Chapter 2

Related Work

This chapter provides a survey on related work. In the first section many different interaction techniques are introduced, discussed, and compared with each other. It is shown which techniques that are used in two-dimensional graphical user interfaces (GUIs) are usable or portable to the three-dimensional Virtual Environments. Additionally new three-dimensional techniques are introduced. The last section deals with Collaborative Virtual Environments (CVE). It provides a survey on selected CVEs and tele-immersion applications. In addition interaction in these environments is discussed.

2.1 Interaction Techniques

Interaction techniques denote the possibility how different input devices are used in order to fulfill a special interaction task. *Interaction tasks* denote the entries of a unit of information by the user [39]. From the programmer's point of view interaction techniques can be interpreted as mechanisms with which the user inputs (events) are recognized and processed by the system. If user inputs consist of a sequence of different events, the interaction techniques are called complex according to [25, 26].
Basic interaction tasks are: selection, positioning, text input, numeric input, and confirmation [38].

When *selecting*, the user chooses an element out of a list of elements. The list is either of a fixed or variable size. In 2D GUIs selection can be implemented by a menu in combination with a mouse or a stylus. An alternative is to enter a name with a keyboard into a character input field.

For the *positioning* the user selects a point (x,y) in two dimensions, (x,y,z) in three dimensions and moves the selected object to this point. Positioning is usually performed by a locator device like the mouse or using a touch screen with a pen-like device as a stylus. Another possibility is to enter numeric co-ordinates using a keyboard.

Text input denotes the input and the manipulation of character strings. The keyboard is the dominant input device for text input. It is, however, also realizable using microphones and speech recognition software.

Numeric input (quantification) denotes the declaration of a value. It is usually performed using a keyboard. Sliders are an alternative if the numeric interval is static and not too large.

Confirmation is type independent and can be realized by any user input. Typically confirmation is performed by voice, by a pressed key of the keyboard or button of a special tool (mouse, joystick etc.). Confirmation mostly makes use of so-called command buttons. In non-graphical systems pressing the white space or the enter key is used in order to confirm actions.

2.1.1 Basic Interaction Techniques

Input and output devices are a very important factor influencing the choice of an interaction technique. For output devices as the monitor and the Head Mounted Display in combination with space mouse, pinch and data gloves many different interaction techniques are under current investigation [10, 13, 33, 46, 62]. Interaction techniques especially developed for rear-projection based displays like CAVEs and Workbenches in combination with, for example, pen-like stylus input devices, force feedback or even new input devices are seldom [33, 42, 56].
In the following sections different interaction techniques are introduced and discussed with respect to the completion of the five basic interaction tasks mentioned above. It is discussed which of these techniques are usable in Virtual Environments according to their advantages and disadvantages.

Command Languages

Command languages are the oldest interaction techniques. They enable a user to perform the interaction tasks: text and numeric input. To apply this interaction technique a keyboard is needed as an input device.
Command languages have a universal expressiveness. In common 2D applications and desktop Virtual Environments command languages cannot be used very efficiently. To use them efficiently is only possible for expert users as com-

mand languages are not easily learnable and they do not offer control outside their sphere of action. Another problem is that the user is forced to switch the communication modi when typing in commands for interaction.
The use of keyboards as input devices for command languages in most rear-projection based Virtual Environments is not very comfortable and satisfying. One reason is that the user is forced to turn the head towards the keyboard and away from the data set when entering commands. Another reason is the difficulty to position the keyboard outside the disposal when it is not needed. Especially CAVEs, Powerwalls and Cylindrical display systems would need to have extras racks in order to position a keyboard which then destroy the feeling of immersion.

Virtual Pick Rays

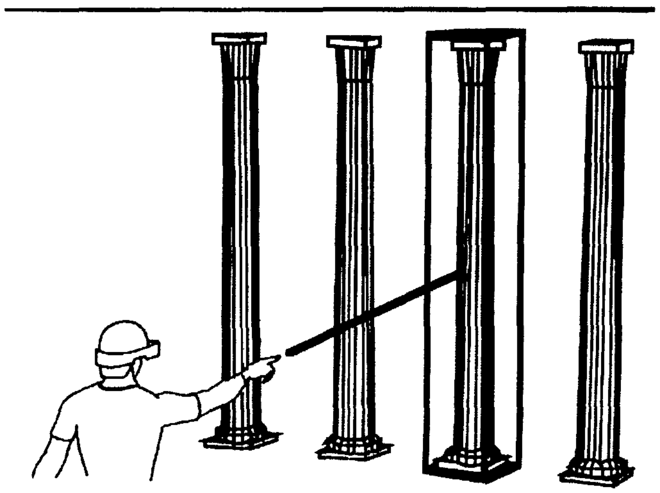

Figure 2.1: Interaction with a virtual pick ray. [Mine et al., [71]]

Virtual pick rays enable a user to select data from a distance. It can be used for selection, confirmation, and positioning.
The metaphor of using a ray is the three-dimensional analogon to the usage of the two-dimensional mouse pointer.
The pick ray interaction technique is difficult to use with 2D graphical user interface and desktop Virtual Environments since the user is sitting too close to the display system.
In rear projection based Virtual Environments as well as with Head Mounted Displays the virtual pick ray is an interaction technique that is easy to use in order to fulfill interaction tasks. For applying this interaction technique when working with these display systems input devices such as the stylus or gloves

can be used (see Figure 2.1).
Usually the interaction with the virtual pick ray such as selecting and picking is easy to learn and apply. A general disadvantage when working with a pick ray is the "lever arm" [71]. The longer the distance between the picked object and the picking device (normally a stylus in the user's hand) the bigger the movements of the object according to the movements of the hand. Interaction tasks which require high interaction precision might be insufficiently fulfilled using this technique. This is especially valid as long as the picked object is not directly attached to the manipulating hand or tool.

Menus

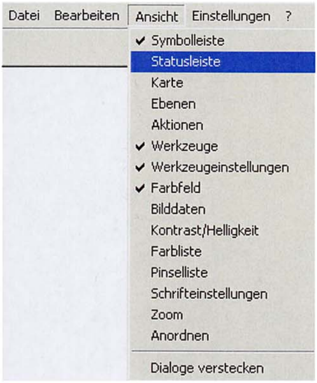

Figure 2.2: Example of a pull-down menu.

Menus enable a user to select from a list of choices. A menu consists of a list of items (icons, commands etc. according to [34]) where the size of the list is static (see Figure 2.2). They support the user's task through offering alternative actions. If the user selects the wrong element out of the list it is always possible to return and to begin the selection again. Menus can be permanently visible or they can be generated dynamically (pop-up and pull-down menus). If the list of choices is too big to show all alternatives at ones, the programmer

needs to think about other visual representations. This can either be done by partitioning the menu into another logically structured hierarchy or by representing the menu as a linear sequence of choices.
In the case of an hierarchical menu the user selects the first element from the beginning of the hierarchy. After that appears either the next list of choices or the choice itself (node of the hierarchy). An example is an hierarchical pop-up menu.
Typical WIMP (windows, icons, menus, pointing) applications usually provide a window and interface elements like menus, icons and toolbars to work on documents.
In two-dimensional graphical user interfaces pull-down menus are often aligned together with menu bars to the upper or lower edge of the screen.
General advantages in comparison to command languages are that the user only needs to visually recognize the information in a menu, whereas the user needs to recall information from its memory when using command languages. However, due to their complexity menu systems are to a certain degree contradictory to the demand for minimizing the search time while looking for a specific function or tool.
In Virtual Environments static menus have the disadvantage of limiting the interaction space and occluding parts of the data set. One of the first approaches which used pull-down menus in Virtual Environments is described by Jacoby et al. in [59]. More issues related the use of menus in Virtual Environments can be found in Darken's work [29]. In his thesis the focus is on visibility of menus and readability of fonts in VEs. In addition, guidelines and principles on menu placement in VEs are suggested by the author.
In Figure 3.6 of section 3.5 a so-called ring-menu is introduced which is developed within this thesis. It is similar to Liang's ring menu [68]. One ring menu is assigned to one data set. When a user asks the data set for its functionality the menu appears and shows a list of choices representing operations that are applicable to the data set. The menu is attached to the user's hand position. It only follows the translation of the user's hand whereas the rotation of the user's wrist is used to intersect the "cake pieces" with the pick ray. Thus selection of operations is possible. The advantages are that the ring menu is generated dynamically and only exists visually as long as the user needs it. As the menu is attached to the hand the user does not need the change the viewpoint towards the menu and away from the data. Unfortunately this involves that the ring menu is aligned between the user and the data set as it is used in a three dimensional space. For not occluding parts of the data the menu is designed to be almost transparent.

Buttons

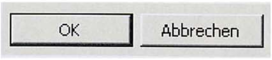

Figure 2.3: Example of a button pair.

Buttons are usually represented by oval graphical forms including text or icons. Pressing a special mouse button above these buttons enables a user to execute certain actions.
There exist different types of buttons: command buttons and check buttons. Command buttons are mostly used for confirmation purposes as the interaction task. Also high order commands and other procedures can be executed using command buttons.
Check buttons represent the states TRUE or FALSE. They are switchable using the mouse click. These check buttons represent a possibility for the selection between two choices.
Buttons can grouped together to button blocks (see Figure 2.4). An exclusive

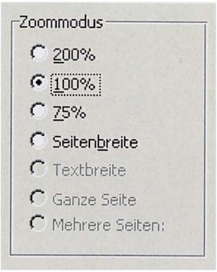

Figure 2.4: Button block with check buttons.

button block for example consists of some check buttons from which only one represents the state TRUE at a time. Invisible buttons which are positioned above other graphics are called hot-spots. With the help of those hot spots interactive graphical objects are simulated.
Buttons are usable in 2D as well as in 3D applications. In Virtual Environments for example buttons can be pressed using virtual pick rays.

However, the interaction designer has to carefully think about where to position the button in a way that they do not occlude parts of the data.
Especially in Virtual Environments the problem is the graphical representation of the buttons. Buttons represented by two-dimensional icons are smaller and do not reduce the limited interaction space as much as three-dimensional icons would. However, for the user who tries to intersect the button with a pick ray from arbitrary positions in the view frustum three-dimensional buttons are much easier to hit. As a consequence the disadvantage is the reduction of the view frustum.

Toolbars

Toolbars enable a user to customize and manipulate applications or data sets while selecting tools from a list of choices.
Toolbars denote a very common interaction technique and are used in two-dimensional GUIs as well as in three-dimensional Virtual Environments.
Even two-dimensional toolbars are possible in a Virtual Environment. However, here occurs the same problem as already discussed for the buttons and menus. Is the toolbar positioned fixed in front of the user at the RWB for example a two-dimensional representation is conceivable.
Within this thesis a toolbar has been designed and implemented which allows the user to select generic operations and apply them to the data set (see Figure 3.7 in section 3.5). This toolbar is positioned in front of the user and thus does not occlude any part of the data. A button block with five buttons on top is chosen as geometrical form for the toolbar. Each button represents another operation, such as drag, zoom, rotate etc..
The basic operations are positioned so that they are always visible and accessible. This supports the work flow since they are frequently used during the interaction with the Virtual Environment.

Sliders

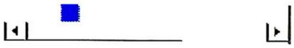

Figure 2.5: Slider.

Sliders enable a user to determine a value within a certain interval. They are used for quantification. Scroll bars are a special implementation of sliders. With their help it is possible to change the visible part of an in principle infinite

screen.
They are used in window-based systems and graphical editors. Sliders can also be used for interaction tasks in Virtual Environments. But the representation of the sliders presents the same problem as it did for the buttons.
The interaction designer needs to decide whether a two-dimensional or a three-dimensional representation is preferred.

Scrollable Selection Lists

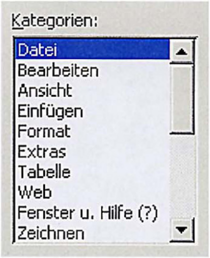

Figure 2.6: Scrollable selection list.

Scrollable selection lists enable a user to select one or more items from a list by clicking them with a mouse. They support the selection interaction task.
The size of scrollable selection lists is independent from the number of items in the list in contrast to button block and menus. With the help of the slider the user is able to access even invisible entries.
In Virtual Environments the usage of scrollable selection lists is possible but very uncomfortable and offers poor usability.
Alignment and graphical representation similar to the ones of the character input fields and buttons present the major problems when trying to use the selection lists directly in the view frustum. Scrollable selection lists on wireless touch screens or PDAs (Personal Digital Assistant like the Palm) are an alternative.

Character Input Fields

Character input fields enable a user for text input. Mostly only single characters are entered. More complex character input fields are multi-line fields with scroll bars for scrolling through the text.
Character input fields are usable in desktop based Virtual Environments with a monitor as output device and a keyboard as input device.
The keyboard is a very uncomfortable input device when combining it with rear-projection based Virtual Environments. In VEs it is better not to use a keyboard for character input. In these VEs it is possible to use wireless touch screens and PDAs with a pen for text input. These input devices do not deliver the discussed keyboard problems but they need a place to store them when they are not used.
At a Responsive Workbench the storage of these input devices does not present a problem and their usage is possible.
In a CAVE only the usage of small PDAs is possible which are storable at the user's body.

Masks and Forms

Masks and forms provide a user with static possibilities to enter information two-dimensionally.
Masks are two-dimensional indicators of the systems state which occlude the whole display screen and consist primarily of alpha numeric characters [31]. In static areas information is represented about different system states, inputs and system messages.
The work area in which the user can enter information usually is designed as a form. Forms are two-dimensional alignments of fields in which information is perceived and entered [31]. They are very suitable for grammatical formulation of orders through the combination of specifications as well as for meta communication since they are able to represent information about the system state.
In two-dimensional graphical user interfaces masks and forms are usable as long as the system state information has to appear and stay in the foreground of the disposal.
In Virtual Environments masks and forms create the same problems as menus and scrollable selection lists. Their size even aggravates the poor usability.

Dialog Boxes

Dialog boxes enable a user to execute complex operations which need a lot of parameters. They support the interaction tasks: text and numeric input,

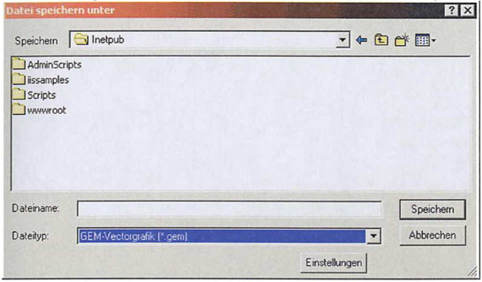

Figure 2.7: Example of a dialog box.

selection and confirmation.
Generally a dialog box consists of other elements like, for example, check buttons, numeric input fields, an OK-button for confirmation and thus approving the parameter set, or a Cancel-button for cancellation and thus ignoring the parameter set.
In either case the dialog box is closed and the dialog ends. Dialog boxes are used in Virtual Environments too.
Figure 2.8 shows a VE designer interacting over a dialog box in a virtual museum application. This VE dialog box consists of check buttons and numeric input fields. The interaction is performed using a pen-like stylus in the user's hand. For not occluding data behind the box is designed to be almost transparent inside the frame.

2.1.2 Advanced Interaction Techniques

Speech Recognition

Speech recognition is a non visible interaction technique. It enables a user to perform actions by the voice which would be far more complex to perform with other techniques.
Speech recognition contributes to user comfort and is dimension independent. This means that it can be used as interaction technique in two-dimensional applications as well as in three-dimensional Virtual Environments as a supplement to hand based interaction techniques.

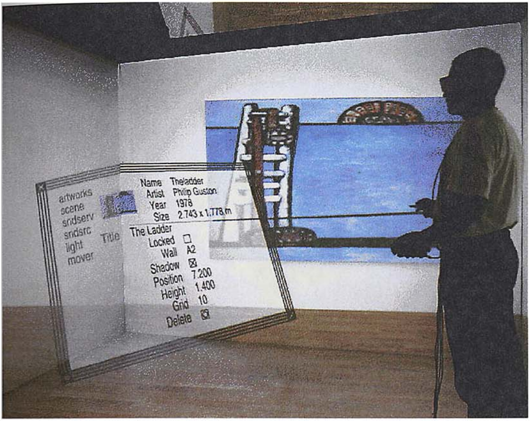

Figure 2.8: This is an example of a dialog box in a CAVE Virtual Environment. Check buttons and numeric input fields with sliders are used to enter information. In order to not occlude data the dialog box is designed to be almost transparent inside the frame.

A general problem with speech recognition software is the recognition rate. Speaker dependent speech recognition software toolkits have a much higher recognition rate. Speaker dependency, however, implies that the speaker has to create its own database with spoken phonemes.
In Collaborative Virtual Environments the interference of speech recognition with the oral communication between the remote users causes problems.
One approach to overcome this problem would be to implement a code word like "computer", which switches the system into the "Listen" mode as soon as it is recognized. The communication with the remote partner is muted and the computer waits for aural commands of the user.
Combinations of voice input with three-dimensional localized sound and two-handed manipulation in Virtual Environments are described by NASA Telep-

resence research in [36].
Systems which offer a method to embed voice communication in Virtual Environments by means of voice annotations are proposed by Harman et al. and Verlinden et al. in [52, 99]. These systems allow a set of capabilities for inserting, iconizing, playing back and organizing annotations in VEs.

Tactile/Force Feedback

Tactile and force feedback interaction techniques enable a user to touch and feel computer generated data. Tactile techniques provide information about surface attributes as a feedback to the user [82]. Force feedback techniques provide information to the user that are related to the object's mass. These techniques can be used as a supplement to visual perception [104].
Haptic interaction techniques are mostly used in desktop based Virtual Environments or together with ReachIn display systems (see section 1.1.2).
In rear-projection based Virtual Environments it is problematic to mount the feedback device when using force and tactile feedback interaction techniques [19].
When working at the Responsive Workbench with a Phantom the force feedback device is mountable as shown in Figure 1.3. If force feedback is not further needed the Phantom arm can be positioned outside the view frustum and thus does not disturb the user during the task.
Different types of haptic and tactile feedback are especially interesting for applications which have an high demand for precision. Examples for force feedback applications are flight simulators with motion platforms or force feedback driven surgical simulators.
A comprehensive introduction and reference on force-feedback can be found in Burdea's work [22].

Direct Manipulation

Direct manipulation enables a user to manipulate data sets. It groups together all interaction techniques which deal with the "handling of objects".
Metaphors dealing with the perception of two-dimensional faces and three-dimensional spaces build the basis for direct manipulation.
The main principles of direct manipulation are:

- permanent visibility of the objects to be manipulated
- permanent visibility of the actions applied to the objects
- substitution of complex commands by physical actions such as movements with an input device

- fast and reversible user interaction with direct and mostly visual feedback.

Advantages of the direct manipulation are:

- beginners can quickly learn to perform direct manipulation
- occasional users can easily recall interactions
- experts can work efficiently
- support of systematic work due to direct feedback of interactions
- low user fatigue due to easy comprehensibility, predictable system reaction and possibilities of undo operations.
- error messages are far less important for direct manipulation than for other interaction techniques.

Nowadays, direct manipulation is the most used interaction technique due to its naturalness. The computer vanished to be only a device for experts mainly because of the development of direct manipulative graphical user interfaces.
In Virtual Environments direct manipulation is the most used interaction technique and many other techniques are making use of the direct manipulative metaphor [9, 37, 54, 70, 77]. The metaphor is the basis for the development of body-relative interaction techniques which are presented in the following.

2.1.3 Body-relative Interaction Techniques

Body-relative interaction enables a user to perform actions relative to the own body. It is not a real interaction technique but rather an additional criterion for other interaction techniques.
Body-relative interaction techniques are more effective than interaction techniques relying solely on visual information as they provide a physical real world frame in which to work and they provide a more direct and precise sense of control.
During body-relative interaction a user can take advantage of *proprioception*. Here *proprioception* denotes the user's sense of position and orientation of his body and its several parts [7]. Proprioception is used again in Awareness-Action-Feedback loops of the CVE interaction taxonomy developed in this thesis (see section 3.6).
Body-relative interaction is proposed by M.Mine and used by other VE interaction designers too [54, 71, 72].
Body-relative interaction is especially important for the direct manipulation.

An example for the combination of different interaction techniques supported by body-relative interaction is constructible for a surgeon simulating an incision. The surgeon who is concentrating visually and tactilly/haptically on the incision turns the hand upside down, opens it, and calls the name of the surgical tool that is needed next.
This complex interaction task includes gesture recognition (turning and opening the hand using proprioception as he does not need to raise the head and look), speech recognition (calling the tools name) and body-relative direct manipulation (closing the hand with the tool and applying it to the data set in front). Although this interaction is very complex it is easy to learn as it exploits a real-world knowledge metaphor.

Gesture Recognition

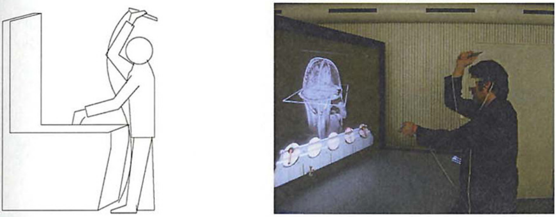
Figure 2.9: The user deselects the current tool by throwing it over the shoulder.

Gesture recognition enables a user to perform actions without any auxiliary tools. The user's gesture recognized by the system can be interpreted as selection, confirmation and positioning, including scaling, rotation and translation. Comparable with speech recognition the gesture recognition interaction technique can easily combined with other techniques that make use of input devices. Gesture recognition interaction techniques are easy to apply in desktop and rear projections based VEs. Usually a video camera is recording the user during the interaction and analysing hand and head movements in real-time [66]. Other approaches make use of the user's input devices that are usually tracked already, such as through the location sensor attached to the shutter glasses and stylus or pinch glove.
Examples for the gesture recognition based interaction techniques are the so-called "over-the-shoulder deletion" and the two-handed flying proposed by

M.Mine [70, 72] (see Figure 2.9).
The system recognizes the movement of hand over the shoulder as the command to deselect the current tool from the input device in the user's hand. In the other case the system interprets the position of the user's two hands relative to each other as the vector which determines the navigation direction. The relative distance of the hands corresponds to the absolute value of the vector and is recognized as the velocity of the movement.
In another application gesture recognition is used to facilitate the interaction

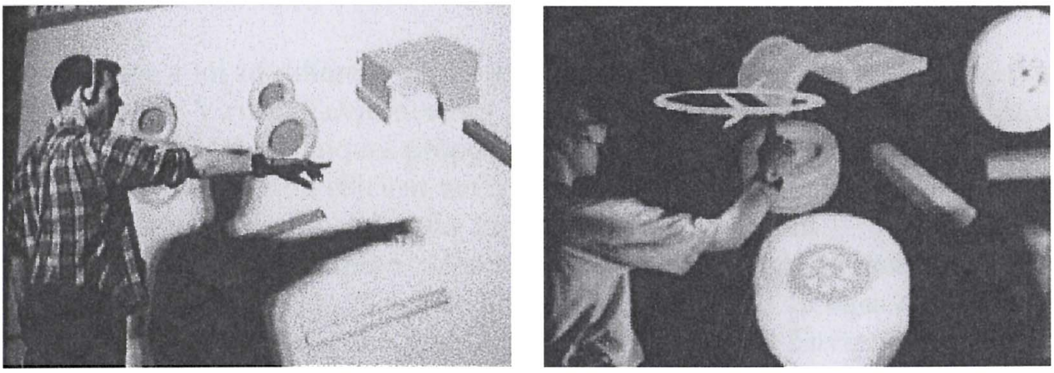

Figure 2.10: Natural speech and gesture based input used for virtual assembly simulations. In the figure on the right hand side a gesture is used for describing an object rotation. [Wachsmuth et al., [61]]

during a virtual assembly simulation as for example in [61, 100, 101]. Here the gestures are recognized as commands for "rotate", "drag", "translate" etc. (Figure 2.10).

Arm Extension

The extended arm interaction technique enables a user to select, grasp and position distant objects without navigating there. The arm extension is a pure three-dimensional interaction technique.
An example is the so-called "Go-Go" technique proposed by Poupyrev et al. and used by many others [9, 11, 77].
The "Go-Go" technique enables a user to interact within arm distance but also with distant objects deeper in the virtual scene. Therefore the position of the arms is under permanent control. A non-linear function allows the virtual arms to "grow" if needed. For this the user performs a special hand movement. If the user wants to return to the normal arm length a corresponding hand movement shortens the arms again.
Although this technique is especially developed for Head Mounted Display

based Virtual Environments it can easily be applied to rear-projection based Virtual Environments too. Applications can be virtual landscape architecture, urban planning or interior design.

2.1.4 Other Techniques and Interaction Frameworks

In combination with other techniques such as speech recognition new and intuitive three-dimensional interaction techniques are designable. The more promising ones of these techniques are presented in [9, 11, 16, 17, 18, 36, 52, 58, 77, 99].
There exist a lot of different works mainly dealing with travel and navigation techniques in Virtual Environments [33, 35]. Among them are the *Worlds in Miniature (WIM)* techniques proposed by Stoakley et al. in [93]. The focus of that work rather lies on efficient navigation and orientation, being independent of the specific activities and tasks in a Virtual Environment.
An approach that attempts to develop a taxonomy of interaction techniques can be found in [10, 11, 15]. The authors Bowman and Hodges developed a methodology to increase the usability aspects of Virtual Environments using Head Mounted Displays.
They tried to facilitate the design and evaluation of interaction techniques in VEs with this taxonomy. They distinguish between different classes of interaction techniques. These are considered to be viewpoint changes (navigation, travel), selection and manipulation.
However, their approaches are restricted to fully-immersive environments using Head-Mounted Displays. The influences of other display systems on the interaction in a Virtual Environment are not considered. In addition, their taxonomy is not able to explain the impact of interaction feedback, Virtual Environment representations, and awareness factors on usability aspects.

Conclusions

Anyway, it is possible to conclude that there are interaction techniques which are applicable in two-dimensional Graphical User Interfaces (GUIs) as well as in three-dimensional Virtual Environments (VEs). However, a direct usage of two-dimensional interaction techniques in three-dimensional VEs is not always possible. Most of the techniques have to be modified to be able to fulfill three-dimensional requirements. These requirements are for example accessibility from arbitrary points in the view frustum without occlusion, direct and precise sense of control, as well as intuitive and effective use. There are for sure many techniques available which tend to address all these requirements but new techniques especially for Virtual and Collaborative Virtual Environments have to be developed. A good reference frame for doing this represents the body-relative interaction even for rear-projection based Virtual Environments.

The introduced interaction techniques are presented according to their usability in Virtual Environments but not especially according to their usability in Collaborative Virtual Environments (CVE). The next section is presenting an overview of CVEs and explaining how the collaborative aspect in these environments is carried out.

2.2 Collaborative Virtual Environments

2.2.1 CVE applications

CVEs are multi-party Virtual Environments which allow a number of users to share a common virtual space, where they may interact with each other and the environment itself.
The problems of multiple users sharing the same workspace are already known from the field of Computer Supported Collaborative Work (CSCW) and groupware [4, 32, 57]. Some of the major problems are: the distribution of objects [98] and information as well as the delegation of rights and the representation of group structures.
Interest in *spatial approaches* to CSCW has grown over recent years. Specific examples of the spatial approach include media spaces [6], spatially oriented video conferencing [55, 57, 83], Collaborative Virtual Environments [3, 95] and tele-presence systems [66].
In contrast to CSCW systems, direct collaborative real-time interaction leads to completely new interaction possibilities, especially concurrent interaction of at least two users with one or more objects [73]. Unfortunately there is a lack of application and design support for CVEs.
In addition, most of the CVEs under investigation are web based collaborative Virtual Environments. Good overviews about examples of these desktop CVEs can be found in the following literature [3, 4, 21, 40, 69, 92, 95].
There exist only a few approaches of back projection based VEs and CVEs. Overviews about these approaches can be found in [20, 28, 43, 81].

2.2.2 CVE evaluations

When designing and implementing collaborative applications it is not sufficient to consider only technological aspects of the application. Usability as well as interactive qualities of the implementation are at least as important.

For measuring usability aspects of applications there exist four different approaches [79]. These are the:

- Interaction oriented approach
- User oriented approach
- Product oriented approach
- Formal approach

The *interaction oriented approach* is the most common one and is concerned with all kinds of usability testing with users. The *user oriented approach* tries to measure usability quality in terms of mental effort and attitude of the users. This is done by using questionnaires and interviews and is mainly used in this thesis (see chapter 6). The *product oriented approach* is concerned with measuring ergonomic attributes and thus quantitative measurements are necessary. Lastly the *formal approach* tries to simulate usability in terms of formal models [75]. This approach can be seen in the context of theory based evaluation.

Although usability testing is very important only a few authors presented evaluation studies for designed Collaborative Virtual Environments. On the other hand, existing evaluations try to draw results from assessments of 4-10 experimental subjects. The reasons are that user evaluation is a really hard task and that it is difficult to find evaluators.
In addition, many existing evaluations are focussing on very different aspects of VEs or CVEs. So for example, were Witmer and Singer one of the first who tried to measure the degree of immersion [103]. Therefore they developed a very detailed questionnaire which, in the research community, is discussed controversially [90].
Slater et al. are also trying to quantify the degree of immersion and presence [91, 92, 97]. Their approaches and experiences are restricted to desktop Virtual Environments using a normal monitor and DIVE as the software framework but no fully- or semi-immersive rear projection-based Virtual Environments.
Other usability evaluations are focussing on interaction devices and techniques as presented by Beaten and Dehoff [2] and Bowman et al. in [14]. Evaluations assessing the usability of Virtual Environment's user interfaces are presented by Hix et al. in [54]. These evaluations are restricted to stand-alone Virtual Environments using a one-sided Responsive Workbench as display system. Unfortunately the usability findings are only focussing on battle-field simulations.
The goal of this thesis is to provide design, application development and interaction support for Collaborative Virtual Environments. This includes that

the approach developed in this thesis provides means to extensively evaluate and assess the usability of CVE user interfaces and collaborative awareness factors [50].

In the following a survey on selected important CVE systems and applications is presented. The particular reason for choosing these CVE systems is the fact that they specifically address the needs of users working cooperatively together in a Virtual Environment.
Teleport and the National Tele-Immersion Initiative (NTII) uses effectively tele-presence techniques to enable face-to-face communication in an augmented office environment.
NICE for example addresses the networking aspect as well as the problems of shared manipulation. Our CVE goes one step beyond this while bringing together tele-presence and collaborative rather than cooperative interaction allowing for shared manipulation of data while preserving face-to-face communication. The following sections are presenting the mentioned CVE systems in more detail.

2.2.3 Teleport

Nowadays personal computers equipped with microphone, speakers, camera, and perhaps additional video monitors, are widely used for desktop video conferencing. Conference participants appear in windows, or on adjoining monitors, and may access shared applications shown simultaneously on each participants' screen. Several desktop video conferencing systems have been described in literature and commercial products are available. But while desktop video conferencing has certainly been shown to be useful for a variety of tasks and has many advantages when compared to earlier forms of video conferencing involving special meeting rooms, it is still recognized that there are many situations where desktop video conferencing is not appropriate.
The Teleport environment is designed to overcome the disadvantages of desktop video conferencing and to establish life-like conference sessions that bring people together as if face-to-face (see Figure 2.11).
Teleport has been developed at the Department of Virtual Environments of the *German National Research Center for Information Technology (GMD)* [20, 45]. The system is based around special rooms, called display rooms, where one wall is a view port into a virtual extension as shown in Figure 2.11. The geometry, surface characteristics, and lighting of the virtual match the real room to which it is attached. When a teleconferencing connection is established, video imagery of the remote participant is composited with the rendered view of the virtual extension. The viewing position of the local participant is tracked,

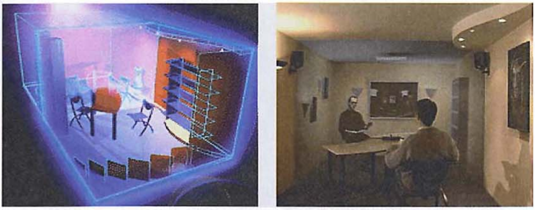

Figure 2.11: Teleport Display Room.

allowing imagery appearing on the wall display to be rendered from the participant's perspective. The combination of viewer tracking, a wall sized display, and real-time rendering and compositing, give the illusion of the virtual extension being attached to the real room. The result is a natural and immersive teleconferencing environment where real and Virtual Environments are merged without the need for head-mounted displays or other encumbering devices.
The system uses a 3x2.25m rear-projected video wall attached to a $3m^2$ room. The video wall is driven by a pair of high luminosity video projectors. Both projectors can display mid-resolution video signals and high-resolution RGB signals. A camera is placed on a stand or table and set approximately at eye height. The field of view is wide enough to take in a full upper body shot of the local participant. The viewer tracking system determines the position of the local participant within the display room, from which the viewpoint is derived.
Chroma keying is used for segmentation in order to determine the regions in the video images where the participant appears. The virtual extension is rendered from the viewpoint of the tracked participant located in the display room. Because this person is free to move within the display room, the virtual extension must be continuously re-rendered. For audio, each participant wears a small microphone. The audio streams from remote participants are mixed together and sent to speakers mounted on either side of the video wall.
The Teleport system provides video conferencing integrated into a room environment as its best. However, the participants are restricted to viewing and talking to each other. 3D interaction is not implemented at all and thus sharing of three-dimensional data is not possible.

2.2.4 Tele-Immersion Initiative

The National Tele-Immersion Initiative (NTII) is an American research community investigating into tele-presence applications together with internet2. Internet2 is a consortium led by over 180 universities working in partnership with industry and government to develop and deploy advanced network applications and technologies. Internet2 is recreating the partnership among academia, industry and government that fostered todays Internet in its infancy. The aim of NTII is to enable users at geographically distributed sites to collaborate in real time in a shared, simulated environment as if they were in the same physical room.
The participating research sites can be found under the following URL: http://www.advanced.org/teleimmersion.html
There exist plenty of different tele-immersion approaches. The sketch from University of North Carolina at Chapel Hill in Figure 2.12 provides an idea about the research activities of NTII [78].
Currently the partners within the NTII consortium are able to do stereo-video

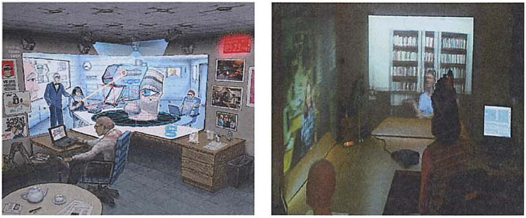

Figure 2.12: The figure sketches a vision of the office of the future. Current tele-immersion systems try to support distributed collaborative work. [Fuchs et al.,1998]

conferencing in real-time. In addition, very rudimentary collaborative interaction support is enabled. The interaction technique implemented so far is the direct manipulation with pick rays. Therefore, until now, the implementation of the office of the future can be seen as a spatial video conferencing tool rather than a collaborative office with remote access to all kinds of information.

2.2.5 NICE

NICE is the acronym for *Narrative-based Immersive Constructionist / Collaborative Environments*. It is a prototype of an educational, distributed Virtual Environment for young users using CAVEs as displays. NICE has been developed at the University of Illinois at Chicago by M. Roussos et al. [60, 80, 81]. The main constructive activity in NICE is to build and develop small local ecosystems on the bare parts of an island. The terrain serves as an open land which the child explores and decides where to plant and populate.
Various seeds for planting garden vegetables and trees are stored in crates and serve as starting points for building micro-ecosystems on the island (see Figure 2.13). When the user drops a seed on the ground, the corresponding plant, flower or tree will start to grow. The pace in which this happens can be predetermined; the user may choose to see the system grow very quickly, or, in the case of a school project, extend it over the period of a semester. The tomatoes, carrots, pumpkins and other plant objects contain a set of characteristics that contribute to their growth. They all have values for their age, the amount of water they hold, the amount of light they need, their proximity to other plants of their kind. These values determine the health of the plant and its size. Visual cues aid the child in determining the state of a plant or flower. When the cloud has been pouring rain over it for too long, the plant opens an umbrella, when the sunlight is too bright, it wears sunglasses.
Sound in the environment enriches the surroundings in a variety of ways. Dif-

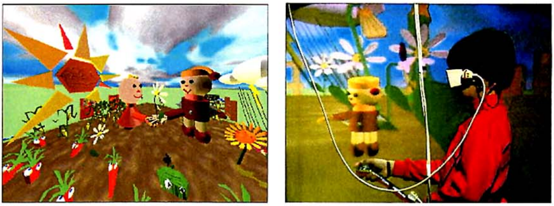

Figure 2.13: Children, and remote children represented by avatars, plant a garden in the *NICE* distributed Collaborative Virtual Environment. [Roussos et al., [80]]

ferent environmental sounds are experienced depending on where each participant is standing, e.g. the children by the shoreline will hear the water, the children in the rainforest will hear the birds, etc.. The user interacts with the Virtual Environment using 'the wand', a simple tracked input device containing a joystick and three buttons. The CAVE's roomsized structure allows for multiple users to move around freely, both physically and virtually surrounded by vivid displays. However, only one person is being tracked in each CAVE, thus reducing the participation of all the children in the same CAVE at one time. One child can set the viewpoint, another child may handle the wand to pick and place objects, while the other children can observe and give their verbal input. It was observed that this has not decreased the feeling of presence and immersion of the participants that are not tracked and the children may exchange roles at any time.

The system is developed in a prototype system named GULLIVR Graphical User Learning Landscapes In VR. Specifically GULLIVR allows multiple participants to share in the exploration of a virtual space, interact with each other, and perform simple tasks. A simple agent architecture based on a frame system, is being developed to support the actions and personalities of the characters. The core of GULLIVR is the CAVE library. On top of this GULLIVR uses SGI Performer and OpenGL to render the Virtual Environment. Although GULLIVR was originally conceived for the CAVE, it is capable of supporting a number of different VR platforms including the Responsive Workbench, and simple graphics workstations. GULLIVR's network component allows multiple networked participants to explore the same virtual space. Multiple distributed GULLIVRs running on separate VR systems are connected via a centralized database server that guarantees consistency across all the separate environments. The communications library supporting GULLIVR is based on a client/server model where the number of remote clients is limited only by bandwidth and latency.

The wand is the physical interface to the virtual world. It is used to navigate around the virtual world, and to manipulate virtual objects within that world. The user can move around within the boundaries of the CAVE, walking around or through virtual objects, and can press the joystick on the wand to move the CAVE through the Virtual Environment. GULLIVR provides the option of flying over the world, or adjusting the floor of the CAVE to coincide with the height of the landscape, thus allowing the user to climb over terrain or ascend and descend stairs, by physically walking in the CAVE. Every element of the scene in GULLIVR is treated as an object. Hence, every object serves as a building block for the construction of other objects. The child can pick up objects in the Virtual Environment by using the wand as an extension of her hand. She moves the wand over to an object and clicks a button on the wand to pick it up. She can then move the object to an appropriate place and let it go.

The NICE application belongs to the most advanced CVEs. However, the system design restricts NICE to be an edutainment application for kids or an artificial basis for game applications and story telling. Artificial avatars are used instead of real looking video representations of remote participants. In this context, view point control and consistence as well as the alignment with distributed data is not implemented as it is of minor importance for the users. In addition, the used techniques allow simple interaction especially designed for kids planting a garden co-operatively.

2.3 Conclusions

This chapter described related work to the thesis. It is shown that there are interaction techniques which are applicable in two-dimensional Graphical User Interfaces (GUIs) as well as in three-dimensional Virtual Environments (VEs). However, a direct usage of two-dimensional interaction techniques in three-dimensional VEs is never possible. Most of the techniques have to be modified to be able to fulfill three-dimensional requirements. Hence, new techniques especially for virtual and collaborative Virtual Environments have to be developed. A good reference frame for doing this represents the body-relative interaction even for rear-projection based Virtual Environments. In combination with other techniques such as speech recognition new and intuitive three-dimensional interaction techniques are designable.
The remaining sections are a survey on selected important CVE systems and applications. They provided an idea about the use of tele-immersion approaches and Collaborative Virtual Environments. From the basic chapter 1 and during the presentation of the different CVE approaches it becomes clear that the absence of a classifying framework for collaborative interaction is the most challenging problem when designing VEs and CVEs. Hence, in the next chapter 3 the goal is to develop an own approach with which designers and programmers are able to analyze user tasks and interaction cycles in VEs. Then, chapter 4 describes an application for a pre-described two user task scenario. Chapter 5 discusses implementation details and in chapter 6 the developed applications are evaluated assessing usability and collaborative awareness.