile usage [SW05].

CHAPTER 2

Gaze Awareness in Augmented Reality Teleconferencing

I start with a discussion of problems in teleconferences and collaborative virtual environments (CVE). Thereby I debate types of awareness and techniques allowing to reflect these information. A section about gaze awareness will lead to gaze awareness between participants and then introduce the concept and idea behind the *shared workspace*. I will complete this chapter with a presentation of benefits that *Augmented Reality* can contribute in CSCW.

2.1 Problems in Teleconferences and Importance of Gaze Awareness

When humans are in face-to-face conversations they do not only communicate by voice. People use their facial expressions, head movement or body language to emphasis their opinions and ideas. But there is even non-verbal communication that leads to fluent and non-interruptive conversation. A speaker seeing her dialogue partners nodding with their heads she can go on explaining facts without asking if everyone understood the previous explanations ([CBB+99]). Vertegaal et al. [VSvdVN01] showed that listeners are actually looking at speakers and individuals talking to someone mostly look at the addressed person in conferences.

2.1.1 Awareness Problems

As discussed already in section 1.2.1 video conferences can convey these spacial cues compared to just audio conferences, but there are problems left.

New problems emerge when persons are working on an object - such as a large UML-class diagram, a model of a new car or a city model - in a teleconference together. While they are discussing everyone want to point on certain regions of the model and have her own view of the object. The first issue is how to share such an object.

If the object exists in reality there is often only the possibility to have a video transmission of the object to the remote side or there are two exact physical copies. In case of the video transmission it is only possible to the persons on the side with the physical copy to have their

independent view of the object and the possibility to point freely. Whereas the persons at the remote side just have the statical video image and no way to point on the object. In case of the two copies either users have to describe the part of the object they are looking at or talking about to the remote side. This demands verbal communication just for coordination and interferes the normal conversation. If there are video channels to transmit the object user can point on the object but on the remote side they always have to switch between the video image of the remote copy with pointing information and their own independent view of the real object. This switching between real and video world is very unpleasant. Again the quality of the remote object through a video stream will not be very good.

In case we have electronic data there is no problem of sharing the data. Depending of the amount of data and the network connection data is either distributed a priori or updated during the collaboration session. But collaborating on electronic models or documents still introduces a lot of problems. There is on the one hand the problem of telepointers already discussed in section 1.2.2. They allow pointing but require extra coordination.

On the other hand on electronic data every user can easily have her own view. For two dimensional data it is easy to build interfaces allowing users to zoom in the data and scroll. But here the problem appears that users cannot just talk about what they see. Typical phrases like *"Do you see the square with the circle down right?"* mislead totally in a setting as you can see in figure 2.1.

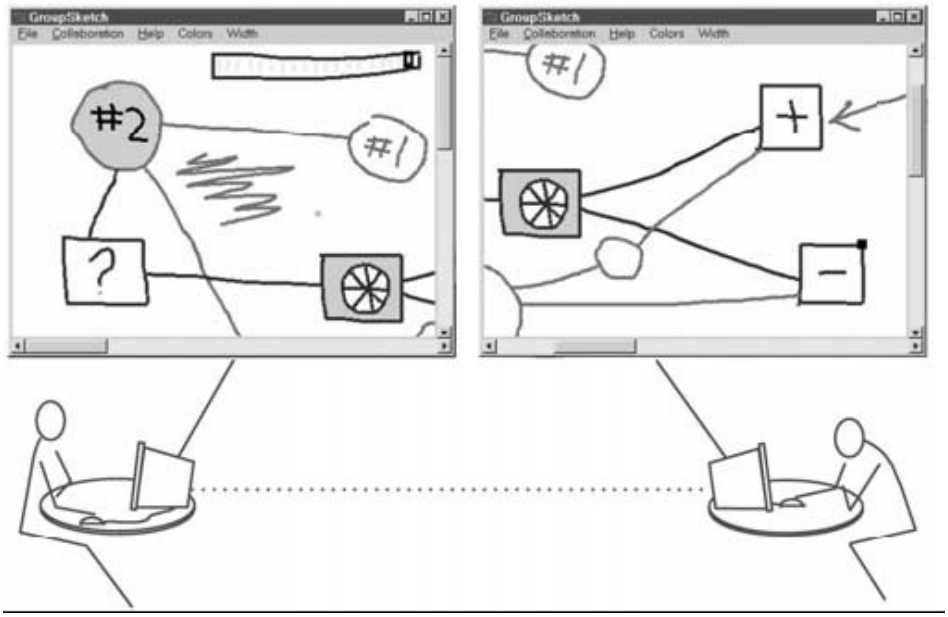


Figure 2.1: Two users with two different views on the same data (taken from [GG02])

2.1.2 Providing Awareness

There are two common ways to conquer that problem. One is to have two windows in the application. A *private* where every user is independent and a *public* that is the same for every user. The control of this public view has either a single moderator or everyone combined with special floor passing regulations.

A second well known possibility is that so called *radar views* are added. In such windows you have an overview of the whole data and you see rectangles representing the view of the other users (see figure 2.2).

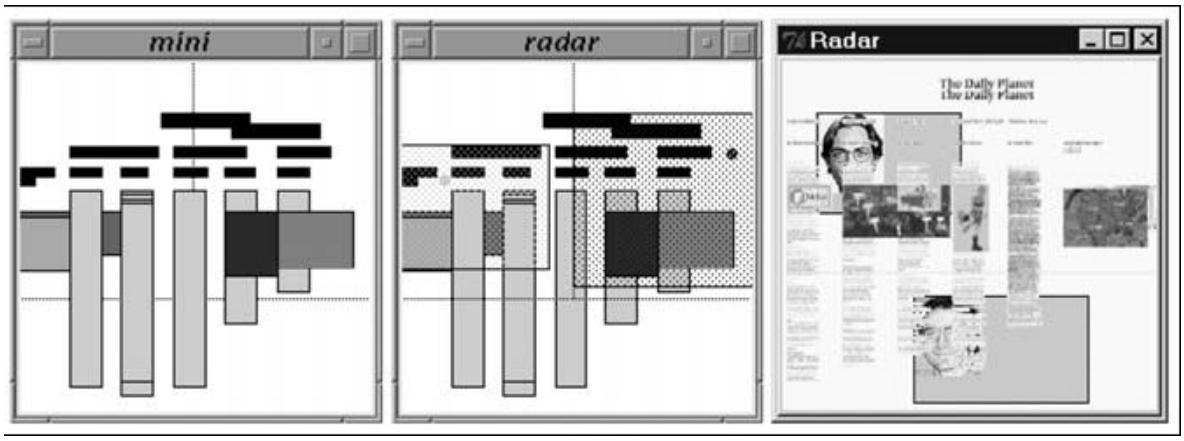


Figure 2.2: Three overview windows: a) shows just the data; b) the shaded rectangles representing the view of the other users; c) rectangles enhanced with photos to ease the identification of the participants (taken from [GG02])

In 3D it is even more complex to visualise information of the other users since there is not only 2D position and zoom but a 3D position, the orientation of the user plus the field-of-view. In [DG01] Dyck and Gutwin call these information *location awareness* in collaborative virtual environments (CVE). This is the *Where?* component of the workspace awareness mentioned in section 1.2.2. In figure 2.3 you see an overview which parts location awareness is composed of.

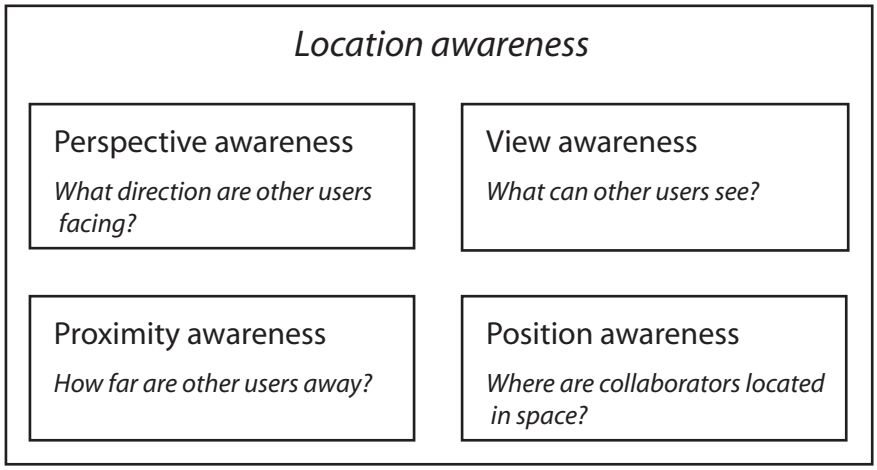


Figure 2.3: Overview of location awareness after [DG01]

Avatars and Location Awareness A common and wide accepted technique to provide location awareness is the use of avatars. An avatar is an embodiment of a person in the virtual space. Such avatars should convey the following issues after Benford et al. [BBF+95]:

Presence: Is someone present in a virtual environment?

Location: Indicates the position and orientation of a user.

Identity: Allows to identify the person the avatar represents.

In CVE avatars can provide location awareness but often they are just an anchor for more detailed visualisation mechanisms. There are two different concepts to achieve location awareness. One allows users to share the view of another user. This can be realised by clicking on the avatar representation of a user and a new window opens with this view ([PW97]). The alternative is to display the views of the other users directly in the own view. Dyck et al. propose in [DG01] three different ways:

Nose ray: The idea of the nose ray is to add an oversize nose to the avatar that allows to recognise the orientation of a user easier and even allows to determine the position of a user even you do not see the avatar (see figure 2.4a).

View cone: The view cone exactly shows the view frustum of a user with a transparent cone pointing away from the avatar. However the cone is additional data that can occlude other important data (see figure 2.4b).

Headlight: The headlight is a spotlight mounted on the head of the avatar and pointing always in direction of view. The big advantage is that there is no additional visual data in the view that can disturb but the location of a not visible avatar is not that easy to determine as with the view cone or nose ray. (see figure 2.4c).

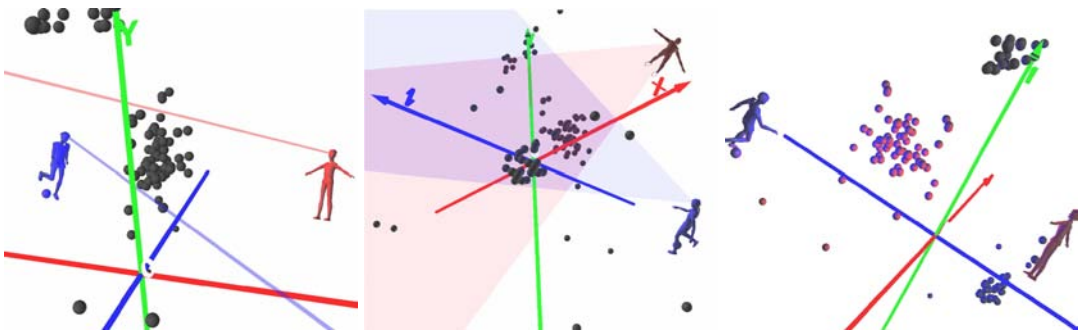


Figure 2.4: Three different techniques to maintain location awareness: a) nose ray, b) view cone, c) headlight (taken from [DG01])

2.2 Gaze Awareness between Users

As you will see in chapter 3 there are several approaches to support gaze awareness in teleconferences. I developed a new idea to conquer that problem because there is no application yet that combines conferencing in the real world and at the same time provides gaze awareness between users and on a shared workspace. Already known applications just satisfy one aspect or suffer from clumsy set-ups.

The idea is to enable teleconferencing on the user's common workspace without complex hardware set-ups. To allow the user to go on working in her workspace and have a teleconference at the same time I use *Augmented Reality* . As presented in section 1.4 *Augmented Reality* makes it possible to enhance the real world with extra information.

Every user sets up for every remote participant of the teleconference a special designed paper card. On these paper cards the user can see avatars representing the remote users. To show these avatars the users use a so called handheld display ([WBL+04]). This is a combination of video glasses and a web cam mounted on a handle bar (see figure 2.5).

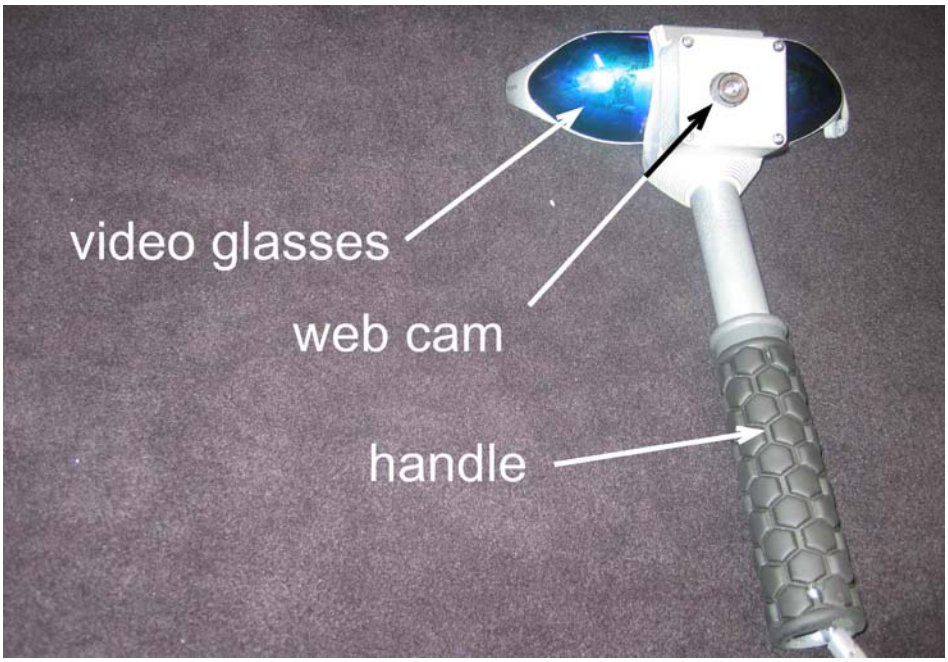


Figure 2.5: Photo of the handheld display

Due to the fix combination of the camera and the video glasses you can capture the gaze of the user. So it is possible to recognise when a user is looking at an avatar. This information can be propagated now to the other users via the network. The clue is now to rotate the avatars depending on the gaze information received via the network.

Let's have a look at an example scenario to clarify the facts. We assume in the following that there is a conference with three participants (A, B and C). Every participant has two paper cards for the representation of the remote users ($P_{A,B}$, $P_{A,C}$, $P_{B,A}$, $P_{B,C}$, $P_{C,A}$, $P_{C,B}$). If user A is looking through the handheld display at paper card $P_{A,B}$ she can see the avatar representation of user B. Assume now that person A looks at person B - actually the paper card $P_{A,B}$ (see figure 2.6). At person C the virtual representation of person A is rotated in direction of person B on paper card $P_{C,B}$.

The rotation of the avatars mirrors the gaze of the users. An important detail is that it is only recognise at which avatar someone is looking and not if someone is turning her head right or left. Thus every user can set-up the paper cards freely in her own workspace. That is important because a user can make up personal connections between a person and real world objects on the desktop. So it is possible for users to remember where a person is located without actually looking for it. This is an important advantage of the application

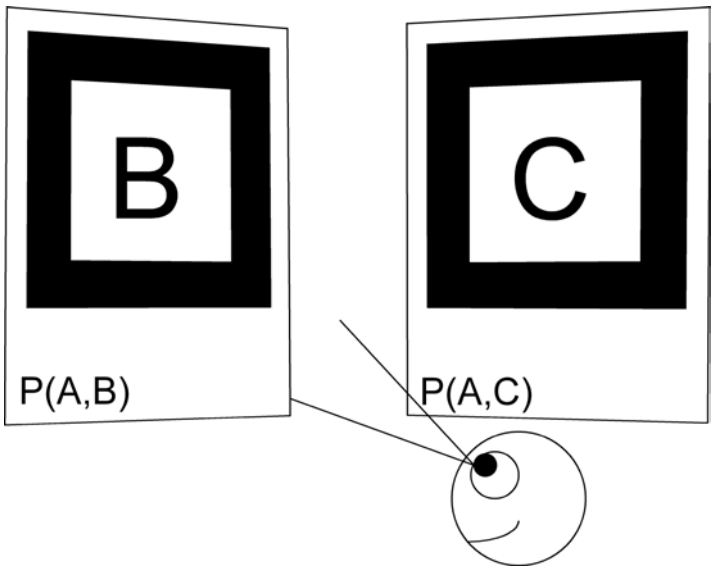


Figure 2.6: Schematic set-up of the paper cards. User A is looking at paper card $P_{A,B}$

as the user can adopt the application to own preferences and does not have to adopt to the application.

2.3 Awareness on the Shared Workspace

There is not only visualised the gaze awareness between the user. A *shared workspace* is implemented in the application as well. So it is possible to share 3D models among all users and everyone has her own view.

An extra paper card works as an anchor to display a virtual model. By rotating and moving the paper card or moving around the paper card with the handheld display it is possible for every user to gain an independent view on the data.

On the shared workspace gaze awareness is provided as well. With help of the paper card it is possible to calculate exactly where the users are looking at the shared workspace. This information is used to display small arrows on the object. To gain the mapping between arrows and the persons two aspects are implemented. Every arrow has the same colour as the corresponding avatar and is directed from the avatar.

This approach satisfies all the claimed requirements for workspace awareness as presented before.

Who? The avatars are personalised by photographs and the arrows are directed from the avatars towards the place where users are working. So the user is aware who is working.

Where? Since the avatars are rotating accordingly to the orientation of the user we know where they are working and what or whom they are looking at.

What? Combined with the voice of the remote participants it is easy for the group to understand the intention of users and what they are talking about.

2.4 Benefits of AR in CSCW

The problem of CSCW applications is that users often have to adopt to new work processes of interaction techniques. Grudin discusses in detail the well-known problem of electronic calendars in companies [Gru88]. Electronic calendars are a very useful tool to plan meetings thus they can find spare time at every user's personal calendar. But therefore users have to take care of their calendars. In the management of a company this is common and sometimes even done by the secretaries whereas it is extra effort for the normal employee to maintain his own calendar and thus not fully accepted.

In real-time collaboration Ishii et al. [IKA94] claim *seamlessness* as a key feature leading to acceptance of CSCW applications. This means on the one hand the possibility to allow users to continue with existing practices and on the other hand to smooth the shift between different spaces.

Augmented Reality can perfectly fulfil these requirements. Users go on working in their real world and get extra information superimposed where they are needed and fit. In combination with technology as gesture recognition common work sequences can be supported and the system adapts to the user.

In my application users can model their workspace freely and set-up the remote persons with help of the paper cards depending on their own preferences. When users want to work on something else than on the shared workspace they can take the paper card and put it aside as everyone would do with a sheet of paper.

CHAPTER 3

Related Work

In this chapter I present an overview of research projects finished or still running that are related to my work in context of remote conferencing, awareness, *Augmented Reality* and Virtual Reality. These are conferencing applications that use video streams or Virtual Reality to convey awareness information. There are also applications that combine *Augmented Reality* and video streaming. I will present benefits I learned from and disadvantages I want to solve with my idea.

3.1 Hydra

The Hydra [SBA92] system was one of the first approaches to conquer the problem of eye contact and gaze awareness in video conferences. As you can see in figure 3.1 a Hydra unit is needed for every remote user. Every unit consists of a camera, a small screen, a microphone and speakers. With this set-up eye contact can be established. Consider following scenario: When user A looks at user B, the camera in unit B at user A captures A's face frontal. So user B can see user A from the front and knows that she is addressed. The software switches video and sound to the correct units that user C and D even can recognise that user A is addressing B.

As mentioned in [SBA92] the video images are very small and do not really lead to direct eye contact. A free set-up of the units is not possible because the switching of the audio/video channels would not work correctly. Another drawback is the absence of a shared workspace compared to my idea. So it is possible to the participants to talk and discuss, but they are not able to actually work on a shared document.

3.2 GAZE

The GAZE Groupware System [Ver99] captures the gaze of users in a remote conference with a commercial eye tracker. The gaze information is used to rotate planes with snapshot photographs of the users in a virtual room as you can see in figure 3.2. These represents are

Figure 3.1: The Hydra system in action: User with three Hydra units (taken from [BR94])

rotating towards other users or the table in the middle. On the table there can be multiple documents placed as icons. A coloured light spot on such an icon shows that one user is reading in that document. When all users are working on the same object the light spots show in the content where the users are looking.

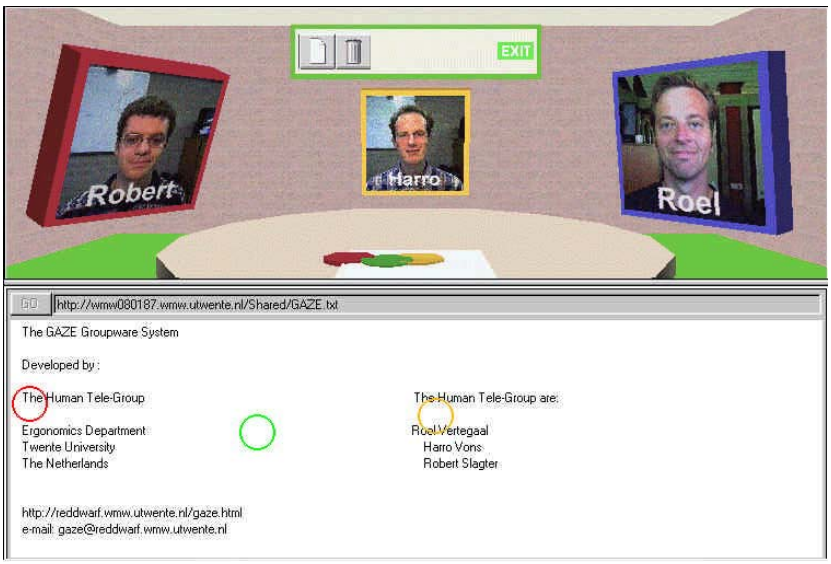


Figure 3.2: Screenshot of the GAZE Groupware System showing the virtual conferencing room (taken from [Ver99])

Because of the limitations of the eye tracker only a head movement of 5-10 cm is possible. Furthermore there is only a rough information in the documents where the user are working and for opening and navigating in the document a mouse is needed as input device. Another drawback is that users are in a virtual room and not in the real world.

3.3 GAZE-2

Vertegaal et al. present with the GAZE-2 system an improved version [VWS02]. They use a special video tunnel to preserve real eye contact. To provide a parallax-free video stream of the faces three cameras are mounted behind a half-silvered mirror in the tunnel (see figure 3.3b). This allows to always send a video image of the face taken directly from the front. The video images in the tunnel are rotating always depending on the information collected by the eye tracker mounted directly below the tunnel (see figure 3.3a). To reduce the network load, the video streams of users that are rotated are compressed harder than the streams that are facing directly at someone. Definitely advantages of this system are that users really get the idea who is looking at them and the solution of the parallax problem. But to provide this feature the shared workspace has been abandoned and of course the set-up is really complex and expensive. Again users cannot work in their own environment but have to immerse totally in a virtual space.

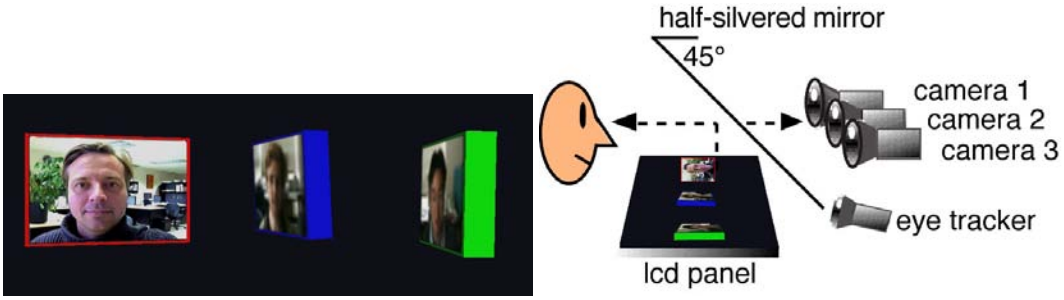


Figure 3.3: a) View in the video tunnel of the GAZE-2 system; b) a schematic set-up of the GAZE-2 system (both taken from [VWS02])

3.4 cAR/PE!

The *cAR/PE!* system [ROW+03] developed at Daimler Chrysler is a virtual conferencing room enhanced with three screens which can be used for presentations, documents and application sharing (see figure 3.4a). On the table in the middle 3D models can be placed for discussion. Users also have the possibility to use the mouse as pointing device on the models. That allows them really to interact easily on the model.

Every user can move freely around in the room. They are represented by planes on which video streams of the users are projected (see figure 3.4b). So all the users are aware where the others are looking.

The set-up of the participants sitting around a table is very naturalistic but the navigation of the view can be really complex. Either you have to use a combination of a SpaceMouse and head tracking or get used to the complex navigation with the keyboard. And still you have to use the mouse as an extra pointing device on the 3D model on the table and cannot use your gaze as in my application.

In a further step Hauber et al. [HBR04] developed new interaction techniques to narrow the gap between real world and the VR room. So it is possible now to use tangible instruments to move, scale and rotate the models on the table and even capture real documents written by hand and share them on a screen (see figure 3.5).

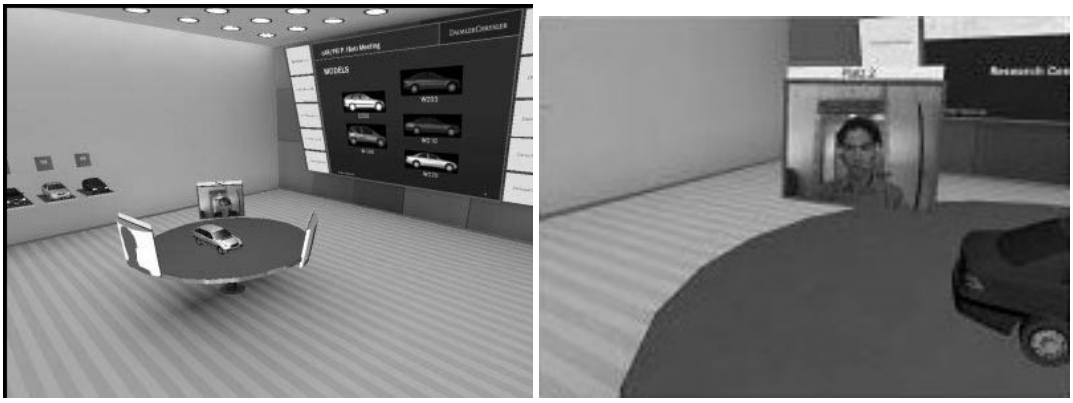


Figure 3.4: a) view of the cAR/PE! room with table in the middle and presentation screen; b) plane with video stream of the user representing his view in the room (taken from [ROW+03])

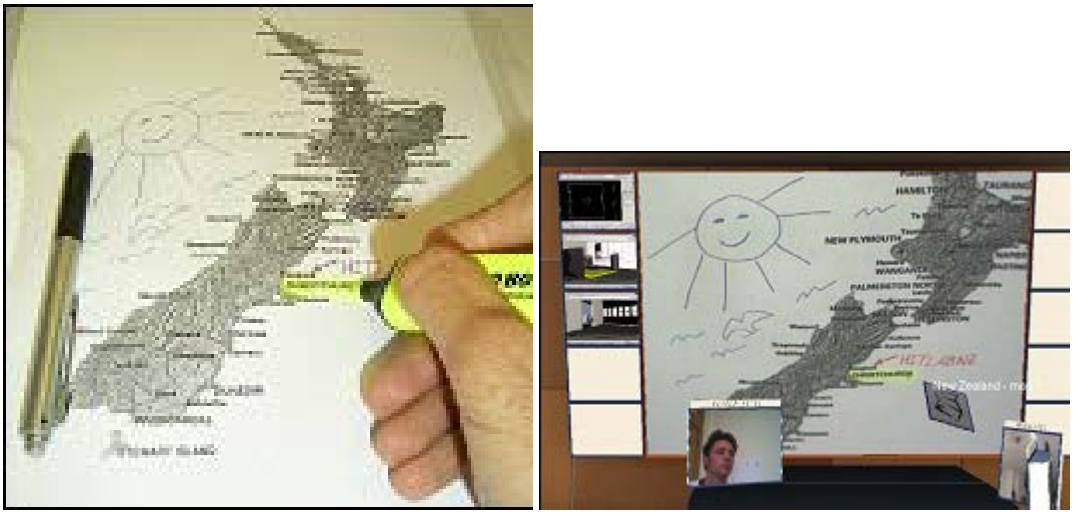


Figure 3.5: a) making annotation on a paper; b) sharing these annotations on a presentation screen (taken from [HBR04])

3.5 AR Teleconferencing

Kato and Billinghurst present an approach to really combine video conferencing and *Augmented Reality*. They use paper cards as anchors to display the video stream of the users ([KB99]). This approach allows them to integrate the video stream of the remote users seamless in the real workspace. Since the paper cards are place holder for these virtual displays showing the remote user they are able to scale the video picture from thumbnail to real human size. Due to not using real displays they can integrate the cameras in the paper cards what solves the parallax problem you have with the placement of cameras on top of ordinary screens.

Since there is no real network connection implemented it is only possible for one user to see the video images of the other participants on the paper cards. To actually see the video images the user need to wear a HMD that of course occludes the eyes, what prevents a real eye contact.

They also integrate a virtual shared white board on which all users can write. The user with the HMD can write with a light pen and the other users with the mouse (see figure 3.6a). To improve the seamless integration of the remote participant in the real world an algorithm was developed to eliminate the background as you can see in figure 3.6b).

If actually every user should be able to wear a HMD and see the remote participants without the parallax failure there are $n^2 - n$ (with n as the number of users) video streams necessary whereas the user cannot really see the eyes due to the HMDs. If you abandon the multiple video streams per user you do not have gaze awareness any more provided to the user.

Figure 3.6: a) typical setup of the AR videoconferencing application with the virtual shared white-board b) background elimination of the remote users (taken from [KBMT01])

3.6 Studierstube

The Studierstube project [SFH+02] is an *Augmented Reality* system targeting different aspects of collaboration and mobility. In one project they present the idea to enrich a desktop video conference with virtual objects [BFS03]. This offers all the advantages of *Augmented Reality* in collaboration and intuitive interaction. At the same time you can see the remote collaborator pointing at objects as if they were real that is what the application definitely benefits from (see figure 3.7).

Paper cards were optical tracked for interacting with the virtual objects. Since the video stream transmitted to the remote person is compressed the marker detection would not have worked stable. So they decided to detect the markers locally and send them synchronously with the videostream.

The application offers great possibilities to work with virtual objects remote, but is limited to only two users where the aspects of gaze awareness are due to the absence of other participants is not that relevant.

Figure 3.7: Screenshot of the *Studierstube* Augmented Reality Videoconferencing application where virtual objects are placed on paper cards during a desktop video conference (taken from [BFS03])

CHAPTER 4

Implementation

The description of the implementation details is structured as follows. After an overview of my system design and some remarks on software design in general I discuss design decisions, list alternatives and give some improvement suggestions before I present the components in detail where I will also put a focus on the underlying network communication.

4.1 System Design

The application was developed as a prototype in iterative steps and is not meant to lead to a final product. I followed design guidelines and design patterns where appropriate and possible.

The application is meant to work as a peer-to-peer application without a central server. So it is possible to every user group to start a session without extra hardware and configuration overhead.

4.1.1 System Decomposition

As you can see in figure 4.1 the system is separated in the following components:

Tracking: This component is responsible for acquiring the video frames from the web cam, tracking the paper cards and calculate their positions.

PoseDataBuffer: The information of the tracking component gets stored here and missing relationships between the paper cards are computed if possible.

GazeAnalyser: Here the gaze of the user is analysed by means of the data retrieved from the PoseDataBuffer.

Networking: This component sends out the data of the gaze analyser over the network and collects the received data from the other users.

SceneGraphAssembler: Gaze information from the networking component and the position information from the PoseDataBuffer get combined to assemble the scene graph.

Viewer Displays the scene built in the SceneGraphAssembler.

Persistent Datamanagement: Application settings as user names etc. are loaded from configuration files and provided to the other components.

A detailed description of the single components you will find in section 4.3.

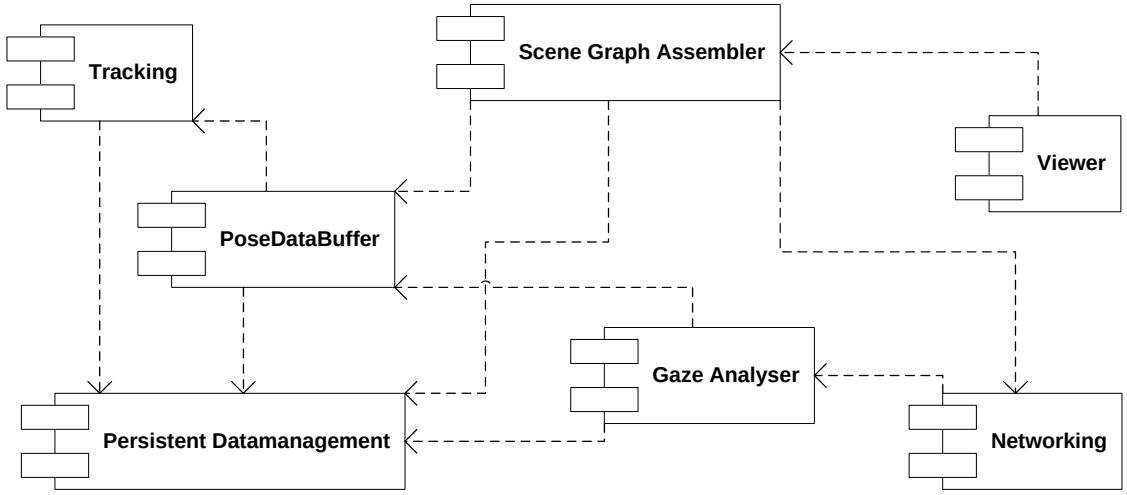


Figure 4.1: UML component diagram: illustrating the system design and the dependencies between the components

4.1.2 Concurrency

To provide the maximal flexibility the application is separated in four different main threads: tracking, gaze analysing, scene assembling and displaying the scene. This allows to provide full priority to the most important threads - tracking and displaying the scene - and to the other both less resources. This distinction is important since tracking must work stable and displaying the scene must be fluid that the user experience is not disturbed by jerking images.

The gaze for example is only analysed every 100 milliseconds and so reduces the CPU load. This update rate is totally sufficient as tests showed.

In figure 4.2 you see a rough overview how the separate threads collect information from the other components, handle them and store them. You can see the four components (Tracking, GazeAnalyser, SceneAssembly and Viewer) starting work independently in own threads without having be called before. The Persistent Datamanagement component is omitted in this diagram for clarity reasons.

4.2 Design Decisions

During development I had to trade off between different techniques and libraries I wanted to use. In this section I present problems, requirements and my decisions.

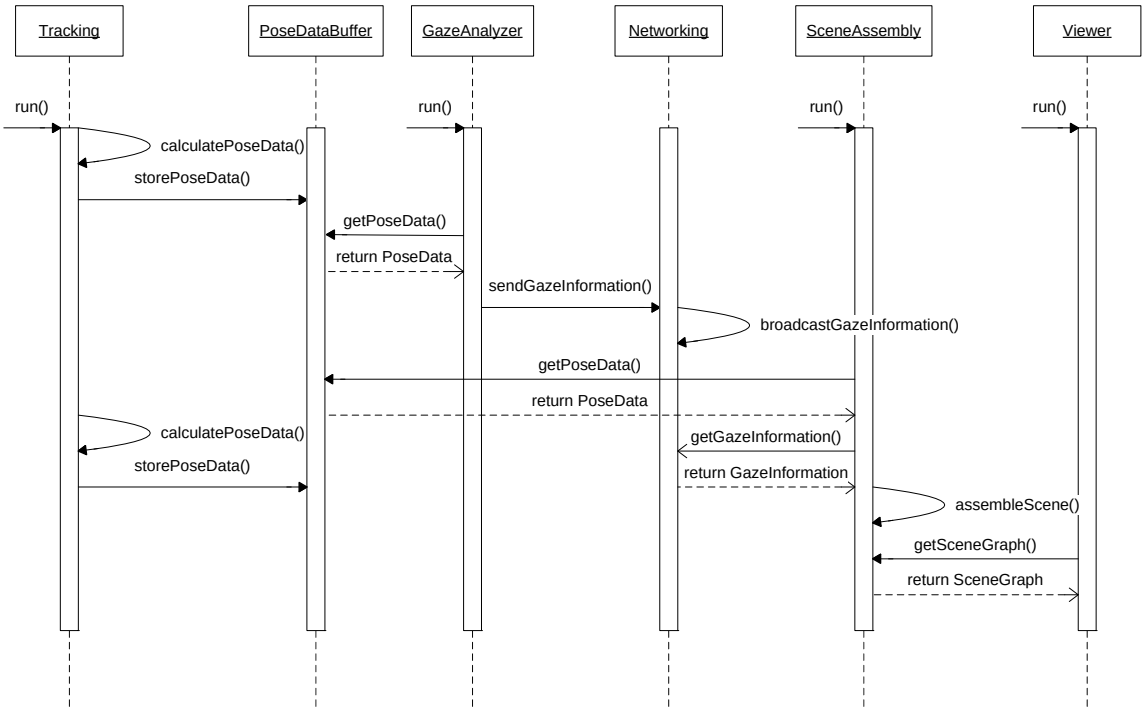


Figure 4.2: UML sequence diagram: showing the four independent working threads

4.2.1 Tracking

The requirements for the tracking are:

- The tracking subsystem should be useful for mobile usage,
- must provide relationships between objects (head - paper cards),
- integrates easily in the workspace and
- preferably goes without extra hardware.

As possible options for the tracking subsystem a magnetic tracking system or an optical tracking system could be considered because inertial tracking systems only allow the tracking of 3-DOF and no relationship between objects. Magnetic tracking systems always require a reference system and their sensors are linked via cables to the system what does not allow totally free movements. Whereas optical tracking systems provide an easy method to track artificial markers in a workspace with full 6-DOF.

I decided to use the ARTOOLKIT , since it is a free library for non-commercial use, easy to set-up and requires only an ordinary web cam.

ARToolKit

The ARTOOLKIT [BK99] was developed by Billinghurst and Kato at the Hitlab, University of Washington. It searches in video images captured from an ordinary web cam for artificial

markers. These markers are squares with a black frame around and a unique symbol in the middle to distinguish them. The recognition of the markers is done in three steps:

1. Due to thresholding the acquired image it is possible to search efficiently for the edges of the black squares with picture processing algorithms.
2. With these edges the corners of the square can be calculated and, as we know the size of the square, the position and orientation of the marker can be calculated. But still there are four possibilities for the orientation of the marker left because the square is rotationally symmetric.
3. In the last step the symbol in the middle does not only help to distinguish the markers but if it is not rotationally symmetric itself it allows to decide the final correct orientation.

To represent the *Pose* - position and orientation - of the markers in a mathematical way the ARTOOLKIT uses a 3×4-matrix.

$$\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_0 \\ r_{21} & r_{22} & r_{23} & t_1 \\ r_{31} & r_{32} & r_{33} & t_2 \end{pmatrix}$$

The first three columns are representing a 3×3 rotation matrix describing the rotations around the three axis. The fourth column stores the position or also called translation. To draw virtual objects directly on the markers in the scene in combination with OpenGL[1] you have to add the row $(0, 0, 0, 1)$ to the matrix and transpose it. Now you can load this matrix as the OpenGL modelmatrix to render the scene graphs on the right position with the correct orientation in the real world. Further information on rotation matrices and a intuitive explanation you can find in section 4.3.2.

All these steps are condensed in figure 4.3 one more time.

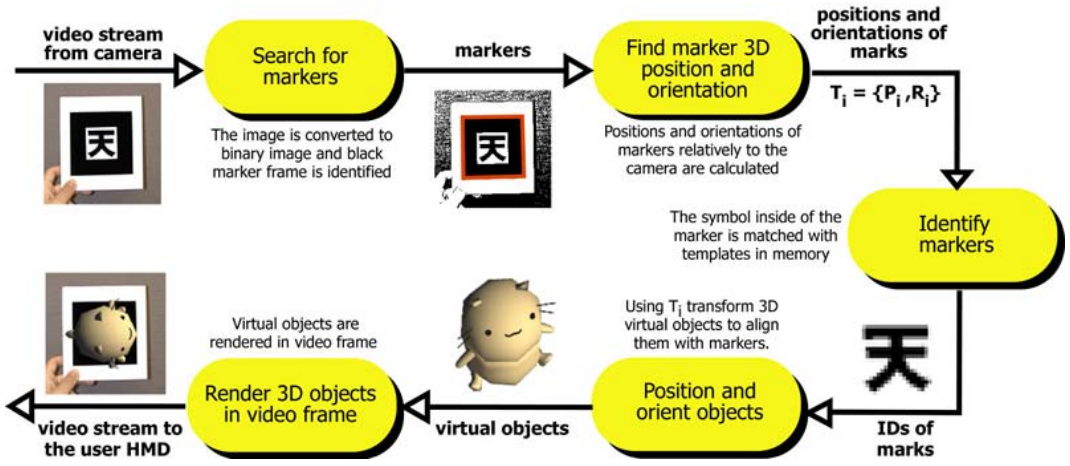


Figure 4.3: Overview of the marker recognition in the ARTOOLKIT (taken from [BK00])

A problem of the ARTOOLKIT with the markers is that the whole black square must be visible and may not be occluded. This is especially a problem if users want to get closer to

[1] http://www.opengl.org

a virtual model and so the whole marker does not fit into the video picture or the model is so big that it does not really fit in the scene. Because the virtual objects often occlude the marker the user interacting with the marker cannot recognise if the marker is going to leave the video frame and so lose the orientation what is very inconvenient.

One solution is the use of multimarkers. This means that there are multiple smaller black squares arranged on one sheet of paper. In a configuration file you specify the size and the distances of the markers and the ARToolKit calculates a *Pose* for the multimarker even if there is only one marker visible.

This technique can also be used to build user interfaces that allow to interact with the multimarker [LBK04]. You can determine markers that are not visible but should be because of the markers around them are (see figure 4.4).

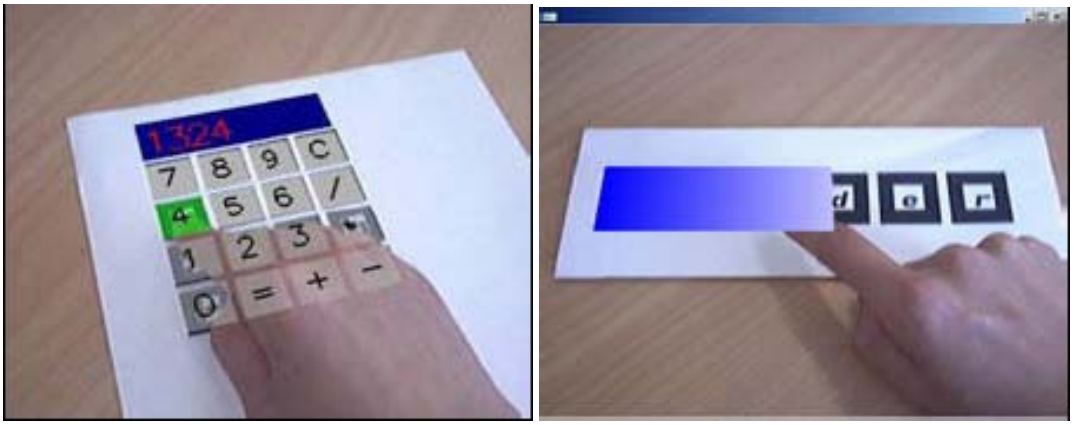


Figure 4.4: Occlusion of markers used as user interface (taken from [LBK04])

Natural Feature Tracking A new improvement is the so-called *Natural Feature Tracking* or *Texture Tracking* added to the ARToolKit in the new version. The aim is to get rid of the black squares and the associated handicaps as mentioned before. The idea is to find feature points in an image and calculate the position by means of these points. Therefore the picture has to get preprocessed one time and the significant feature points are calculated. Since today computer power does not allow to search for these feature points in a whole video frame in real-time it is necessary to place one small black square marker in the image before preprocessing. So this marker can be used for an initial pose estimation. In the next steps the algorithm can go on working just by the feature points found in the previous video frame. To hold down the computational effort these new feature points are only searched in a region around the last feature points since you can assume that movements are not too big. In case there is too much movement and no feature points can be found the user has to start at the initial marker again.

Natural Feature Tracking still suffers from instability caused by computational expensive calculations done in real-time. The work of Felix Löw tries to find repeating interaction patterns in the user's behaviour to improve the tracking algorithm [Löw05].

MagicBook The MagicBook is a popular example application that allows users to see virtual scenes on a natural book. The book is a picture-book telling a story. The pages of the

book are equiped each with a small marker and preprocessed to extract all the feature points.

When a user now looks through a handheld display at the pages of the book she can see a virtual 3D scene popping up to illustrate the story with small animations (see figure 4.5).

Figure 4.5: MagicBook: a 3D scene pops up if you look at the book through a handheld display (taken from [MD03])

4.2.2 Network Communication

For the network communication layer the following requirements should be accomplished:

Multicast/broadcast: A protocol should be chosen that allows to send messages to multiple users per multicast if that is not feasible broadcast would be acceptable.

Support peer-to-peer infrastructure: Due to the peer-to-peer requirement the network software should work without a central server.

Easy to configure: If possible it should not be necessary to set-up the IP addresses or computer names but they should find each other autonomous.

Possible protocols and frameworks that I examined and compared were DWARF , Ice and a handcrafted solution based on the UDP protocol.

DWARF

The *Distributed Wearable Augmented Reality Framework* (DWARF) [BBK+01] developed at the chair for Applied Software Engineering at the Technische Universität München is a component-based framework designed for the development of *Augmented Reality* applications.

The component-based approach allows the easy reuse of modules for rapid prototyping. Components for different tracking techniques are integrated as well as viewer, data storage and user input.

The whole system is based on CORBA that allows platform independent development and even enables the use of handhelds. The main concept is that *Services* find each other ad-hoc on basis of certain criteria. A *Service* is a programme that provides *Abilities* and requires *Needs* or both. The Service Manager running on every machine builds the connection to the middleware. It is responsible to locate other Service Managers in the network and to connect *Services* that match together by means of their *Needs* and *Abilities*. The Service Managers find each other with help of the *Service Location Protocol* (SLP)[2]. The *Needs* and *Abilities* of a service can either be defined a priori in a XML file (Service Description) or dynamically at run-time via an API.

DWARF offers the *Services* three different ways of communication:

Event based: *Services* can subscribe to certain channels and receive suiting events that are send out from another *Service* .

Object references: *Services* export interfaces specified in IDL[3] that other *Services* can import to call methods on the remote object.

Shared memory: *Services* can exchange data via a shared part of the memory. This is especially useful if a great amount of data has to be transferred. This feature is currently only available under Linux and both *Services* must run on the same machine.

Ice

The *Internet Communications Engine* (ICE) [4] is an object oriented-middleware with ports to different operating systems supporting the programming languages C++, C#, Java, Python, PHP and Visual Basic.

It supports different paradigms as synchronous and asynchronous method invocation. It is scalable and supports location transparency. The authors claim that Ice is much less complex than CORBA and so faster to develop.

Ice comes along with some services that make programming easier [Hen04]. A security service provides secure communication through firewalls, a persistence service allows storage of object states and an event service distributes events efficiently to the users.

To define interfaces and data types the language Slice is developed with compilers for all supported programming languages.

[2] http://www.ietf.org/rfc/rfc2608.txt
[3] http://www.omg.org/cgi-bin/doc?formal/04-03-01
[4] http://www.zeroc.com

Handcrafted Solution based on UDP

The *User Datagram Protocol* (UDP) [5] is based on the IP-Protocol and is in contrast to the *Transmission Control Protocol* (TCP) connectionless for packet-switched communication. It is unreliable but therefore fast because it does not require to establish a connection before transmission. It allows a simple broadcast, but many router do not forward broadcast packages, so the broadcast is often restricted to subnets. As it is only gaze information that must be spread it is easy to design a simple protocol based on UDP that allows to broadcast the gaze information to all users.

As the application should be evaluated in a final user study it was necessary to build an application that can be booted easily and runs stable. These requirements DWARF could not fulfil because of stability problems. Because of the simple task for the network layer in this step of the implementation to just broadcast the gaze information the UDP protocol was sufficient. In a further development step Ice would be a good alternative as it would enable the communication even over firewalls and so in the whole internet.

4.2.3 Persistent Datamanagement

To decide which kind of persistent data management to choose, the data to be stored has to be specified and possible solutions must be discussed.

The following data has to be stored:

3D models: Since the models of the avatars and the objects on the shared workspace are not build by code but designed in a 3D modelling tool these data are existent as files.

Pictures: To personalise the avatars photographs of the users have to be saved.

Marker data: For every marker on a paper card you need a file specifying the shape of the symbol.

Configuration data: Settings like the user name, the other users that are participating in the conference and the scene that should be visible on the shared workspace should be configurable in one central point.

Since a first design goal was to get by without a central server I skipped the option of a database. So the easiest and fastest solution was to use files and a configuration file. The problem is that these files have to be consistent and updated at every machine. Since it is only a first prototype this option was accepted.

For the configuration file the Extensible Markup Language (XML) [6] was used because it allows to structure the content and is readable by humans and machines at the same time. I used the Apache Xerces-C[7] XML parser instead of the Microsoft MSXML [8] to enable the port to another operating system than Microsoft Windows.

[5] http://www.ietf.org/rfc/rfc768.txt
[6] http://www.w3.org/XML/
[7] http://xml.apache.org/xerces-c/
[8] http://msdn.microsoft.com/library/en-us/xmlsdk/html/xmmscXML.asp

4.3 Components in Detail

In this section I will present the important components of my application in detail. This will illustrate on the one hand the concepts of gaze capturing and visualisation and on the other hand provide implementation details for developer that are interested in techniques and methods used for this application.

4.3.1 The Tracking component

The Tracking component mainly consists of the before mentioned ARTOOLKIT . That is responsible to acquire the video images from the web cam, detect the markers and calculate the *Pose* of the markers relative to the camera. All the configuration data that the ARTOOLKIT needs such as marker data are provided by the Persistent Datamanagement component.

As mentioned before the ARTOOLKIT delivers a 3×4 translation matrix storing the *Pose* - position and orientation. Whereas in this application the internal representation of the *Pose* is stored as a 3-dimensional vector $\mathbf{p}$ for the position and a quaternion $\mathbf{o}$ for the orientation. This representation is further referenced in this work as *PoseData* $(\mathbf{p}, \mathbf{o})$.

Quaternions

Quaternions were discovered by the mathematician William Rowan Hamilton (1805-1865) and are an extension of the complex numbers. Whereas the complex numbers come along with the $\mathbf{i}$ as complex element for quaternions three are used and defined as:

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$$

So you can represent a quaternion as

$$\begin{aligned} \mathbf{q} &= (q_0, q_1, q_2, q_3) \\ &= (q_0, \mathbf{w}) \\ &= q_0 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3 \end{aligned}$$

Intuitively you can imagine quaternions similar to the axis-angle representation. The three complex components build an axis and the real component specifies the angle of rotation $2\arccos(q_0)$.

Quaternions not only allow to save a rotation more efficient but also multiplications are less computational expensive. The main reason why quaternions are used in this work is because they allow the linear interpolation between two rotations, called *spherical linear interpolation* (SLERP). The interpolation with matrices are impossible and with the axis-angle representation often error-prone [Muk02].

The following equation allows to calculate the interpolation between two rotations for every value of h between 0 and 1:

$$\begin{aligned} \cos\Omega &= \mathbf{q}_1 \cdot \mathbf{q}_2 \\ Slerp(\mathbf{q}_1, \mathbf{q}_2, h) &= \frac{\mathbf{q}_1 \sin((1-h)\Omega) + \mathbf{q}_2 \sin(h\Omega)}{\sin\Omega} \end{aligned}$$

NB: $\mathbf{q}_1 \cdot \mathbf{q}_2$ means the inner product and not the multiplication of the quaternions.

SLERP is used in the application that the avatars turn smoothly and not flip from state to state.

Registering Markers against each other

The *PoseData* of every marker is stored in the PoseDataBuffer component. It is necessary that the application is aware where all the markers are even if only one marker is visible. You can achieve that by calculating the position and orientation of the markers against each other as follows.

Position At the time t_0 the translation $\mathbf{t}_w(t_0)$ in world coordinates (the red arrow in figure 4.6) is calculated as the connection between the two markers:

$$\mathbf{t}_w(t_0) = \mathbf{p}_B(t_0) - \mathbf{p}_A(t_0)$$

This connection vector must be transformed now into the coordinate system of marker A. You can do that by rotating the connection vector around the inverted quaternion:

$$\mathbf{t}_A(t_0) = \mathbf{o}_A^{-1}(t_0)\ \mathbf{t}_w(t_0)\ \mathbf{o}_A(t_0)$$

At the time t_1 when you need the information where marker B is located but you only see marker A with the camera you can calculate the position $\mathbf{p}_B(t_1)$ in case markers A and B have not moved but just the camera. So the marker B is still at the same position in the coordination system of A and the vector $\mathbf{t}_A(t_0)$ is still valid. Since the camera has moved the vector $\mathbf{t}_w(t_0)$ is not valid anymore and we have to calculate $\mathbf{t}_w(t_1)$ by rotating $\mathbf{t}_A(t_0)$ around the quaternion $\mathbf{o}_A(t_1)$ representing the current orientation of the marker A in the world coordinate system:

$$\mathbf{t}_w(t_1) = \mathbf{o}_A(t_1)\ \mathbf{t}_A(t_0)\ \mathbf{o}_A^{-1}(t_1)$$

To get the position of marker B we just add the vector $\mathbf{t}_w(t_1)$ to the position of marker A that we know:

$$\mathbf{p}_B(t_1) = \mathbf{p}_A(t_1) + \mathbf{t}_w(t_1)$$

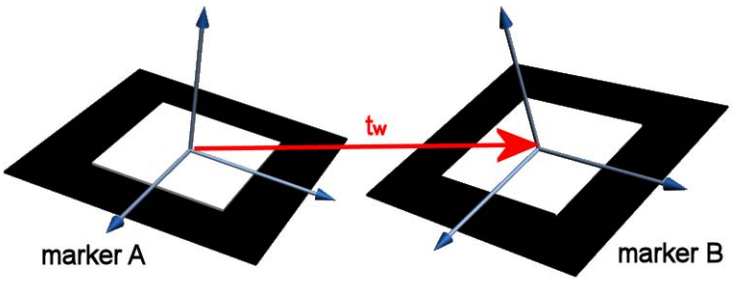


Figure 4.6: Two markers with coordinate systems and the translation vector

Orientation The orientation is even easier to calculate. The inverted quaternion $\mathbf{o}_A^{-1}(t_0)$ representing the orientation of marker A gets multiplied with the quaternion of marker B:

$$\mathbf{o}_{AB}(t_0) = \mathbf{o}_B(t_0)\ \mathbf{o}_A^{-1}(t_0)$$

At time t_1 you can calculate the orientation of the invisible marker B by:

$$\mathbf{o}_B(t_1) = \mathbf{o}_{AB}(t_0)\ \mathbf{o}_A(t_1)$$

So it will be possible later to rotate avatars in the right direction even if the marker the avatar should turn to is not visible.

4.3.2 Analysing the Gaze

The gaze analysing process is divided in the two parts gaze at an user or at the shared workspace. The GazeAnalyser component first fetches all *PoseData* from the PoseDataBuffer component that is available and then handles every *PoseData* depending if it belongs to a user's marker or the marker of the shared workspace.

Gaze at a User

A marker of an avatar is in the gaze if the marker is in a certain area around the middle of the gaze. The eye vector is the vector pointing from the camera directly straight on (see figure 4.7). So we have to test if the middle of the marker is around that eye vector. Since the z-direction of the position vector (p_x, p_y, p_z) is the same as the eye vector the following constraint checks if the deviation in x- and y-direction is under a certain limit ϵ:

$$\frac{p_x}{p_z} < \epsilon \wedge \frac{p_y}{p_z} < \epsilon$$

In case this constraint is true the information is passed onto the network component.

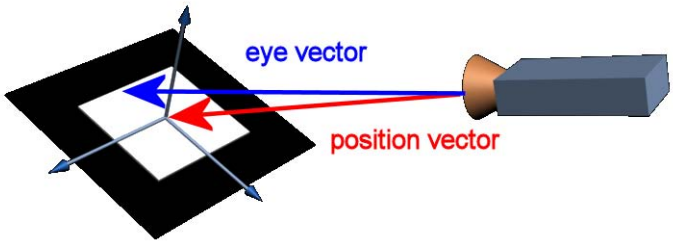


Figure 4.7: Checking if the marker is in gaze of the user

Gaze at the Shared Workspace

For the shared working space the calculations are more complicated as the exact position should be calculated where the user is looking and not only if a marker is within a certain area.

In this version of the application only two dimensional models are shown on the shared workspace. This allows to calculate the intersection point of the eye vector **e** with the plane of the marker. In figure 4.8 you can see a vector chain made up between the camera, the centre of the marker and the intersection point that can be written as:

$$\mathbf{e} = \mathbf{p} + \mathbf{mx} + \mathbf{my}$$

From the rotation matrix M representing the orientation of the marker, one can extract the

vectors pointing in x- and y-axis of the coordinate system on the marker.

$$M = (\mathbf{x}, \mathbf{y}, \mathbf{z}) = \begin{pmatrix} x_0 & y_0 & z_0 \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{pmatrix}$$

It is necessary to calculate the coordinates (α, β) of the intersection point in the coordinate system of the marker. Since the vectors **x** and **mx** and the vectors **y** and **my** are collinear you can write:

$$\begin{aligned} \mathbf{mx} &= \alpha \frac{\mathbf{x}}{|\mathbf{x}|} \\ \mathbf{my} &= \beta \frac{\mathbf{y}}{|\mathbf{y}|} \end{aligned}$$

Now we can almost calculate the position of the intersection point, but we do not know the eye vector **e** exactly, just the direction $(0, 0, 1)^{\mathbf{T}}$ not the length. So we can write:

$$\mathbf{e} = \gamma(0, 0, 1)^{\mathbf{T}}$$

And we get:

$$\gamma(0, 0, 1)^{\mathbf{T}} = \mathbf{p} + \alpha \frac{\mathbf{x}}{|\mathbf{x}|} + \beta \frac{\mathbf{y}}{|\mathbf{y}|}$$

From this you can set up a linear equation system that can be solved due to three unknown variables α, β, γ and three equations:

$$\begin{aligned} 0 &= p_x + \alpha \frac{x_x}{|\mathbf{x}|} + \beta \frac{y_x}{|\mathbf{y}|} \\ 0 &= p_y + \alpha \frac{x_y}{|\mathbf{x}|} + \beta \frac{y_y}{|\mathbf{y}|} \\ \gamma &= p_z + \alpha \frac{x_z}{|\mathbf{x}|} + \beta \frac{y_z}{|\mathbf{y}|} \end{aligned}$$

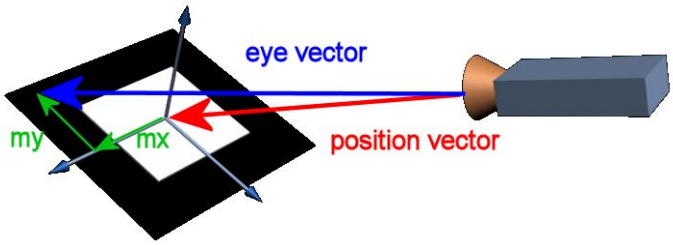


Figure 4.8: Vector chain to calculate the position where the user is looking at the shared workspace marker

This equation system can now be solved with the Gaussian elimination method. In α and β the coordinates of the point where the user is looking at are saved and passed on to the network component.

Network Communication

The Networking component is responsible for sending and receiving messages. Gaze information received from the GazeAnalyser component is sent out immediately whereas messages received from the network are parsed and stored locally so that the SceneAssembler component can poll the information when necessary.

For the network layer I used UDP-sockets that allow the broadcast of messages in subnets (see section 4.2.2). The UDP protocol is not save that means users cannot be sure that packets they send really arrive at the destination. That is no problem in this application because information is sent continuously and a loss of one packet does not matter. Even if the network communication would break down totally I implemented visual notifications as described in section 6.1.4.

As there is only one kind of message to send - the gaze information - a simple protocol could be designed. A message consists of five fields separated by #:

```
#<subject>#<object>#<extended info>#<x>#<y>#
```

`<subject>` specifies the user from whom the message is send and so who is looking.

`<object>` means either the user name of the avatar the user `<subject>` is looking at or a string representing the shared workspace marker.

`<extended info>` is set to zero (0) in case a user is looking at an avatar and one (1) if the user is looking at the shared workspace.

`<x>` is the x-coordinate of the point the user is looking at the shared marker (in the last section referenced as α).

`<y>` is the y-coordinate of the point the user is looking at the shared marker (in the last section referenced as β).

The both coordinate fields (`<x>`, `<y>`) are only valid if the `<extended info>` field is set to one (1).

This simple protocol allows to parse arriving messages efficiently. Of course optimisations could be done to compress the amount of data sent but as tests showed that was not necessary.

4.3.3 Assembling the Scene

The SceneAssembly component combines all gaze information and user position to build a scene graph that is stored in a buffer so that the Viewer component can display it.

Again I will first describe the preparations of the scene required when users are looking at each other and afterwards the preparations for the shared workspace.

Avatar to Avatar Rotation

To visualise the gaze information captured from the users the avatars should always face towards the corresponding avatar. To gain avatars turning as natural as possible they should only pitch and yaw (also called head) as you can see in figure 4.9. So we actually have to

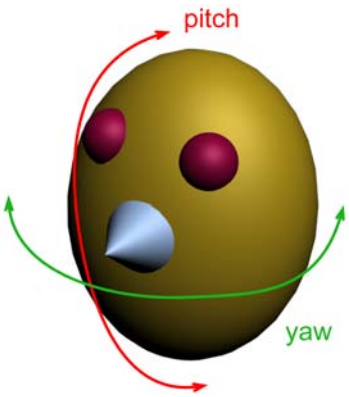


Figure 4.9: Pitch and yaw at a head

calculate the angles the avatars have to turn and cannot use other techniques as used for the arrows on the shared workspace for example as they would cause the avatar rolling.

To be even more realistic the best solution would be to just turn and rotate the head and not the whole avatar. That option was abandoned because of too much effort.

To know in which direction the avatar has to turn you need the vector **c** connecting the source and target avatar (see figure 4.10). This vector is already calculated and stored in the PoseDataBuffer component as it was used to calculate invisible marker positions.

To calculate the pitch and yaw the angles α and β have to be calculated. α is the angle between **c** and the x–y–plane and β is between **c** and the x–z–plane. To do this you have to project the connection vector in the corresponding plane. The projection in the x–y–plane is done by setting the z–component zero so you get $\mathbf{c}_{xy}$ as you can see in figure 4.11. The same is done as well with the x–z–plane for what the y–component is set zero.

$$\mathbf{c} = \begin{pmatrix} c_x \\ c_y \\ c_z \end{pmatrix} \Rightarrow \text{projection in } x\text{–}y\text{–plane} \Rightarrow \mathbf{c}_{xy} = \begin{pmatrix} c_x \\ c_y \\ 0 \end{pmatrix}$$

$$\mathbf{c} = \begin{pmatrix} c_x \\ c_y \\ c_z \end{pmatrix} \Rightarrow \text{projection in } x\text{–}z\text{–plane} \Rightarrow \mathbf{c}_{xz} = \begin{pmatrix} c_x \\ 0 \\ c_z \end{pmatrix}$$

The angles between the vectors can then be calculated with help of the scalar product and the arc cosine:

$$\begin{aligned} \alpha &= \arccos\left(\mathbf{c} \cdot \mathbf{c}_{xy}\right) \\ \beta &= \arccos\left(\mathbf{c} \cdot \mathbf{c}_{xz}\right) \end{aligned}$$

Because the arc cosine returns only values between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$ you have to take note of several case differentiations to cover the whole range of possible angles between $-\pi$ and π.

As mentioned before the avatar should not roll so the angle for the third axis is set zero. With these three angles you can build now the quaternion $\mathbf{q}_r$ describing the necessary rotation to turn one avatar towards the other [9]. To apply this rotation the original orientation **o**

[9] http://www.euclideanspace.com/maths/geometry/rotations/conversions/eulerToQuaternion

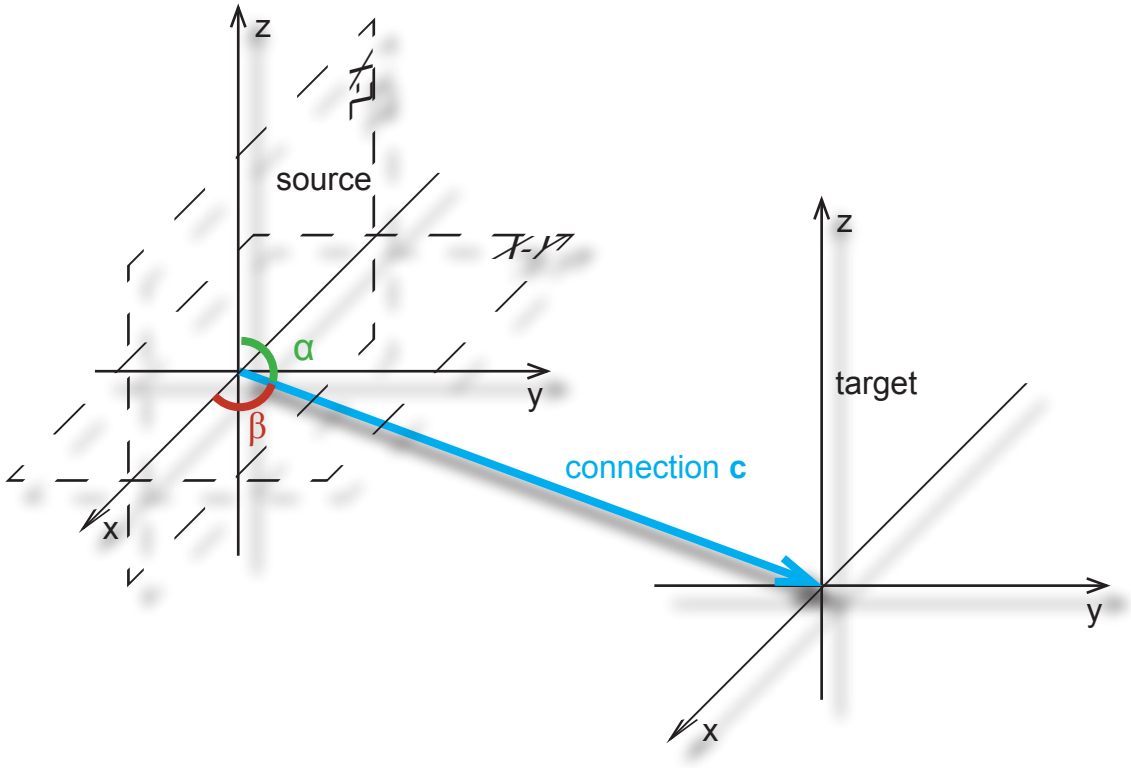


Figure 4.10: Connection vector

from the marker is multiplied with this quaternion.

$$\mathbf{o}_{final} = \mathbf{o}\ \mathbf{q}_r$$

In case a user is looking at an avatar that is looking at him the connection vector **c** is calculated a bit different. Since the avatar has to face directly at the camera you only have to take the versed direction of the position vector **p** of the avatar. As this vector is in world coordinates you have to rotate this vector now with the inverted quaternion representing the orientation **o** of the marker.

$$\mathbf{c} = \mathbf{o}^{-1}\mathbf{p}\mathbf{o}$$

The position of the avatars are fixed by the markers and so do not change.

Workspace Preparations

The object on the shared workspace is not modified at all. The position and orientation of the marker are used. Whereas it is more difficult to put the arrows on the right position and orientate them in a way that they are pointing from the avatars. The rotation of the avatars looking at the shared workspace are also a bit different.

In the gaze information message received from another user the offset on the shared workspace plane is given by (α, β). You can extend that pair now with zero as z–component to the vector $(\alpha, \beta, 0)^{\mathbf{T}}$ and rotate it around the quaternion $\mathbf{o}_{sw}$ representing the orientation of the shared workspace marker. Adding this rotated vector to the position vector $\mathbf{p}_{sw}$ of the

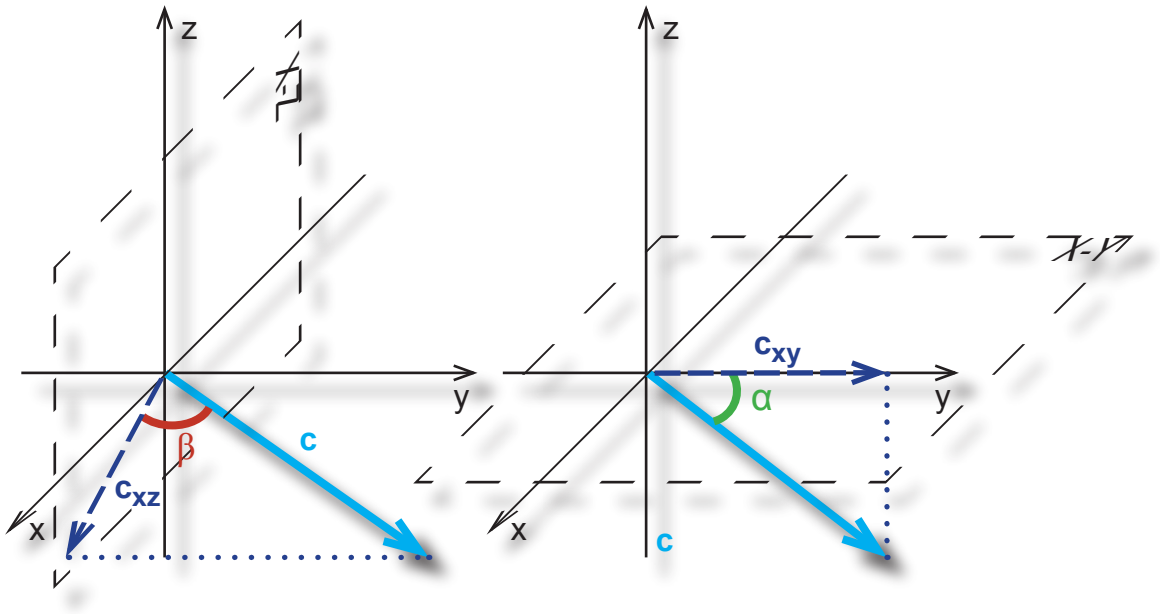


Figure 4.11: Projecting the connection vector to calculate α and β

shared workspace you get the point $\mathbf{p}_{arrow}$ on the shared workspace where the remote user is looking.

$$\mathbf{p}_{arrow} = \mathbf{p}_{sw} + \mathbf{o}_{sw} \begin{pmatrix} \alpha \\ \beta \\ 0 \end{pmatrix} \mathbf{o}_{sw}^{-1}$$

Now you need the connection vector $\mathbf{c}_w$ in world coordinates between the point $\mathbf{p}_{arrow}$ and the position of the avatar. This vector is used for the rotation of the avatar. So the avatar really rotates towards the point the user is looking at and not only towards the shared workspace. This enables that in case an avatar is moving a lot because the user is probably searching for something another user can recognise that by just seeing the avatar and so provide help.

For the rotation of the arrow you need a vector from the point $\mathbf{p}_{arrow}$ to the position of the avatar in the coordinate system of the shared workspace. The PoseDataBuffer component stores already the translation vector **t** from the centre of the shared workspace to the avatar. If you subtract the offset $(\alpha, \beta, 0)^{\mathbf{T}}$ from the gaze information you get the vector **c**.

$$\mathbf{c} = \mathbf{t} - (\alpha, \beta, 0)^{\mathbf{T}}$$

Imagine the arrow that should be displayed is directed at the moment parallel to the normal $\mathbf{n}_{xy}$ on the x–y–plane with the peak downwards (see figure 4.12). To rotate the arrow in the way that its axis is laying directly in the vector **c** you need to calculate the bisecting line **b** between the vectors **c** and $\mathbf{n}_{xy}$. Therefore you normalise the vector **c** to $\mathbf{c}_N$ and add it to the

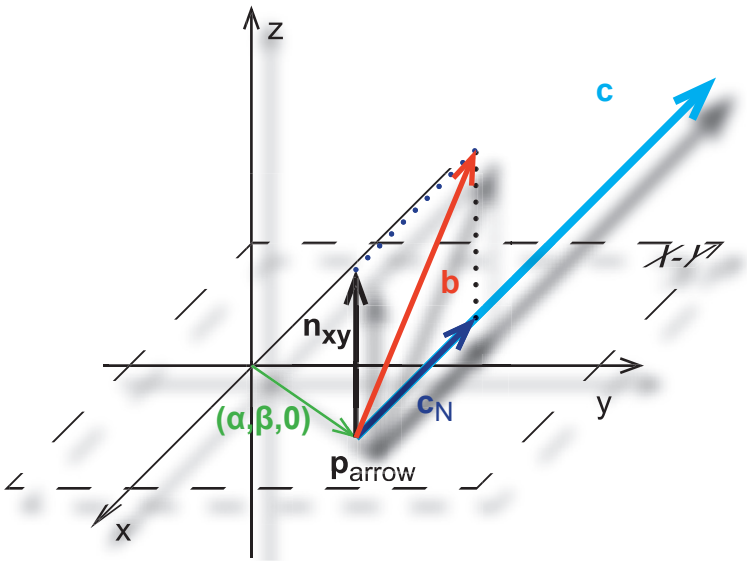


Figure 4.12: Calculating the bisecting line to rotate the arrow around it

normal $\mathbf{n}_{xy}$.

$$\begin{aligned} \mathbf{b} &= \mathbf{c}_N + \mathbf{n}_{xy} \\ &= \frac{\mathbf{c}}{|\mathbf{c}|} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \end{aligned}$$

To rotate the arrow you have to build a quaternion $\mathbf{q}$ that rotates the arrow around the bisecting line $\mathbf{b}$ about π.

$$\mathbf{q} = \begin{pmatrix} \mathbf{b}_x \sin(\frac{\pi}{2}) \\ \mathbf{b}_y \sin(\frac{\pi}{2}) \\ \mathbf{b}_z \sin(\frac{\pi}{2}) \\ \cos(\frac{\pi}{2}) \end{pmatrix} = \begin{pmatrix} \mathbf{b}_x \\ \mathbf{b}_y \\ \mathbf{b}_z \\ 0 \end{pmatrix}$$

This quaternion $\mathbf{q}$ is now multiplied with the orientation of the shared workspace marker and stored together with the position of the avatar in a buffer so that the Viewer component can visualise the whole scene.

4.3.4 Displaying the Scene

The Viewer component is build on the OpenVRML[10] runtime. It is an open source implementation of the VRML standard[11]. OpenVRML allows to read 3D models from files. So it is possible to design the avatars, arrows and 3D models for the shared workspace before hand in a 3D modelling tool and not to define them in the source code.

The viewer component loads all prepared objects with the corresponding *PoseData* from the SceneAssembly component. Then it applies the translations and orientations to the mod-

[10] http://www.openvrml.org
[11] http://www.web3d.org/x3d/vrml/index.html

els and renders the objects in the scene. Furthermore it loads the current video image and shows it in the background.

CHAPTER 5

User Study

After an introduction to the theory of user studies I describe the task designed to evaluate my application and discuss pros and cons. In the end of this chapter I present the set-up of the user study.

5.1 Theoretic Issues of User Studies

It is a widely accepted statement that testing and evaluation of a software product belongs to the software life cycle. It is necessary to ensure the quality of the final product and to meet the requirements of the customer. Not only a test with the client but with the user is necessary. There are several reasons project managers find to omit tests as low budget or deadlines. In fact a good tested and evaluated product can satisfy the user much more and so excuse a later release date for example. The following listing presents briefly some reasons why to evaluate and how [PRS+94].

Understanding the real world: This step is done normally before or during requirements elicitation and should provide the developer a detailed knowledge of the work environment the system is deployed in a final stage.

Comparing designs: To evaluate different solution options or design alternatives and compare them against each other tests are run with mock-ups for example.

Engineering towards a target: A certain metric is made up to test if the product fulfils certain requirements. This can be a product of a competitor or an upgrade that should iron out weak spots of a former version.

Checking conformance to a standard: Here the software is tested to accomplish certain standards and wide-accepted rules.

There are two different ways of evaluating a system: laboratory studies and field studies [DFAB97]. Studies in a special equipped *laboratory* allow more detailed analysis of the users.

Special designed rooms with audio/video recording facilities and two-way mirror allow the user to act undisturbed and fully concentrated in the task. This can introduce problems as well because users act different in a foreign space. This cannot happen in a *field* study where observers visit the user in her "natural environment´´ to watch the users interacting with other persons and the system.

5.1.1 Evaluating Implementations

For evalutating implementations of a system three methods are known: *experimental evaluation*, *observational techniques* and *query techniques*.

Experimental Evaluation

This method allows to prove with a special designed experiment, a hypothesis or claim. The experiment has to be designed carefully. Important factors of an experiment are the subjects, variables and the hypotheses.

Often you cannot have the actual users to test a system. So you have to choose persons of the same type, so same age, education, skills, experience etc., always depending what is important for the experiment. You should also try to have a group of at least ten subjects because you can not draw stable conclusions with a lower count.

To get comparable results it is important to classify two types of variables. *Independent variables* are defining the different conditions the subjects will execute the task for example one time with and one time without a mouse as input device. The number of values possible for such a independent variable is called *level*. *Dependent variables* can be measured in the experiment and will be evaluated to prove the hypothesis. The time needed to fulfil a task could be such a dependent variable.

The *hypothesis* is the fact you want to prove with help of the experiment. So you define the conditions by varying the independent variables. You have to predict the result of the experiment and then evidence that by disproving the null hypothesis.

During designing the experiment it is important to choose the best suiting *experimental method*. In a *between-groups* design every subject participates in one condition of the experiment. To actually make the results comparable the subject is tested in two conditions: one constant that every subject has to do and one modified. So differences between the subject can be eliminated. This method has the advantage that users do not learn during the tests. On the other hand you need a lot more subjects and differences because of the individuality of the subjects can bias results. The alternative is the *within-groups* design. Here every subject is tested under every condition. So you get a much higher sample rate and direct comparable results with less subjects. To avoid learning effects the order of the conditions should vary.

To draw conclusions the collected data (*dependent variables*) has to be *statistically evaluated*. You have to distinguish by means of the data which method to use. First it is important to know if you got *discrete variables* or *continuous variables* and how they are distributed. If they are normally distributed you can apply *parametric* tests and otherwise *non-parametric* tests. Depending on these criteria you can then choose the correct statistical method.

Observational Techniques

Observation techniques are applied in two ways. One is to watch how a user interacts with a system. This is often done during the design process of an application to derive requirements. Observers visit the users in their normal environment.

Another version is to encourage the users to think aloud. So observers get a better idea why the user is doing something. To really make the user to a collaborator the test is called *cooperative evaluation* [MWDH93]. It is also possible that the observer asks specific questions during the task to get a deeper insight.

Several techniques allow the observers to protocol actions and behaviour of the user. *Paper and pencil* are simple but still one of the fastest methods to take notes. Of course you can use *audio and/or video recording* for detailed evaluation afterwards but one have to take care that evaluating this material can be very time consuming. Another possibility is *logging computer interaction*. So keystrokes or command execution order can easily be logged if taken to account early enough in the development process. For long going evaluations it is also possible to ask the *user to take notes* what provides already interpreted records.

Query Techniques

Querying techniques provide a more detailed insight of the user's view of a system. So it is possible to access directly the opinion of users about the product. Especially *interviews* are a powerful tool to discover flaws in a product as investigators can start with general questions and then get deeper at critical issues. Whereas *questionnaires* are more inflexible they provide an easy access to numerous people and are much more comparable than interviews. Questionnaires can exist of several parts. In the *general* part details about the subject as age and occupation are requested. In a *scalar* part user can mark how far they agree or disagree with a statement on a scale normally reaching from one to seven. Other scales are possible but suffer from disadvantages as too rough or too fine results. In *multiple-choice* tests several responses are proposed to a question and the user has to mark the one best suiting to her opinion. Another possibility is to provide the user several items and ask her to *rank* them regarding some criteria. A good possibility to cover issues that cannot formulated or answered exactly is to provide *open-ended* questions. For example you can ask for improvement ideas.

5.1.2 What is the Appropriate Evaluation Method?

You can give no answer to this question because every study is different. It is often advisable to make clear what aspects should be investigated and to pose a set of questions before hand that should be answered by the results of the study.

In my user study I combined different methods to cover as many points of interest as possible. In the next sections I will first present the task I made up and then describe my methods to gather data and analyse it.

5.2 Task

I first list some requirements the design of the task should meet and then describe the task that I developed and which requirements are met.

5.2.1 Requirements

The design of the task required to consider the two aspects: gaze between the users should be necessary as well as gaze at the shared workspace. So the task should encourage the users to discuss with and look at each other. During collaboration on the shared workspace it should be essential to know where the other users are looking at or about what they are talking. To prevent that users are describing where they are looking at the object should be as hard to describe as possible.

To test the benefits of an *Augmented Reality* application it would be useful to make the users interact with a real world object.

5.2.2 Design of the Task

For the design of the task and the development of the conditions to test the subjects I worked together with Mark Billinghurst.

The exercise was to plan a trip through a city. To support this task the users could see a city map on the shared workspace. On the map there were ten sightseeing spots marked with pictures of the attraction standing upwards away from the map (see figure 5.1). On a sheet of paper the users got information in note form to the sightseeing spots and the picture that was on the map. So the users got the mapping between the description on the paper and the spot on the map very intuitively.

The users should imagine that they only have half a day in the city and can only visit five of the ten attractions listed for the city. So they had to discuss which places to pick and make up a tour.

There were no time constraints to the users but they were encouraged to discuss actively so that they would look at each other.

The cities were all east European cities. Since most of the subjects were New Zealanders the cities were unfamiliar and the names of the attractions were hard to pronounce. So it was easier for the users to point and not explain what they wanted to visit.

Due to the description of the sightseeing on a real sheet of paper the mixture of real and virtual world is achieved.

5.3 Conditions and Hypotheses

For acquiring the hypotheses and the further evaluation I collaborated with Martin Wagner.

I decided to evaluate the task in a *within-group* test. One reason was because there was no access to a huge group of subjects and also little time. On the other hand it was easier to compare the collected data. There were three conditions to test:

1. The avatars were fix and no arrows were displayed.
2. The avatars rotated but there were still no arrows on the shared workspace.
3. The avatars rotated and arrows were displayed on the shared workspace to visualise the exact point where the user were looking.

I tested ten groups with each three persons and all participated in the three tasks with random order so learning effects were excluded. Thirty subjects is a relative big sample as ten subjects is a typical sample size for a user study evaluating a first prototype.

Figure 5.1: Example of the virtual city map with pictures of the sightseeings as photographs standing away from the map

The aim of the user study was to test if and how the rotation of the avatars and the arrows could help the users. The following hypotheses were tested:

1. The application increased the satisfaction of users in the collaboration session.
2. It is easier to complete the task with more awareness information.
3. Users pay more attention to the other users.
4. The users communicated and understood their ideas better with awareness support.
5. The avatars help with turn taking.

To evaluate these hypotheses a questionnaire was designed that the users had to complete after every task. The questionnaire posed questions as *"I could easily understand my partners' ideas"* or *"My opinions were clear to the others"* to evaluate the hypotheses. Before the tests started every user had to answer some questions regarding her background and personnel details. Every user had to sign as well a participant consent that confirmed that she took part voluntarily and allowed to record audio and video. This is a necessary part of every user study. The whole questionnaire is presented in detail in Appendix B.

5.4 Setup

To record all three users at the same time with audio and video it was necessary that all three subjects were placed in one room. To give the feeling that they are really in different rooms they were separated by curtains. That the observer could see the same view of the scene as the users can see in their handheld display in front of every user a monitor was placed directed towards the video camera (see figure 5.2).

Figure 5.2: Photograph of the set-up: three users collaborating with the handheld displays and their view of the scene in the monitors

Every user was equipped with two markers for the avatar representation and one multi-marker as anchor for the virtual city map.

A first pilot study was carried out with two groups. The intention of a pilot study is to discover problems with the set-up and to provide a first look at the data recorded. Since no problems occurred the study was continued and the two groups where merged to the real study.

In addition to audio/video recording and the questionnaires application data was logged. From every user the event where or at whom she is looking at was recorded. That are exactly the events which are broadcast over the network. These data were collected on an extra computer on which the video camera was attached directly as well. So it was not necessary to change tapes in the camera and copy them afterwards again.

CHAPTER 6

Results

This chapter is divided in two separate parts. The first part discusses problems, improvements and design aspects that came up during the implementation phase and a heuristic evaluation. In the second part the evaluation of the user study is described.

6.1 Implementation

During the implementation and a heuristic evaluation some ideas came up to improve the application. There was the design of the avatars, visualisation aspects of the awareness and some feedback and interaction concerns for using the application.

6.1.1 Heuristic Evaluation

Nielsen and Molich define in [NM90]:

> Heuristic evaluation is done by looking at an interface and trying to come up with an opinion about what is good and bad about the interface.

Normally several evaluators inspect the interface with either a list of criteria or by their own intuition. The evaluators can take down notes and write a final report or a developer is observing the evaluator and taking down the issues that came up. The first version leads to a formal report that can be followed to improve the system. The second version takes load of the evaluator and makes the results accessible much faster, because the reports do not have to be reviewed again.

Five evaluators is claimed to be a number to get a good ratio of found problems with the interface and costs caused by the evaluators. Studies showed that one or two persons is not enough and too much evaluators do not find significant more problems that would be worth to pay more.

To find the improvements listed in the following sections I inspected the interface alone.

6.1.2 Avatar Design

I started with a simple avatar built just with a cone as torso, a sphere as head and a pyramid as nose. This model was enough to represent a user and even the direction the avatar was turned to could be recognised with help of the nose. But during a first test it became clear that just different colours of the avatars were not enough and so I mapped a picture of the user on the head (see figure 6.1a). That the face was really visible on the head and not covered by the virtual nose the nose pyramid was rendered half transparent (see figure 6.1b). In this stage the personalised avatar visualised the direction the user was gazing.

But the first version still had drawbacks. So the picture of the user's face was distorted because of the sphere as underlying model. So the sphere is cut now and the face is projected on the flat plane (see figure 6.1c). The nose as direction indicator is dismissed since the shape of the torso is changed as well. Arms are symbolised and provide so even direction information when the nose of the old representation would not be visible because it is hidden behind the head when viewed from the back (see figure 6.1d).

Figure 6.1: Different avatar representations form different perspectives: a) side view of the first version with photograph on the head, with the nose indicating the direction; b) avatar from front with the half transparent nose; c) second version of the avatar with a plane face and no nose (front); d) side view of the second version.

6.1.3 Smooth Avatar Turning

Another aspect I recognised during the tests was that the avatars flipped from one direction to another when a change in the gaze happened. This flipping disturbed the human-like representation and so I implemented the SLERP algorithm. With help of SLERP it is possible to interpolate between two quaternions. When a gaze change occurs the last orientation is stored and in the next few scene assembling steps the orientation approximates the new orientation. Now the avatars turn smoothly between two gaze directions and provide a more natural feeling.

6.1.4 Feedback

As users can of course not see their own avatar turning they do not know how their gaze is represented. That is difficult because users feel unconfident when they do not know how others receive their acting. To solve this problem two aspects are distinguished again: gaze towards another user and gaze towards the shared workspace.

On the shared workspace not only the arrows representing the gaze of other users is displayed but as well an arrow for the user's own gaze. The arrow is directed perpendicular to the plane so that the peak of the arrow points directly to the spot where the user is looking at (see figure 6.2a).

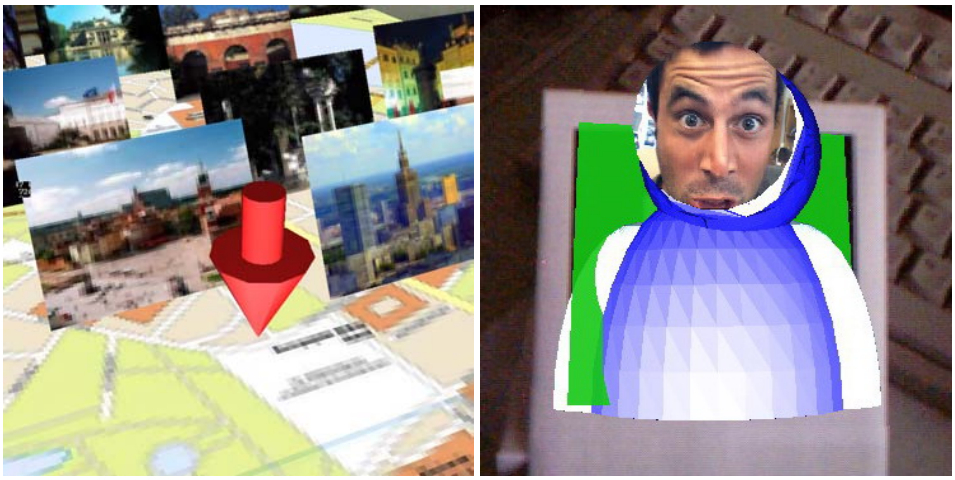


Figure 6.2: Visual application feedback: a) the own gaze on the shared workspace represented with an arrow perpendicular to the plane; b) a green square illustrating that a user is in the gaze.

To allow the user to recognise that she is looking at another user a green square shows up behind the avatar in gaze (see figure 6.2b).

6.1.5 Gaze Timeout

It is very important to not only visualise the gaze of users but this gaze should be up-to-date as well. Normally the gaze information is broadcast fast enough but there are cases that the gaze cannot be captured and thus not send out. For example, the user is not looking at markers because she has put the handheld display aside.

So it is necessary to provide the user information if the gaze that is represented by the avatars is still valid. One possibility was to rotate the avatar in a position that indicates clearly that the user is not looking anywhere specific. This could be turning away from the map or lay the avatar down. But so the last orientation gets lost what can be unfavourable. So I decided to turn the avatars half-transparent after a certain time-out (see figure 6.3). So users can still see the last orientation and are still aware that the gaze information is old.

6.1.6 Extended Workspace Awareness

The design of the arrows allows to find them quick and easily. But imagine the case that a arrow is hidden behind some three dimensional object. Another case could be that a user is working on a part of the shared workspace and so does not see the whole one. So she cannot see the arrows all the time and is not aware where the users are looking at or working.

To bridge that gap additional lines are drawn in the scene connecting every avatar with the corresponding arrow (see figure 6.4). These lines do have two effects. The first is that the mapping between the arrow and the avatar is even easier now because of the direct visualisation. Second, users find the arrows faster because they just have to follow the lines

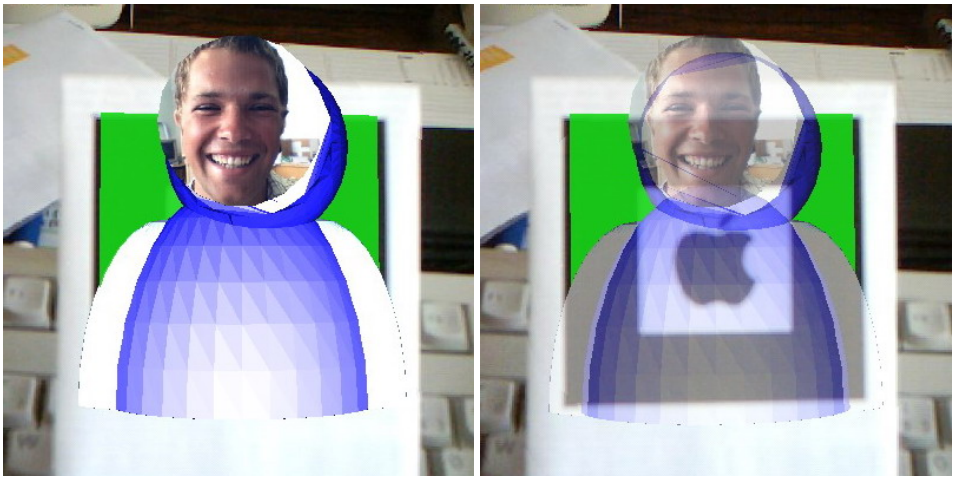


Figure 6.3: Avatar with two different states: a) normal avatar state with up-to-date gaze information; b) half-transparent avatar signalling that the gaze information is old.

from the avatar to the arrow and even get an idea of the position when the arrow is not visible. Third users are aware about where the other users are working approximately when they see these lines crossing their field of view. Also from the movement of the lines the activity of the other users can be derived as well.

6.1.7 Freezing the Arrows

The movement of the arrows is very sensitive. This is necessary for a smooth flow and the gaze of users is pretty quick. Since the arrows are also a pointing device it would be desirable that users can fix their own arrow, so they can point and look at the same time where the other users are looking without moving their own arrows away again.

That is also important for the user study where users put their handheld displays aside to probably read the notes on the paper about a sightseeing.

To allow this fixing of the own arrow the user can click the left mouse button one time to freeze the movement of the arrow and click a second time to release it again. A better solution would be to mount a switch on the handheld display so you can utilise it with the same hand you are holding the handheld display.

6.2 Evaluation of the User Study

The evaluation of the user study was developed in cooperation with Martin Wagner. We evaluated basically the questionnaires and compared them against the logging data of the application and observations done during the user study.

The questionnaire was designed that cumulating two to four questions should permit to test one of the five hypotheses proposed in section 5.3 (see Appendix B for details). So for every thesis we got six to twelve values (three conditions times two to four questions) from thirty users. To test the correctness of these hypotheses we first averaged the results of the connected questions of every user and got for every thesis three statistical series each of thirty values.

Figure 6.4: Whole set-up with arrows

To compare these data we used the *analysis of variance (ANOVA)* that allows us to make a decision if there are differences between the series. It does not say where the differences are or which row is different from the others. For an introduction to ANOVA see Appendix A.

The main results as means, variance and significance level are abstracted in table 6.1. The values are scaled to fit on a scale from one to seven. Additionally to the evaluation of the questionnaire the times needed to fulfil the task are shown in the last row.

Overall the results show that especially the arrows bring a big advantage. So the users were more satisfied with the collaboration session and could better communicate their ideas when they had the arrows as additional gaze information. Although the means would seem that the users could complete the task easier we have to reject the hypothesis because of a p-value of 0.13. The third and fourth hypothesis cannot be accepted as well since the p-value specifying the significance of the ANOVA test was too high. For the third hypothesis you can recognise that users spent less attention to their partners when they had the arrows. This can indicate that they were more concentrated on the map and did not actually perceive the arrows as their partners. For the fourth hypothesis it is similar as the second. You can recognise again an slightly higher mean for the third condition but the ANOVA does not allow to accept that hypothesis. Whereas we can definitely accept the hypothesis that the users perceived an increased awareness.

The users reported in a short informal interview after the tests that they could easier communicate their ideas, could concentrate better on the task and did not have to describe what they were looking at or which point they were talking about. The statement *"The arrows on the map helped me a lot."* in the questionnaire was accepted with a value of 6.4 on a scale up

	Overall F Test	No avatar rotation, no arrows	Avatar rotation, no arrows	Avatar rotation and arrows
User satisfaction	$p < 0.0003$	$5.0(1.1)$	$5.0(1.0)$	$6.0(0.8)$
Ease of task completion	not sig.	$5.0(1.3)$	$4.9(1.1)$	$5.5(1.0)$
Degree of attention paid to partners	not sig.	$4.7(0.9)$	$4.7(0.9)$	$4.5(1.0)$
Clarity of expression	not sig.	$5.3(0.9)$	$5.3(0.9)$	$5.7(1.1)$
Awareness of turn taking	$p < 0.0002$	$1.8(1.1)$	$2.9(1.6)$	$3.3(1.4)$
Time in Seconds	not sig.	$270(67)$	$310(160)$	$300(120)$

Table 6.1: Results of the user study: For every hypothesis you can find the mean and variance (in brackets) for every condition. The last row presents the time used to solve the task.

to 7. So the arrows really helped the users to fulfil their task and made it more comfortable to them.

As the order of the conditions was random groups that started with arrows responded that they really missed them in the succeeding conditions.

Another interesting observation revealed that most of the time users only looked at the shared workspace. This fact was proved with three methods. During the test it was obvious that users only looked at the shared workspace by just following the video screens with the view of the users. In the questionnaire users answered to *"I almost only used the shared workspace"* with a value of 5.6 that they agree with this statement. The logging data confirmed that as well. With help of the logging data we could measure the time users where gazing at the avatars (7.7%), the shared workspace (86.3%) and somewhere else (6%).

Although users did not look at the avatars often to see at whom they were turned the avatars were used as anchors for the arrows to recognise to whom the arrow belonged. So users did not have to ask *"Whose arrow is the red one?"*, for example. In most of the cases they only looked the first time at the avatars to learn the mapping between arrow and user. Afterwards the users could remember this connection.

Considering the measured time we can recognise that there is no improvement in the speed-up of finding a decision. But if we have a closer look at the data it is sticking that it took the groups with the arrows either much longer or much less time. We found that users on the one hand came to a faster conclusion because of improved pointing abilities and a common sense where they were and what they were doing. On the other hand groups discussed more lively the alternatives and so it took them more time. So either a fast or a well discussed solution can be found with that application.

CHAPTER 7

Conclusion

After a short outline how the outcome of this work could influence current research I give ideas how the application could get improved and what other issues could be investigated in future with modifications to the application.

7.1 Contribution to Current Research

This work presents a new approach to enhance remote teleconferencing with awareness information. To provide gaze awareness between users and on a shared workspace *Augmented Reality* techniques were used. As presented in chapter 3 there is research going on to use *Augmented Reality* for teleconferencing but aspects of gaze awareness were left aside. In the area of Collaborative Virtual Environments the research of awareness is more emphasised and several techniques developed. Compared to this work the important fact is that users are not interacting in a virtual space but can work in their accustomed real environment that is augmented with avatars and virtual objects.

Due to *Augmented Reality* it was possible to develop an application with a seamless transition between the remote participants represented as virtual avatars and the local, real workspace.

In a final user study the satisfaction of the users and how *Augmented Reality* can help to convey gaze awareness in teleconferences was investigated. Therefore a task was designed and with help of questionnaires, application logging data, video/audio recording and interviews evaluated.

The results showed that the acceptance by the users was high and especially the attempt to visualise the gaze on a shared workspace was very successful. The awareness of the users was significantly increased and users were more satisfied with the collaboration session. Users did like to control their pointers just with their gaze. That is important and should be considered in further research. Developers should analyse if gaze can be a possible input device. Gaze cannot only be captured by tracking in direction of the users view but also with already commercially available eye trackers. So the interaction would be less disturbing than a HMD or handheld display.

This work showed that for special tasks video streams are not necessary to convey gaze awareness. Video streams can be desirable for more informal meeting situations or to get to know partners. For remote conferences that should solve certain problems specialised applications can be more suiting.

7.2 Future Work

In this section I present some possible modifications of the application that either lead to improved usability or offer new research issues.

7.2.1 Improvements of the Application

At the moment the network communication and persistent data management is very rudimentary implemented. So it is necessary to store all 3D model files, photos of the users (to personalise the avatars) and the configuration in the file system accessible for every user. It would be preferable if pictures of the users would be spread via the network in the beginning of the session. So no time-consuming copying of files per hand would be required. Furthermore it is necessary right now to specify all participating users at every user in a XML-file (see appendix C). To overcome these problems all theses files are stored on a central network drive that can be accessed by all users. So the configuration can be changed on one central place.

Another point is to improve the 3D models. It would be nice if only the head of the avatars would rotate. This is not possible right now since the whole avatar is loaded as one VRML model. In addition some modifications would be necessary to allow the use of multiple objects on the shared workspace. You could either develop an interaction metaphor to change the 3D model on the shared workspace or just introduce multiple paper cards for the shared workspace. In the second solution you have to take care that the gaze is always directed to the correct shared workspace object. So you always need to know where every shared workspace and every paper card is. With multiple paper cards it is likely that users will just put uninteresting markers a side and the application does not know where they are. So keeping the gaze up-to-date could become pretty difficult.

7.2.2 Possible Research Modifications

During the study of the literature, implementation, observations during the user study and the feedback after it some aspects of further research came up that I present in the following.

Avatar Position

In the current version of the application the avatars are fixed in their positions to the paper cards. There are two other possible solutions.

One would be to display the avatar relative to the shared workspace with respect to the actual positions of the users. So if the user moves with the handheld display closer towards the shared workspace the avatar moves closer as well. And when the user rotates the paper card the avatar turns around it as well. A problem could be that if all users are looking at the shared object from the same direction they do not see each other because they are all beside

each other (see figure 7.1). To allow gaze awareness between users could be difficult as the marker of the shared workspace is probably not in the view of the user any more.

Figure 7.1: All three avatars with the same view direction on the shared workspace

Another interesting idea is to put the avatars directly on the map instead of arrows. The avatars could turn on the map always towards the other avatars depending at whom the users are looking.

Awareness Models

The gaze information on the shared workspace are displayed as arrows and lines from the avatars. This is similar to the nose ray already discussed in section 2.1.2. The arrows sometimes conceal important things on the shared workspace. So it would be another possibility to implement awareness representations as the view cone or the headlight proposed in [DG01]. This could be especially interesting if real three dimensional objects are rendered on the shared workspace and the calculation of the exact spot where the user is looking becomes more complex than in two dimensions. The size of the view cone or of the light spot could also indicate how close users are to the shared workspace and so provide the information how much a user can see of the objects.

In combination to position the avatars freely around the shared workspace depending on their view it is another aspect to investigate how the well known focus/nimbus theory could help [BF93]. The focus is the direction a user is looking. Every user has the nimbus that can be perceived by another user. So A can be aware of user B because B is in A's focus or A is in B's nimbus. The problem with avatars being next to each other and facing towards the same direction and cannot see each other could probably be solved. When focus is visualised as a half transparent cone and the nimbus as an half transparent sphere around the avatar (see figure 7.2) users could sense other avatars next to them if they are in such a half transparent body.

Mobile Application

Today mobile phones get more and more powerful and almost all are equipped with video cameras. Henrysson and Ollila [HO04] have just ported the ARTOOLKIT to a mobile phone. As the paper cards with the markers can be taken everywhere you could imagine to have a conference with multiple users on your mobile phone. Instead of the handheld display the

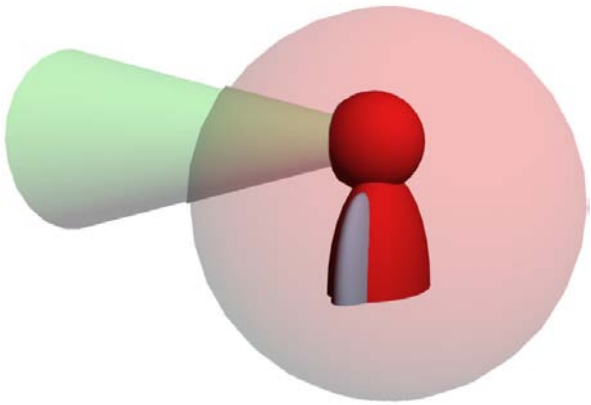


Figure 7.2: Avatar with nimbus (red sphere) and focus (green cone)

mobile phone can be used. Because of the simple protocol the network traffic is very small and so even useful for radio connection as General Packet Radio Service (GPRS) or Universal Mobile Telecommunication System (UMTS) ([SW05]).

Audio Channels

A problem in teleconferences is that side conversations between two persons that the others cannot hear are impossible. Right now there is no audio support implemented in the application. A possible feature to implement could be to enable a side conversation when two users look at each other for a certain time.

APPENDIX A

ANOVA - Analysis of Varianz

To evaluate the data collected during a user study some statistical knowledge is necessary. This chapter should provide a brief introduction of the methods we used for our evaluation. This is not a complete discussion of ANOVA but will present the needed background information to understand the analysis we did. For more information we refer to [Byr86] for example.

To prove the hypothesis or the null hypothesis of an experiment the statistical series of the collected data must be compared. To compare two sets of samples the well-known two-sample t test can be applied [Smi97]. In the user study we conducted were three different conditions and so three groups of samples to compare. For this case and even more populations the ANOVA is a powerful tool.

In the further part of this chapter we use the following syntax: an experiment with k conditions or *treatments* lead to k populations each with size n_i where i varies from 1 to k. x_{ij} is the value of the jth subject in the ith population. $T_i = \sum_j x_{ij}$ is the sum of all values within one population and $\bar{x}_i$ the mean.

The main idea of ANOVA is to compare the variance within the groups with the variance between the groups. The *sum of squares for error (SSE)* is the sum of all variance within the groups.

$$SSE = \sum_i \sum_j (x_{ij} - \bar{x}_i)^2$$

The *sum of squares for treatments (SSTR)* is the variance of the means of each population

$$SSTR = \sum_i n_i (\bar{x}_i - \bar{\bar{x}})^2$$

where $\bar{\bar{x}}$ denotes the weighted average of all means $\bar{\bar{x}} = \sum_i n_i \bar{x}_i / \sum_i n_i$ which can be transformed to $\bar{\bar{x}} = \left(\sum_i \sum_j x_{ij} \right) / N$ as the weights n_i all sum up to the number of subjects N. The last interesting variable is the *total sum of squares (SSTO)* that describes the difference between the sum of all values x_{ij} squared and the square of the sum of all values T divided by the total count of values N. At the same time the *total sum of squares* is the sum of SSTR

and SSE.

$$SSTO = \sum x^2 - \frac{T^2}{N} = SSTR + SSE$$

The degree of freedom (DF) is the "number of observations minus the number of parameters estimated from the data" [ASE00]. This will probably become clearer in the example a bit later.

By dividing the square sums through the degree of freedom you can now calculate the *mean square error (MSE)* and the *mean square for treatments (MSTR)*:

$$\begin{aligned} MSTR &= \frac{SSTR}{k-1} \\ MSE &= \frac{SSE}{N-k} \end{aligned}$$

With this information you can now built the ANOVA table (see table A.1).

SOURCE OF VARIATION	DF	SS	MS
Treatments	$k-1$	SSTR	MSTR
Error	$N-k$	SSE	MSE
Total	$N-1$	SSTO	

Table A.1: The ANOVA table: degree of freedom (DF), sum of squares (SS), Mean Squares (MS)

MSTR and MSE are two independent estimates for the variance of the experiment. To test if the estimates MSTR and MSE are similar and so the null hypothesis (the means of all populations are the same and so no difference between the treatments) can be accepted or must be rejected the quotient of MSTR and MSE is build and compared to the F distribution. The critical F value is dependent of the two degree of freedom values $k-1$ and $N-k$: $F_{crit}(k-1, N-k)$. If the quotient of MSTR and MSE is bigger than F_{crit} the difference of the estimates is too big, the null hypothesis must be rejected and so the treatments are different.

$$F = \frac{MSTR}{MSE} = \begin{cases} < F_{crit} \Rightarrow \text{accept null hypothesis} \Rightarrow \text{all treatments are equal} \\ > F_{crit} \Rightarrow \text{reject null hypothesis} \Rightarrow \text{at least two treatments are different} \end{cases}$$

Example Here is an example to illustrate the problem: Imagine we want to test two different drugs supporting a diet. For the study 30 subjects are divided in three groups or populations. Two groups get the two different drugs and the reference group gets placebos. After some weeks the weight loss is measured. Table A.2 shows the lost weight of the test subjects. The sum and mean of each population are calculated as well.

	1	2	3	4	5	6	7	8	9	10	T_i	$\bar{x}_i$
drug A	5	7	11	3	4	4	2	5	8	3	52	5,2
drug B	11	7	5	6	5	7	1	3	4	0	49	4,9
placebo	3	2	3	1	-1	4	2	4	4	5	27	2,7

Table A.2: Weight loss of subjects in a diet study

To calculate the SSTR we first need the mean over all values $\bar{\bar{x}} = \frac{T}{N} = (52 + 49 + 27)/30 = 4,2\bar{6}$. Now we build the sum of squares over the means $\bar{x}_i$ and $\bar{\bar{x}}$ and get $SSTR = 37,267$.

For the SSE we build the differences between the values in a population and the mean of the population, square them and sum them all up to $SSE = 186,6$.

Since we have three observations and only one parameter to test we get two degree of freedom for SSE. So we can calculate MSE and MSTR and also the F-value. From a table we can read the value $F_{crit}(2, 27)$ with a significance of 5% as 3,35. Table A.3 presents all the calculated data.

SOURCE OF VARIATION	DF	SS	MS	F-VALUE	P-VALUE	F_{crit}-VALUE
Treatments	2	$37,267$	$18,63$	2,69	0,085	3,35
Error	27	$186,6$	$6,91$			
Total	29	$223,867$				

Table A.3: The ANOVA table for the example extended with the F, P and F_{crit} value

As we can see the F-value is smaller than the critical F-value and so we have to accept the null hypothesis and we cannot say that the treatment with drugs is more efficient to loose weight. The P-value is the actual significance of the F-value.

TIP Spreatsheet applications as Excel[1] and OpenOffice[2] have built-in macros to calculate the ANOVA table as presented in table A.3.

[1] http://office.microsoft.com/excel
[2] http://www.openoffice.org

APPENDIX B

Questionnaire

The questionnaire, I used for my user study, is divided in three parts: a general part to gather personal information about the subjects, a constant part that is used to compare the three conditions and a third with some additional questions to get extra feedback.

B.1 General

The general part was answered before the tests and used on the one hand to collect personal information of the subjects as age, sex and writing hand. I also checked for my study how much experience the users had before with *Virtual Reality* , *Augmented Reality* and conferencing applications. The subjects had to mark the amount of time they already spend with the technology on the following scale:

- None
- Less than 5 hours
- 5-50 hours
- 50-500 hours
- More than 500 hours

It is also important to clarify the participants of the user study and make them sign the participant consent. Therefore I posed the following statements on the questionnaire:

- I have been informed on the procedure and purpose of the study and my questions have been answered to my satisfaction.
- I have volunteered to take part in this study and agree that during the study information is recorded (audio and video as well as my interaction with the system). This information may only be used for research and teaching purpose. I understand that my participation in this study is confidential. All personal information and individual results will not be released to third parties without my written consent.

- I understand that I can quit my participation at any time, including the withdrawal of any information I have provided.

B.2 Constant Part

The constant part was used to compare the impression of the users after the different conditions. All questions had been the same and answered after every test from every subject.

To test the five hypothesis presented in section 5.3 two to four statements had been developed for each. In the following list you can see which statements were posed for which hypothesis.

1. *The application increased the satisfaction of users in the collaboration session.*
   - I felt the collaboration session overall went great. We had no problems and did not struggle to complete the task.
   - I could easily understand my partners' ideas.
   - I could easily communicate my ideas.
2. *It is easier to complete the task with more awareness information.*
   - The task was easy to complete.
   - The task required little effort.
   - I did not have to concentrate very hard to do the task.
3. *Users pay more attention to the other users.*
   - I paid close attention to the other individuals.
   - The other individuals paid close attention to me.
   - The other individuals tended to ignore me.
   - I tended to ignore the other individuals.
4. *The users communicated and understood their ideas better.*
   - My opinions were clear to the others.
   - The opinions of the others were clear.
   - The others understood what I meant.
   - I understood what the others meant.
5. *The avatars help with turn taking.*
   - I was gazing at the avatars often.
   - The avatars helped me with turn taking.

The statements for hypothesis three and four are taken from [BHG01].

Additionally there were four statements that should indicate problems with the application if some would appear. This is important because you can check if outlier are probably caused by problems and so you can exclude them from the test result. The questions were:

- The application was working stable.
- It reacted fast enough to work with the application.
- I understood the meaning of the visualisation.
- The feedback of the application was easy to understand.

B.3 Additional Questions

After the condition two and three, when the avatars rotated, the following statements were posed to get information if the idea of turning avatars is useful and recognisable:

- I got the information at whom everyone was looking.
- I could recognise at whom the avatars were rotated.

To test if the arrows helped in the users opinion after the third condition the following questions were asked:

- The arrows on the map helped me a lot.
- I almost only used the “shared workspace”.

APPENDIX C

Configuration and Setup

This appendix provides all necessary information how to configure the application and run it.

C.1 Configuration

The configuration of the application is dependent on the 3D models and one XML-file.

C.1.1 How to store the 3D models

The application uses the VRML support implemented in the ARTOOLKIT version 2.71. To load a VRML file a description file must be provided. This file has the following structure:

```
filename.wrl
0.0 0.0 0.0       # Translation
0.0 0.0 0.0 0.0   # Rotation
1.0 1.0 1.0       # Scale
```

Instead of `filename.wrl` you have to insert the file name of the VRML file. The rotation you have to provide in axis-angle representation.

C.1.2 XML-Configuration file

There is one configuration file for every user. In figure C.1 you can see an exemplary listing of such a configuration file. In the `MySelf` tag you specify the local user name. With the attributes `avatar` and `sharedObject` of the tag `Rotation` you can control if the avatars should turn and if arrows should be displayed on the shared workspace.

In the `Users` element you specify for every user a `User` tag. The meaning of the attributes you can see in table C.1.

With the attributes of the `SharedObject` element you can specify the settings for the shared workspace. Depending if you want to use a multimarker you set the `multiMarker`

`id`	the user name
`markerFile`	the name of the pattern file of the marker
`avatarFile`	the name of the file used to import the VRML model (see section C.1.1)
`avatarFileTransparent`	same as `avatarFile` but with the transparent avatar
`coneFile`	the name of the file describing the VRML model of cone used for that user

Table C.1: Meaning of the attributes of the `User` element

attribute to `true` or `false` and the `markerFile` attribute to the multimarker description file or the pattern file for the ARToolKit. The `avatarFile` specifies the description of the VRML model again. In the `HighLightObject` element you specify with help of the attribute `avatarFile` the model that appears at the avatar when the user is looking at it. At the moment this is just a green plane but you could imagine to load a half transparent sphere.

C.1.3 Pictures on the avatars

The pictures of the users that should be displayed on the avatars are specified in the VRML-model files. It is enough to exchange the picture files to put another users face on the avatar. Furthermore I wrote a small programme that shows the video stream of the web cam and takes photos when one of the buttons `'g'`,`'k'` or `'s'` is pressed. The photos are automatically saved on the correct position in the file system to be displayed on the avatars.

C.2 Starting and Running the Application

The application can easily be started by executing the file `AACmain.exe`.

When starting the application without any parameter the configuration file has to be named `AACConfigurator.xml`. Otherwise you can also pass the file name of the configuration file as first parameter at the start of the programme.

Table C.2 lists all possible keyboard keys or mouse buttons you can interact with during the application is running.

space bar or left mouse button	freezes the own arrow on the shared workspace
'l', 'L' or right mouse button	toggles the light on and off
'f' or 'F'	toggles between full screen and window size
'q' or 'Q'	quits the application

Table C.2: Interaction possibilities when the application is running

```
<AAC>
  <MySelf>Kasperl</MySelf>
  <Rotation          avatar="true"
                     sharedObject="true" />
  <Users>
    <User            id="Kasperl"
                     markerFile="Data/patt.hiro"
                     avatarFile="Wrl/kasperl.dat"
                     avatarFileTransparent="Wrl/trans_kasperl.dat"
                     coneFile="Wrl/cone_kasperl.dat" />
    <User            id="Gretl"
                     markerFile="Data/patt.apple"
                     avatarFile="Wrl/gretl.dat"
                     avatarFileTransparent="Wrl/trans_gretl.dat"
                     coneFile="Wrl/cone_gretl.dat" />
    <User            id="Seppl"
                     markerFile="Data/patt.kanji"
                     avatarFile="Wrl/seppl.dat"
                     avatarFileTransparent="Wrl/trans_seppl.dat"
                     coneFile="Wrl/cone_seppl.dat" />
  </Users>
  <SharedObject      multiMarker="true"
                     markerFile="Data/multi/marker.dat"
                     avatarFile="Wrl/warsaw.dat" />
  <HighLightObject   avatarFile="Wrl/highlight.dat" />
</AAC>
```

Figure C.1: Listing of the XML configuration file

Bibliography

[ABB+01] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *IEEE Comput. Graph. Appl.*, 21(6):34–47, 2001.

[AHNS99] R. Azuma, B. Hoff, III Neely, H., and R. Sarfaty. A motion-stabilized outdoor augmented reality system. In *Virtual Reality, 1999. Proceedings., IEEE*, pages 252–259, 1999.

[ASE00] R. A. Armstrong, S. V. Slade, and F. Eperjesi. An introduction to analysis of variance (anova) with special reference to data from clinical experiments in optometry. *Ophthalmic Physiol*, 20(3):235–241, 2000.

[Azu95] R. Azuma. A survey of augmented reality, 1995.

[BBF+95] Steve Benford, John Bowers, Lennart E. Fahlén, Chris Greenhalgh, and Dave Snowdon. User embodiment in collaborative virtual environments. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 242–249, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[BBK+01] Martin Bauer, Bernd Bruegge, Gudrun Klinker, Asa MacWilliams, Thomas Reicher, Stefan Riss, Christian Sandor, and Martin Wagner. Design of a component-based augmented reality framework. In *Proceedings of the International Symposium on Augmented Reality (ISAR)*, October 2001.

[BF93] Steve Benford and Lennart E. Fahlén. A spatial model of interaction in large virtual environments. In *ECSCW*, pages 107–, 1993.

[BFS03] Istvan Barakonyi, W. Frieb, and Dieter Schmalstieg. Augmented reality videoconferencing for collaborative work. *Proceedings of the 2nd Hungarian Conference on Computer Graphics and Geomtery, Budapest, Hungary*, May 2003.

[BGBG95] Ronald M. Baecker, Jonathan Grudin, William A. S. Buxton, and Saul Greenberg. *Readings in human-computer interaction. Toward the Year 2000*. Morgan Kaufmann Publishers, 2nd edition, 1995.

[BHG01] Frank Biocca, Chad Harms, and Jenn Gregg. The networked minds measure of social presence: Pilot test of the factor structure and concurrent validity. Technical report, Media Interface and Network Design (M.I.N.D.) Labs, Dept. of Telecommunication, Michigan State University, East Lansing, 2001.

[BID98] Scott Brave, Hiroshi Ishii, and Andrew Dahley. Tangible interfaces for remote collaboration and communication. *CSCW '98: Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 169–178, 1998.

[BK99] Mark Billinghurst and Hirokazu Kato. Artoolkit. http://www.hitl.washington.edu/artoolkit/, 1999.

[BK00] Mark Billinghurst and Hirokazu Kato. Artoolkit documentation. http://www.hitl.washington.edu/people/grof/SharedSpace/Download/-ARToolKit2.33doc.pdf, November 2000.

[BKP01] M. Billinghurst, H. Kato, and I. Poupyrev. The magicbook - moving seamlessly between reality and virtuality. *Computer Graphics and Applications, IEEE*, 21(3):6–8, 2001.

[BR94] Bill Buxton and Ron Riesenbach. Hydra. http://www.dgp.toronto.edu/tp/techdocs/Hydra.html, 1994.

[BVBC04] Volkert Buchmann, Stephen Violich, Mark Billinghurst, and Andy Cockburn. Fingartips: gesture based direct manipulation in augmented reality. *GRAPHITE '04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and Southe East Asia*, pages 212–221, 2004.

[Byr86] Donald R Byrkit. *Statistics today: a comprehensive introduction*. Benjamin-Cummings Publishing Co., Inc., Redwood City, CA, USA, 1986.

[CBB+99] J. Cassell, T. Bickmore, M. Billinghurst, L. Campbell, K. Chang, H. Vilhjailmsson, and H. Yan. Embodiment in conversational interfaces: Rea. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 520–527, New York, NY, USA, 1999. ACM Press.

[CT03] Siemens CT. Head-up-display. http://www.siemens.com/Daten/siecom/-HQ/CC/Internet/Research_Development/WORKAREA/fue_inno/-templatedata/Deutsch/file/binary/themen_fo_usability_2_lg_1151213.jpg, 2003.

[DB92] Paul Dourish and Victoria Bellotti. Awareness and coordination in shared workspaces. In *CSCW '92: Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, pages 107–114, New York, NY, USA, 1992. ACM Press.

[DBBO96] P. Dillenbourg, M. Baker, A. Blaye, and C. O'Malley. The evolution of research on collaborative learning. In H. Spada and P. Reinmann, editors, *Learning in Humans and Machines. Towards an Interdisciplinary Learning Science*, pages 189–211. Oxford: Elsevier Science., 1996.

[DFAB97] Alan Dix, Janet Finlay, Gregory Abowd, and Russell Beale. *Human-computer interaction*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1997.

[DG01] Jeff Dyck and Carl Gutwin. Where are you and what can you see? maintaining location awareness in collaborative 3d workspaces. Technical Report HCI-TR-2001-02, University of Saskatchewan HCI Lab, 2001.

[GG02] Carl Gutwin and Saul Greenberg. A descriptive framework of workspace awareness for real-time groupware. *Computer Supported Cooperative Work (CSCW)*, 11(3):411–446, 2002.

[Gru88] Jonathan Grudin. Why cscw applications fail: problems in the design and evaluation of organization of organizational interfaces. In *CSCW '88: Proceedings of the 1988 ACM conference on Computer-supported cooperative work*, pages 85–93, New York, NY, USA, 1988. ACM Press.

[HBC+92] Hewett, Baecker, Card, Carey, Gasen, Mantei, Perlman, Strong, and Verplank. Acm sigchi curricula for human-computer interaction. Technical report, ACM SIGCHI, 1992. Chairman-Thomas T. Hewett.

[HBR04] Jörg Hauber, Mark Billinghurst, and Holger Regenbrecht. Tangible teleconferencing. In Masood Masoodian, Steve Jones, and Bill Rogers, editors, *APCHI*, volume 3101 of *Lecture Notes in Computer Science*, pages 143–152. Springer, 2004.

[Hen04] Michi Henning. A new approach to object-oriented middleware. *IEEE Internet Computing*, 8(1):66–75, 2004.

[HK99] Joerg Heuer and Andre Kaup. Global motion estimation in image sequences using robust motion vector field segmentation, 1999.

[HO04] Anders Henrysson and Mark Ollila. Umar: Ubiquitous mobile augmented reality. *To appear at 3rd Mobile Ubiquitous Multimedia 2004, Washington, USA*, pages 41–45, 2004.

[IKA94] Hiroshi Ishii, Minoru Kobayashi, and Kazuho Arita. Iterative design of seamless collaboration media. *Commun. ACM*, 37(8):83–97, 1994.

[Int] Clifton Office Interiors. Boardroom. http://www.cliftonoffice.co.uk/boardroom.html.

[IT93] Ellen A. Isaacs and John C. Tang. What video can and can't do for collaboration: a case study. In *MULTIMEDIA '93: Proceedings of the first ACM international conference on Multimedia*, pages 199–206, New York, NY, USA, 1993. ACM Press.

[KB99] Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *IWAR '99: Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality*, page 85, Washington, DC, USA, 1999. IEEE Computer Society.

[KBMT01] H. Kato, M. Billinghurst, K. Morinaga, and K. Tachibana. The effect of spatial cues in augmented reality video conferencing. *In Proceedings of the 9th International Conference on Human-Computer Interaction*, pages 478–481, 2001.

[LBK04] Gun A. Lee, Mark Billinghurst, and Gerard Jounghyun Kim. Occlusion based interaction methods for tangible augmented reality environments. In *ACM Siggraph International Conference on Virtual Reality Continuum and its Applications in Industry (VRCAI 2004)*, pages 419–426, Singapore, 15-18th June 2004. ACM Press.

[LTJ] Alexander Lamaison, Stuart Trail, and Kevin Judson. Medical computing website. http://www.doc.ic.ac.uk/project/2003/163/.

[Löw05] Felix Löw. Improving augmented realty table top applications with hybrid tracking. Master's thesis, Technische Universität München, May 2005.

[MD03] John McKenzie and Doreen Darnell. The eyemagic book. a report into augmented reality storytelling in the context of a children's workshop. Technical report, Human Interface Technology Laboratory New Zealand, 2003.

[MSW+03] Asa MacWilliams, Christian Sandor, Martin Wagner, Martin Bauer, Gudrun Klinker, and Bernd Bruegge. Herding sheep: Live system development for distributed augmented reality. In *ISMAR '03: Proceedings of the The 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality*, page 123. IEEE Computer Society, 2003.

[Muk02] R. Mukundan. Quaternions: From classical mechanics to computer graphics, and beyond. In *Proceedings of the 7 th Asian Technology Conference in Mathematics*, 2002.

[MWDH93] A. Monk, P. Wright, L. Davenport, and J. Haber. *Improving your human computer interface: a practical approach*. Prentice-Hall, 1993.

[NM90] Jakob Nielsen and Rolf Molich. Heuristic evaluation of user interfaces. In *CHI '90: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 249–256, New York, NY, USA, 1990. ACM Press.

[NY99] U. Neumann and S. You. Natural feature tracking for augmented reality. *Multimedia, IEEE Transactions on*, 1(1):53–64, 1999.

[PIHP01] James Patten, Hiroshi Ishii, Jim Hines, and Gian Pangaro. Sensetable: a wireless object tracking platform for tangible user interfaces. *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 253–260, 2001.

[PRS+94] Jenny Preece, Yvonne Rogers, Helen Sharp, David Benyon, and Tom Carey. *Human-Computer Interaction*. Addison-Wesley Longman Ltd., Essex, UK, UK, 1994.

[PW97] Alex Pang and Craig Wittenbrink. Collaborative 3d visualization with cspray. *IEEE Comput. Graph. Appl.*, 17(2):32–41, 1997.

[RF00] J. P. Rolland and H. Fuchs. Optical versus video see-through head-mounted displays in medical visualization. *Presence: Teleoperators & Virtual Environments*, 9(3):287–309 (23), June 2000.

[RG96] Mark Roseman and Saul Greenberg. Teamrooms: network places for collaboration. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 325–333, New York, NY, USA, 1996. ACM Press.

[ROW+03] H. Regenbrecht, C. Ott, M. Wagner, T. Lum, P. Kohler, W. Wilke, and E. Mueller. An augmented virtuality approach to 3d videoconferencing. In *ISMAR '03: Proceedings of the The 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality*, page 290, Washington, DC, USA, 2003. IEEE Computer Society.

[SBA92] Abigail Sellen, Bill Buxton, and John Arnott. Using spatial cues to improve videoconferencing. *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 651–652, 1992.

[Sch02] Kjeld Schmidt. The problem with 'awareness': Introductory remarks on 'awareness in cscw'. *Comput. Supported Coop. Work*, 11(3):285–298, 2002.

[SFH+02] Dieter Schmalstieg, Anton Fuhrmann, Gerd Hesina, Zsolt Szalavári, L. Miguel Encarnaçã, Michael Gervautz, and Werner Purgathofer. The studierstube augmented reality project. In *Presence: Teleoperators & Virtual Environment*, volume 11, pages 33–54. MIT Press, 2002.

[SIKH86] David Canfield Smith, Charles Irby, Ralph Kimball, and Eric Harslem. The star user interface: an overview. *National Computer Conference*, 55:383–396, 1986.

[Smi97] Peter J Smith. *Into Statistics*. Springer, 1997.

[SW05] Michael Siggelkow and Martin Wagner. An AR teleconferencing system for increased remote awareness and low bandwidth connections. In *submission to ISMAR 05*, 2005.

[SWC76] J. Short, E. Williams, and B. Christie. *The Social Psychology of Telecommunications*. Wiley, New York, 1976.

[TKHS04] Yu-Pao Tsai, Ching-Che Kao, Yi-Ping Hung, and Zen-Chung Shih. Real-time software method for preserving eye contact in video conferencing. *J. Inf. Sci. Eng.*, 20(5):1001–1017, 2004.

[Ver99] Roel Vertegaal. The gaze groupware system: mediating joint attention in multiparty communication and collaboration. *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 294–301, 1999.

[VSvdVN01] Roel Vertegaal, Robert Slagter, Gerrit van der Veer, and Anton Nijholt. Eye gaze patterns in conversations: there is more to conversational agents than meets the eyes. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 301–308, New York, NY, USA, 2001. ACM Press.

[VWS02] Roel Vertegaal, Ivo Weevers, and Changuk Sohn. Gaze-2: an attentive video conferencing system. In *CHI '02: CHI '02 extended abstracts on Human factors in computing systems*, pages 736–737, New York, NY, USA, 2002. ACM Press.

[WBL+04] Eric Woods, Mark Billinghurst, Julian Looser, Graham Aldridge, Deidre Brown, Barbara Garrie, and Claudia Nelles. Augmenting the science centre and museum experience. Technical report, Human Interface Technology Laboratory, New Zealand (HITLabNZ), New York, NY, USA, 2004.

[Wil91] Paul Wilson. *Computer Supported Work*. Kluwer Academic Publishers, 1991.