Dimitar Valkov

Multi-touch Interaction with Stereoscopically Rendered 3D Objects

Münster
- 2013 -

Fach Informatik

Multi-touch Interaction with Stereoscopically Rendered 3D Objects

Inauguraldissertation

zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften
durch den Fachbereich Mathematik und Informatik
der Westfälischen Wilhelms-Universität Münster

vorgelegt von

Dimitar Valkov
aus Sopot, Bulgarien

- 2013 -

Dekan:	Prof. Dr. Martin Stein
Erster Gutachter:	Prof. Dr. Klaus H. Hinrichs
Zweiter Gutachter:	Prof. Dr. Antonio Krüger
Tag der mündlichen Prüfung:	21.01.2014
Tag der Promotion:	21.01.2014

Abstract

While touch technology has proven its usability for 2D interaction and has already become a standard input modality for many devices, the challenges to exploit its applicability with stereoscopically rendered content have barely been studied. In this thesis we exploit different hardware and perception based techniques to allow users to touch stereoscopically displayed objects when the input is constrained to a 2D surface. Approaches to handle this problem can roughly be separated into three groups: (1) approaches which separate the interactive and the visualization surfaces, such that the user can move the interactive surface and manipulate an object "in place"; (2) approaches which exploit the limitations of the human's visual system, i .e., which engage some visual illusions, such that the virtual scene is perceived and understood in 3D while the interaction tasks are carried out on a 2D surface and (3) approaches which shift the problem to the interface design space, i .e., which distinguish the "3D touch" as separate input modality with its own set of interaction techniques. Since the third approach only partially solves the problem we have mainly focused on the first two options and therefore analyze the relation between the 3D positions of stereoscopically displayed objects and the on-surface touch points, where users touch the surface, and we have conducted a series of experiments to investigate the user's ability to discriminate small induced shifts while performing a touch gesture. The results were then used to design a practical interaction technique, the *attracting shift* technique, suitable for numerous application scenarios. In addition, our results indicate that slight object shifts during touch interaction make the virtual scene appear perceptually more stable compared to a static scene, thus applications *have to manipulate* the virtual objects to make them appear more static to the user. Furthermore, we demonstrate how multi-touch hand gestures in combination with foot gestures can be used to perform navigation tasks in interactive 3D environments with the special focus on Geographic Information Systems (GIS), which are well suited as a complex testbed for evaluation of user interfaces based on multi-modal input.

Traditionally, interaction techniques for interactive graphics applications are implemented in a proprietary way on specific target platforms, e. g., requiring specific hardware, physics or rendering libraries, which hinders reusability and portability. Even though abstraction layers for hardware devices are provided by numerous virtual reality libraries, they are usually tightly bound to a particular rendering environment and hardware configuration. In the last part of this thesis we introduce *VINS (Virtual Interactive Namespace)*, a seamless distributed memory space, which provides a hierarchical structure to support reusable design of interactive techniques. We describe the underlying concepts of the framework and present examples how to integrate VINS with other frameworks or already implemented interactive techniques.

Contents

List of Figures

List of Tables

Acknowledgements

First and foremost, I would like to thank my supervisor, Prof. Dr. Klaus Hinrichs, for giving me the opportunity to conduct this exciting research in his group and for his guidance and constant encouragement throughout the years. My special thanks go to my former colleague Prof. Dr. Frank Steinicke for his valuable advice and all the hours spent in discussions, which have helped me at the beginning of this research. I also want to thank my colleagues from the Ubiquitous Media Technologies Lab at the German Research Center for Artificial Intelligence in Saarbrücken (Deutsches Forschungszentrum für künstliche Intelligenz GmbH, DFKI), in particular Prof. Dr. Antonio Krüger and Florian Daiber, for the exciting collaborative project we have been working on. I also want to express my gratitude to Prof. Krüger for agreeing to be referee of this thesis.

Furthermore, I would like to thank all former and present members of the Visualization and Computer Graphics Research Group for the nice working atmosphere and the numerous inspiring on-topic as well as off-topic discussions. In particular, I want to thank Dr. Gerd Bruder for his help in the design of many of the psychological experiments and for the countless discussions in which some of the ideas in this thesis were born. I am grateful to Alexander Giesler for his help co-implementing and conducting some of the experiments. My thanks go also to Evelyn Egelkamp for all the help with administrative problems.

This work was partly supported by grants from the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) through the Project Nr. HI 441/12-1.

Last but certainly not least, I would like to express my deepest gratitude to my wife Petya and my children Valeria and Brian for their unconditional loving support during the time I was working on this thesis.

1 Chapter 1 Introduction

Everything is best for something and worst for something else. The trick is knowing what is what, for what, when, for whom, where, and most importantly, why.

(Bill Buxton)

1.1 Multi-touching Stereoscopic Objects

Two different technologies have dominated recent tech exhibitions and the entertainment market: multi-touch surfaces and 3D stereoscopic displays. These two technologies, which are orthogonal in the sense that multi-touch is about *input* and 3D stereoscopic visualization about *output*, have recently been combined in different setups [ziMb, SSV+09, VSB+10], and first commercial systems that support stereoscopic display and multi-touch interaction are already available [zIMa]. Furthermore, interdisciplinary research projects address the question how users interact with stereoscopic content on a two-dimensional surface [ziMb, zIn]. This combination has the potential to provide more intuitive and natural interaction setups with a wide range of applications, e. g., geo-spatial applications, urban planning, architectural design, collaborative tabletop setups, or 3D desktop environments [zIMa, SSV+09]. However, until now these systems are mainly used for navigation purposes whereas the interaction with the stereoscopically displayed objects is supported only rather rudimentarily.

Multi-touch technology extends the capabilities of traditional touch-based surfaces by tracking multiple finger or palm contacts simultaneously [DL01, ML04, MTSH+10]. The ability to directly touch and manipulate graphical elements without using any input devices has been shown to be very appealing for novice as well as expert users [BWB06]. In particular, the inherent tactile feedback and the ability to directly touch virtual objects increase the acceptance of such techniques and usually allow novices to attain more advanced skill levels swiftly [BF07]. Therefore, the FTIR (frustrated total internal reflection) and DI (diffused illumination) technologies and their inexpensive footprints [Han05, MTSH+10] have led to

the widespread usage of multi-touch on large displays. These setups detect direct touch contact and thus provide tangible feedback without requiring any further user instrumentation. Since humans in their everyday life usually use multiple fingers and both hands for interaction with their real world surroundings, such techniques have the potential to form building blocks for intuitive and natural interfaces. One important observation of previous studies with spatial multi-touch interfaces, e .g., Geographic Information Systems (GIS), is that users initially preferred simple gestures, which are familiar from systems with mouse input using the WIMP desktop metaphor [SHR+08]. After experiencing the potential of multi-touch, users tended towards more advanced physical gestures to solve spatial tasks, but often these were single-hand gestures or gestures, in which the non-dominant hand just sets a frame of reference that determines the navigation mode, while the dominant hand specifies the amount of movement [WIH+08].

Stereoscopic visualization has been known for decades, but recently it has been reconsidered again due to the rise of 3D motion pictures and the upcoming 3D television. Stereoscopy is of particular interest in many application domains, since stereoscopically rendered content provides the user with additional depth cues, which usually decrease the overall cognitive load for understanding complex scenes [KT04]. This is of great importance in application domains such as geovisualization, urban planning or architectural design [SSV+09, SRHM06], since in these domains large amounts of data must be presented on a display of limited size in such a way that it is instantly comprehensible even for untrained users. Recent investigations have revealed further benefits of stereoscopic visualizations, such as the improved stereoscopic image quality and background separation, or enhanced emphasizing techniques [MTH+08, SKK+09]. Nevertheless, stereoscopic visualization introduces new kinds of problems for touch-based interfaces, since the displayed objects are floating freely in the vicinity in front of or behind the display surface, while tactile haptic is only available upon direct contact with the display [SSV+09, VSBH11]. While one could simply use a 3D tracking technique to capture the user's finger motions in front of the display surface, the demand for haptic feedback (i. e., *touching the void* [CKC+10]) has been shown to cause confusion and a significant number of overshooting errors, and some current findings indicate that direct touch may outperform 3D interaction in shallow depth scenarios [BSS13, CKC+10]. Furthermore, passive haptic has the potential to considerably enhance the user experience with touch or grasp based interfaces.

With stereoscopic display, objects might be displayed with different parallaxes resulting in different stereoscopic effects. Until recently, multi-touch interaction research was mainly focused on monoscopically rendered 2D or 3D data sets. In this thesis, we are addressing the challenge to provide multi-touch interaction in stereoscopically rendered environments of different scale. Therefore we have investigated different aspects of the mechanics of the hand-eye coordination, as used for touch interaction, and propose some practical interaction metaphors based on the results of these investigations. Furthermore, we have made some first steps toward enabling stereoscopic touch interaction for multi-modal navigation in virtual environments. Finally, we present the design and the implementation of the VINS interaction framework, which is designed to support development of interaction metaphors.

1.2 Roadmap

This thesis assumes a basic understanding of computer science, in particular, computer graphics, as well as psychology, as related to the field of human computer interaction (HCI). Specialized background information is provided at the appropriate places throughout the thesis.

The remainder of this thesis is structured as follows:

First we formulate the main problems and challenges in Chapter 2. In particular, in the beginning of Chapter 2 we discuss the parallax problems of 3D touch interfaces and the most probable reasons for the oncurrence of these problems. Thereafter, we propose a simple taxonomy of the existing techniques. We have grouped them according to three main interaction design paradigms – move the *surface*, move the *touches* and move the *objects*, and we discuss their benefits and limitations. The core of the chapter, however, is devoted to the question how users touch stereoscopically displayed object, i .e., since stereoscopically rendered objects are floating in the vicinity in front of and behind the display surface, how will a user's touch gesture change to reflect this mismatch. The (to some extent surprising) results of the psychological experiment, which we have conducted to address this question, were the basis for a set of practical design implications. The validity of these implications was confirmed in a preliminary user evaluation.

Chapter 3 is the core contribution of this thesis. In this chapter we generalize our empiric observations of the different states, which a user is passing when interacting with stereoscopic content, and investigate the applicability of the *perceptual illusions* paradigm [Koh10, Ste11] to allow users to interact with stereoscopically displayed objects when the input is constrained to a 2D touch surface. We discuss the possibility to apply unnoticeable manipulations to a virtual scene, such that the user interacts with the desired objects on the touch-enabled 2D surface while consciously perceiving and understanding them in 3D. This paradigm is then substantiated by a series of experiments in which we determined the constraints of these manipulations. These findings were than used in the design and preliminary evaluation of the *object attracting shift* interaction technique described in Chapter 4, which may be usable for a large range of applications.

In Chapter 5 we change the focus from object interaction to multi-touch supported navigation and present a multi-modal metaphor for traveling in large scale virtual environments. The metaphor, which is based on the Nintendo's Balance Board and a multi-touch enabled transparent prop, was tested in a formal user evaluation and found overall great acceptance by the users. Nevertheless, it also revealed some significant problems for the used transparent prop, e .g., severe accommodation convergence problems and difficulty to maintain the stereoscopic half-images merged for a longer period, which could not be alleviated at the time of writing mainly due to missing hardware. Nevertheless, the positive results of the preliminary usability test motivated us to further develop the proposed metaphor.

Due to the diversity of the different interfaces, hardware setups and experiment designs in the scope of this thesis, most of the implementations used a different set of tools and frameworks, which have resulted in a lot of re-writing of existing modules and the impossibility

to share and reuse particular implementations. In order to alleviate this to some extent, we first started developing the generic interaction library called ViARGo! [VBBS11]. Nevertheless, after its initial success with simple scenes and camera manipulation techniques, it has quickly become obvious that the conceptual design of ViARGo! lacks the extensibility and the flexibility needed in the rapidly changing HCI domain. The *VINS (Virtual Interactive Namespace)* interaction framework, described in Chapter 6, is designed to overcome these limitations. In particular, the framework implements a seamless distributed memory space, which provides a hierarchical structure to support reusable design of interactive techniques, with the special focus on 3D interactive environments. Chapter 6 describes the underlying concepts and presents examples on how to integrate VINS with different frameworks or already implemented interactive techniques.

The last Chapter 7 concludes the thesis and gives an overview of possible directions for future work.

1.3 Scientific Publications

The contributions presented in this thesis are based on the following publications:

The evaluation results for the mapping of an *on-surface touch point* to the corresponding object point in virtual scenes with stereoscopic projections and the resulting design implications, i.e., the formulated α-offset, described in Chapter 2 have been presented at the ACM Conference on Human Factors in Computing Systems (**CHI'2011**) [VSBH11]. The human interaction states as well as the evaluation of the perception thresholds for scene shifts while walking toward the surface, described in Chapter 3, were presented at the Joint Virtual Reality Conference (**JVRC'2010**) [VSB+10]. The *scaled shift technique* described in the same chapter was presented at the ACM International Conference on Interactive Tabletops and Surfaces (**ITS'2012**) [VGH12a], and the *generalized scaled shift technique* will be presented at the ACM Conference on Human Factors in Computing Systems (**CHI'2014**) [VGH14]. In the same publication we also present the *object attracting shift technique* [VGH14] described in Chapter 4. In Chapter 5 we propose an intuitive navigation technique, which we have first presented at the IEEE Symposium on 3D User Interfaces (**3DUI'2010**) [VSBH10a], and thereafter (in its extended form) at the Virtual Reality International Conference (**VRIC'2010**) [VSBH10b]. The VINS interaction framework was presented at the ACM Symposium on Virtual Reality Software and Technology (**VRST'2012**) [VGH12b].

Further work in the domain of multi-touch and ubiquitous interaction has been conducted by contributing to the development of an object prediction technique based on the user's hand posture during a grasp gesture, presented at the ACM CHI Workshop on "The 3rd Dimension of CHI: Touching and Designing 3D User Interfaces" (**3DCHI**) [DVS+12]. A multi-touch technique for interaction with objects above a tabletop display, which we call *triangle cursor*, was presented at the ACM International Conference on Interactive Tabletops and Surfaces (**ITS'2011**) [SVH11]. The predecessor of the VINS framework – the (currently semi-retired) ViARGo! library was presented at the Workshop on Software Engineering and Architectures

for Realtime Interactive Systems (**SEARIS'2011**) [VBBS11]. The author of this thesis also contributed to the design of the *"immersive vitual studio"* [BSVH10b, BSVH10a] presented as poster at IEEE Symposium on 3D User Interfaces (**3DUI'2010**) and as full-paper at the Virtual Reality International Conference (**VRIC'2010**), as well as to the initial formulation of the 3D touch design space [SSV+09] as presented at IFIP TC13 Conference in Human-Computer Interaction (**INTERACT'2009**). We have also presented the *Haptic Prop* tangible device [VMH13] as a poster at the ACM Symposium on User Interface Software and Technology (**UIST'2013**).

In addition to the scientific research carried out, the software frameworks ViARGo! and VINS were developed and extensive contributions to the initial design and implementation of the new visualization framework Nixie were done.

2 Chapter 2

Understanding 3D Touch

Without a theory the facts are silent.

(Friederich A. von Hayek)

2.1 Understanding Touch

Notwithstanding the initial excitement around multi-touch interfaces it has quickly become apparent that using touch as primary input modality poses (even in 2D contexts) some fundamental limitations for traditional interface design [BW10, MTSH+10]. Some of the most important problems are the missing hover, occlusion and precision problems and – depending on the implementation – missing or non-adequate visual feedback.

In particular, the size of the human fingers and the lack of sensing precision make precise touch screen interactions difficult [BWB06, HB09]. The approaches to handle this can be roughly separated into two groups. Approaches from the first group try to shift the problem into the interface design space. Therefore, *precise selection* is distinguished as new interface requirement, which demands additional functionality and thus an extended set of interaction metaphors or techniques. Some examples of such techniques are the adjustable [BWB06] or fixed cursor offset [PWS88], or the scaling of the cursor motion [BWB06].

Characteristic for the second group of solutions is that they try to overcome or reduce the problem by modeling the user perception and action during the touch. Thus, these approaches try to identify a set of traceable features, which may help to better recognize the intended touch position. Examples of such features are the orientation of the user's finger [HB09] or visual features on the upper finger surface [HB11]. The primary benefit of these approaches over the pure "brute-force" interface solutions is that they help to understand the mechanics of a touch gesture, when used for input, and provide indications which help to identify the sources of the inaccuracy in traditional touch devices. For instance, Holz and Baudisch [HB11] have formulated the "projected center model" of touch interaction, which attributes the inaccuracy of the traditional input devices to – what they called – "parallax

artifacts" between the finger motion control based on the visual features extracted at the top of the finger and device sensing based on the bottom side of the finger. Recent work has also identified the hand pre-shaping as valuable source of information in this regard [DVS+12]. Indeed, as the investigations of many neuro-psychological and robotic research groups have shown, there is a strong correlation between the course of hand shaping and the object, which is subject to interaction [DVS+12, SFS02, San00].

Extending the interaction environment to the third dimension usually intensifies the impact of these issues on the user experience and satisfaction [SSV+09] and introduces new problems which are negligible in monoscopic contexts. In this chapter we examine these problems in more detail and consider several high level approaches to address them. Furthermore, we investigate the effect of parallax on the touch precision and discuss some of the design implications resulting from our evaluations.

2.2 Problems when Touching Parallaxes

Recently many approaches for extending multi-touch interaction techniques to 3D applications with *monoscopic* rendering have been proposed [HCC07, MCG10, RDH09, WIH+08]. For instance, Hilliges et al. [HIW+09] have tested two depth sensing approaches to enrich the multi-touch interaction space beyond the touch surface in a tabletop setup with monoscopic projection. Hancock et al. [HCC07] have introduced the concept of *shallow-depth 3D*, i. e., 3D with limited depth, in order to extend the interaction with digital 2D surfaces and have developed one, two and three fingers interaction techniques for object selection and manipulation in this context. Martinet et al. [MCG10] have designed a multi-view direct and a single-view indirect technique for 3D object positioning, and Reisman et al. [RDH09] propose an energy-minimization technique for adapting 2D interaction to 3D transformation. The benefits of using physics engines for multi-touch input specification are discussed by Wilson et al. [WIH+08], and the interaction with objects with negative parallax on a multi-touch tabletop setup is further addressed by Benko et al. [BF07], who have proposed the *balloon selection* metaphor to support precise object selection and manipulation in augmented reality setups.

In 2007 Grossman and Wigdor [GW07] provided an extensive review of the existing work on interactive surfaces and developed a taxonomy to classify the current work and to point out new directions. This framework takes into account the perceived and the actual display space, the input space and the physical properties of an interactive surface. As shown in their work, 3D volumetric visualizations are rarely being considered in combination with 2D direct surface input. More recent surveys, e .g., Argelaguet and Andujar [AA13], still identify 3D direct touch interaction as promising research direction, which is still not sufficiently investigated.

Nevertheless, direct touch interaction with *stereoscopically* rendered scenes introduces new challenges, as described by Schöning et al. [SSV+09]. In their work an anaglyph- or passive polarization-based stereo visualization was combined with FTIR-based touch detection on a

multi-touch enabled wall, and approaches based on mobile devices for addressing the formulated parallax problems were discussed. A similar option for direct touch interaction with stereoscopically rendered 3D objects is to separate the interactive surface from the projection screen, as proposed by Schmalstieg et al. [SES99]. In their approach, the user is provided with a physical *transparent prop*, which can be moved on top of the object of interest. This object can then be manipulated via single- or multi-touch gestures, since it has almost zero parallax with respect to the prop. Nevertheless, this requires instrumentation again, which may defeat some of the benefits of touch interaction.

Stereoscopic perception requires each eye to see a slightly different perspective of the same scene, which results in two distinct projections on the display. Depending on the disparity between the two projections, virtual objects can be presented with *positive*, *negative* or *zero* parallax, resulting in different visual impressions.

If objects are rendered with *zero* parallax they appear aligned with the plane of the display surface and are therefore perfectly suited for touch-based interaction [SSV+09]. Unlike the positive and negative parallax half-spaces, the zero parallax plane poses considerable constraints on the placement, dimensions and form of the objects, and therefore contradicts the benefits of using stereoscopy, or 3D in general. In this context the question arises how sensitive humans are with respect to misalignment between visually perceived and tactually felt contact with virtual objects. For example, if an object is rendered at some small distance in front of the display surface, is the user going to move her finger through the object until she receives tactile feedback due to the contact with the display, and how small may this distance be? In particular, this may allow touch (and possibly multi-touch) interaction within stereoscopic environments without loosing the advantages of common 2D techniques. While it is reasonable to assume that users tolerate a certain amount of misalignment between the perceived visual depth and the exact point at which haptic feedback is received [VGH12a], similar effects may lead to misalignment between perceived and actual object depth depending on object size, form, texture, etc. This may then infer the perceived alignment between two objects or between an object and the plane of the display surface. Nevertheless, if 2D interaction is intended or the displayed virtual objects have no associated depth information (e. g., UI widgets), the zero parallax plane may provide superior user experience compared with alternative depth distributions.

Objects displayed with *positive parallax* are perceived to be behind the screen surface. These objects cannot be accessed directly, since the user's reach is limited by the display. Since the display surface has usually no visual representation in a stereoscopically rendered scene, trying to reach an object with strong positive parallax may become unnatural and in some cases even harmful. Nevertheless, if the object is close to the surface – rendered with shallow depth – the only effect is that the user receives haptic feedback shortly before its visual representation is reached, i .e., the points of receiving haptic and visual feedbacks are spatially misaligned. In the following Chapter 3 we investigate the problem in more detail and make the first steps toward determining within what range this misalignment is still unnoticeable for the user. For objects rendered with *strong positive* parallax, indirect techniques might be

(a) (b)

Figure 2.1: Illustration of the accommodation-convergence problem; The user is either focused on the finger (a), which makes the selection ambiguous, or on the object (b), which disturbs the visual perception of the finger.

more adequate. For instance, one could cast a virtual ray from the camera's origin through the on-surface touch point and determine the first object hit by that ray [Ste06] or use some abstract interface widget [DFK12] to virtually move the user's touches in the 3D space below the surface. Even though such techniques are indirect, it is often claimed that users experience them to be "natural" and "obvious" [BKLP04, DFK12].

Objects that appear in front of the projection screen, i. e., objects with *negative parallax*, introduce the major challenge in this context. When the user wants to interact with such an object by touching, she is limited to touch the area behind the object, since most touch sensitive screens capture only direct contacts, or hover gestures close to the screen. Therefore the user has to penetrate the visual objects to reach the touch surface with her finger. In addition to the fact that users commonly consider this as unnatural, the stereoscopic perception may be disturbed, since the user's visual system is fed with contradicting information. If the user penetrates an object while focusing on her finger, the stereoscopic effect for the object would be disturbed, since the user's eyes are not accommodated and converged on the projection screen's surface. Thus the left and right stereoscopic images of the object's projection would appear blurred and could not be merged anymore (Figure 2.1 (a)). However, focusing on the virtual object would lead to a disturbance of the stereoscopic perception of the user's finger, since her eyes are converged to the object's 3D position (Figure 2.1 (b)). In both cases the stereoscopic impression may be lost due to these artifacts.

Another significant problem in this case is the discrepancy between the disparity and occlusion cues. Indeed, as illustrated in Figure 2.2 (b) if the users finger penetrates the object in the last phase of the touch gesture, binocular disparity cues are suggesting that her finger is already behind the object. Nevertheless, the stereoscopic projection on the display surface cannot occlude the finger (or any object for that matter) in front of it. Thus, the finger is occluding parts of the object, and occlusion cues are confirming that the object is in front of the screen (s. Figure 2.2 (a)). Since occlusion cues usually dominate over disparity, disparity cues may be ignored and the images for the left and the right eye may not be merged any

(a) (b)

Figure 2.2: Illustration of the occlusion problem; while the occlusion cues (a) indicate that the user's finger is in front of an object, binocular disparity cues (b) are suggesting that the user's finger is behind the object.

more, which results in loss of the stereoscopic impression. In both cases touching an object may become ambiguous. However, as discussed in detail in Chapter 3, users have difficulties to precisely estimate the depth of an object, which is displayed close enough to the surface, when they try to touch it.

2.3 Design Paradigms for 3D Touch

While one could simply use a 3D tracking technique to capture the user's finger or hand motions in front of a display surface, it has been shown that touching an intangible surface (i. e., *touching the void*) leads to confusion and a significant number of overshooting errors [CKC+10], and passive haptic has the potential to considerably enhance the user experience with touch or grasp based interfaces. Furthermore, touch interaction is nowadays becoming a standard for most mobile devices or tabletop setups, thus a change to other technology is usually not desirable, and sometimes not possible.

Existing approaches to deal with the problems of touch interaction in stereoscopic contexts could be roughly separated into three distinct paradigms:

- **Move the interactive *surface*** (cf. Figure 2.3 (a))
- **Move the *touches*** (cf. Figure 2.3 (b))
- **Use perceptual illusions** (cf. Figure 2.3 (c))

The main point in the *move the surface* concept is that one can decouple the interactive surface from the display and move it freely in the 3D volume above the display. One possibility to achieve this is to use a multi-touch enabled transparent prop [SES99, VSBH10a], which can be aligned with a floating object and used as input to interact with this object in place. Thus, the user interacts "directly" with the object through the prop and receives haptic feedback. Nevertheless, since the objects aligned with the prop are projected with very large disparity,

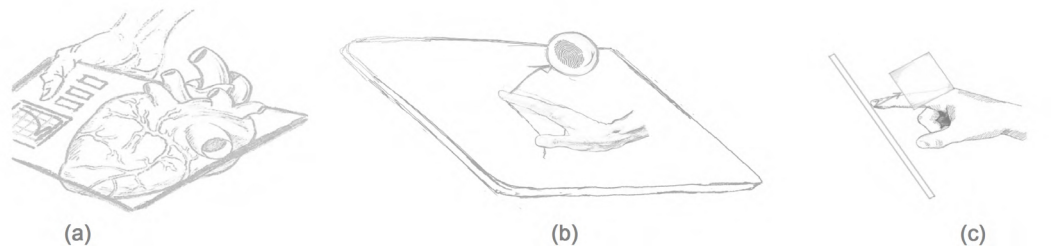

Figure 2.3: Illustration of the three design paradigms for touch interaction with stereoscopic content: (a) "move the surface" paradigm; (b) "move the touch" paradigm and (c) "perceptual illusions" paradigm

the users often have considerable problems to maintain the fusion of the images for the left and the right eyes. This is further impaired by even very small scratches on the surface of the prop, which may distract the eye accommodation on the top of the prop instead of on the display surface. These and similar problems are discussed in more detail in Chapter 5, where a combination of a multi-touch enabled transparent prop and an intuitive navigation technique is presented. Another recently published alternative is to use opaque props and a top projection exactly on the surface of these props, i. e., to use tangible views [STSD10]. Nevertheless, up to our best knowledge the "tangible views" have not been considered with stereoscopic projections.

With the second paradigm the touches are moved into the 3D space above or below the display surface by using the on-surface 2D positions of multiple touch points to calculate a 3D position of a distant "cursor" [BF07, SVH11]. As with the touch precision, the approach shifts the problem into the interface design space by defining the *stereo touch* as distinct input modality. Examples of interface techniques based on this approach are the *balloon selection* metaphor [BF07], the *triangle cursor* [SVH11], the *fishnet* metaphor [DFK12] and many more [HBCdlR11, CDH11, SGH$^+$12]. The main drawback of these techniques is that 2D interaction on the surface of the display is either not supported or realized with a different set of techniques, which leads to frequent switching between different interaction modes.

Use of perceptual illusions to manipulate the properties of the rendered scene or parts of it in such a way that the user's finger is redirected onto the display surface while reaching to touch a floating object, is the core idea of the last paradigm. The essential part of this approach is that such manipulations have to be imperceptible for the user, i. e., the visual effects of their application must remain below her perceptual detection threshold. Indeed, as shown by Dvorkin et al. [DKK07], there is only a (small) finite number of parametric functions for ballistic arm motions which are selected and parametrized according to the arm and object positions prior to the execution. Thus, if the user detects a change in the scene she would abort the entire gesture and "reprogram" a new gesture rather than adjust the current one. This usually takes more than $200ms$ [DKK07] and may thus significantly impair performance. Perceptual illusions for 3D touch interaction are discussed in detail in Chapters 3 and 4. While the next chapters describe particular incarnations of the presented design paradigms we first concentrate on the effect of parallax shifts on the touch precision.

2.4 Touching Parallaxes

In the monoscopic case the mapping between an *on-surface touch point* and the *intended* object point in the virtual scene is straightforward, but with stereoscopic projection this mapping introduces problems. In particular, since there are different projections for each eye, the question arises where users touch the surface when they try to "touch" a stereoscopic object. In principle, the user may touch anywhere on the surface to select a stereoscopically displayed object. However, according to observations we have made, it appears most reasonable that users try to select a stereoscopically rendered object by touching:

- **the midpoint between the projections for both eyes (so called *middle* eye projection)**
- **the projection for the *dominant* eye**
- **the projection for the *non-dominant* eye**
- **the orthogonal projection of the object onto the touch surface (i. e., the object's *shadow*)**

A precise mapping approach is important to ensure correct selections, in particular in a densely populated virtual scene. In order to allow the user to select arbitrary objects, a certain area of the touch surface, which we refer to as *on-surface target*, must be assigned to each object. Therefore, it is important to know where the user will touch the surface for a given object. In the following we present the results of an experiment that we have performed in order to determine the on-surface targets for objects stereoscopically rendered at different 3D positions. We found that users tend to touch between the projections for the two eyes with an offset towards the projection for the dominant eye. Our results give implications for the development of future touch-enabled interfaces, which support 3D stereoscopic visualization.

2.4.1 Experiment

In this section we describe the experiment in which we have analyzed where users would touch the surface of the projection wall for objects at different 3D positions in space rendered stereoscopically with positive, negative and zero parallax. We have also examined if the stereoscopic parallax impacts users' performance time or the kinematics of the touch gestures.

Experimental Setup

For the experiment we used a multi-touch enabled passive stereoscopic back projection system. The prototype is illustrated in Figure 2.4. The multi-touch technology of this surface is based on the *Rear-DI* [MTSH+10] principle. Using this approach, infrared (IR) light illuminates the screen from behind the touch surface. When an object, such as a finger or palm, comes in contact with the surface it reflects the IR light, which is then sensed by a camera. In our case this method can also detect hover, i .e., blurred indication of objects that are close to the

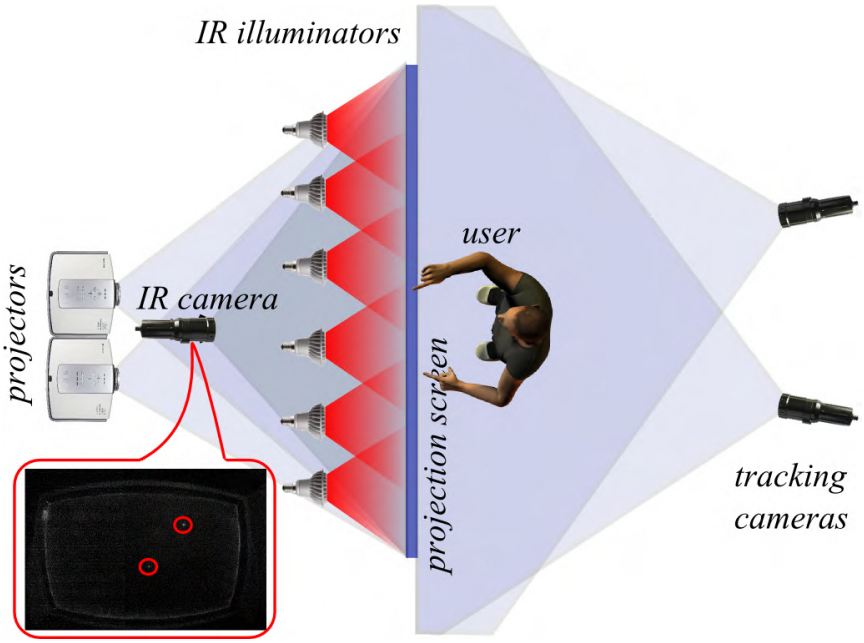

Figure 2.4: Illustration of the multi touch enabled stereoscopic projection wall used in the experiment. Six high power IR-LED lamps are used for back-lighting the projection surface. The reflected IR light is captured by a Point Gray's Dragonfly2 camera as a 8-bit monochrome video stream with a resolution of 1024×768 pixels. The inset illustrates a frame from this stream. Stereoscopic projection is provided by two DLP projectors and the user's head position may be tracked with an IR tracking system.

interaction surface, but are still not in contact with it. We use a $200cm \times 161cm$ projection screen as touch surface, and a total of six infrared (IR) illuminators (i. e., high power IR-LED lamps) for back-lighting this surface. Since our projection screen is made from a mat, diffusing material, we do not need an additional diffusing layer for it. A digital video camera (PointGrey Dragonfly2) equipped with a wide-angle lens and a matching IR band-pass filter is mounted at a distance of $3m$ from the screen and captures an 8-bit monochrome video stream with a resolution of 1024×768 pixels ($2.81mm^2$ precision on the surface) at 30 frames per second (fps). For visualization we use passive stereoscopic back projection with circular polarization. Two DLP projectors with a resolution of 1280×1024 pixels ($1.56mm$ effective pixel-width, brightness 2800 $ANSIlumen$) provide stereo images for the left and right eye of the user. In order to perceive a stereoscopic image, subjects wear circular polarization glasses. For detection of the touch input we use a modified version of NUI Group's CCV software. The software needed for the experiment runs on a computer with Intel Core i7 @ $2.66GHz$ processor, $6GB$ RAM and nVidia GTX295 graphics card. As illustrated in Figure 2.4, an optical tracking system tracks the position of the user's head in order to provide view-dependent rendering. However, during the experiments subjects were not moving in front of the projection wall, and therefore head tracking was not used. All participants were recorded with a video camera (640×480 @ $30fps$) during the experiment.

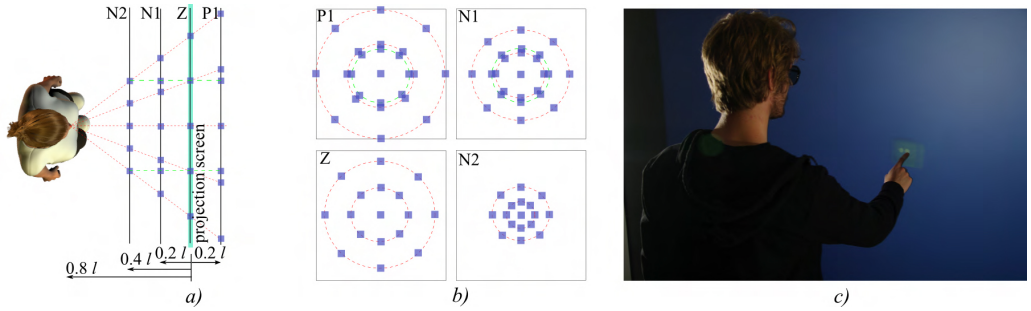

Figure 2.5: Experiment design; (a) top view of the object arrangement; (b) object arrangement for each parallax plane; (c) photo of a subject while participating in the experiment.

Materials and Methods

We have used the *Porta test* and the *Dolman test* to determine a subject's *sighting-dominant eye* [MOB03]. Subjects exhibiting differing eye dominance in the two tests were excluded from the experiment. Next, subjects judged in a *two-alternative forced choice* [Fer08] task the parallax of four small shapes displayed stereoscopically on the projection wall (two with positive and two with negative parallax) in order to verify their ability of stereoscopic vision.

Subjects were positioned in front of the projection screen in such a way that they could conveniently perform all touch gestures during the experiment with their dominant arm. In a pilot experiment we determined an optimal distance to the projection screen of about 0.8 the subjects arm-length (l). This distance provided an operational radius of $r = 0.6 \cdot l$ around the projection of the subject's head position on the wall (see Fig 2.5 (a)). We marked the corresponding position for each subject on the floor. Subjects were told to remain in this position during all trials of the experiment. If both stereopsis and eye-dominance tests were accomplished successfully, a written task description of the experiment was presented via slides on the projection wall.

For the experiment we have used the method of constant stimuli. In this method the object positions are not related from one trial to the next, but presented randomly and uniformly distributed. For visual stimuli we have used small spheres with a size of $1.5cm$, which ensured a clearly visible target with a reasonable stereoscopic impression; the center of the sphere indicated the exact position subjects should touch. For each trial, the sphere was surrounded by a semi-transparent box to provide additional depth cues (such as perspective distortion, texturing, etc.) to the user. As illustrated in Figure 2.5 (c), we adjusted the color of the box and sphere as well as the background in such a way that stereoscopic crosstalk between the stereoscopic images for the left and right eye was minimized.

In each trial, the subject's task was to touch the center of the sphere, hold her finger at the same position until the object disappeared ($200ms$ after the touch was detected) and then release her finger from the touch wall. $200ms$ after subjects moved their fingers away from the touch surface a new object was displayed, which indicated the beginning of the next trial.

As illustrated in Figures 2.5 (a) and (b), the objects used in the trials were arranged in concentric circles on four different planes parallel to the projection plane at $z = 0$ in a left-

handed coordinate system. Since users are more sensitive to discrepancies between visual and tactile feedback if objects are displayed with positive parallax not accessible for direct touch interaction (cf. [VSB+10]), we have focused in this research primarily on objects exhibiting negative parallax. Therefore we have tested two parallax planes, called N1 and N2, with negative parallax at distances $z = 0.2 \cdot l$ and $z = 0.4 \cdot l$, respectively. In addition, we have tested one plane P1 with positive parallax at $z = -0.2 \cdot l$, and the plane Z aligned with the projection plane ($z = 0$), i. e., with zero parallax. The plane P1 was chosen to be relatively close behind the projection surface. If the plane had been chosen to be further behind, it would have been more likely for the subjects to accidentally hit the projection screen while still in the ballistic phase of the motion. The arrangement in concentric circles was used to provide symmetrical view conditions for the objects on the same plane, i. e., with same stereoscopic parallax. As mentioned above, for the $z = 0$ plane we have chosen the radius of the outer circle to match the maximal (convenient) reach distance r of the user, i. e., $0.6 \cdot l$. The inner circle had half the radius ($0.3 \cdot l$). On the planes N2, N1 and P1 the radii of the circles were selected in such a way that corresponding objects across all planes were positioned on a line of sight extending from the user (see Figure 2.5 (a)), thus the user's hand movement distances were the same across all conditions. In addition to these locations, we have added on each plane circles with a constant radius of $0.3 \cdot l$ in order to test also different stimuli that depend only on the stereoscopic parallax, i. e., differ only in their z values.

Participants

13 male and 9 female subjects (age 22-29, $\varnothing$: 25.4; height $163cm - 196cm$, $\varnothing$: $179.9cm$) participated in the experiment. All subjects were right-handed. We determined for 15 subjects that their right eye was dominant (8 male and 7 female), and for 7 subjects (5 male and 2 female) that their left eye was dominant. All had normal or corrected to normal vision. 11 of the subjects wore glasses or contact lenses and none of them reported amblyopia or known stereopsis disruptions. 11 subjects had experience with stereoscopic projections, and 9 had already participated in a study in which stereoscopic projections were used. All subjects were naïve to the experimental conditions. The total time per subject including pre-questionnaire, instructions, training, experiment, breaks, and debriefing took 35 minutes. Subjects could take a break at any time. In addition, after each 45 trials subjects had to take breaks of two minutes in order to minimize errors due to exhaustion or poor concentration. Subjects were students or members of the departments of computer science, mathematics and psychology at the University of Münster. Some subjects received class credits for their participation.

2.4.2 Results

For one subject we observed differing eye dominance in the *Porta test* and the *Dolman test*, and therefore excluded this subject from the experiment. The error rate was 0% across all subjects, i. e., no subject reported that she accidently touched a wrong position. Therefore we considered the data from all trials.

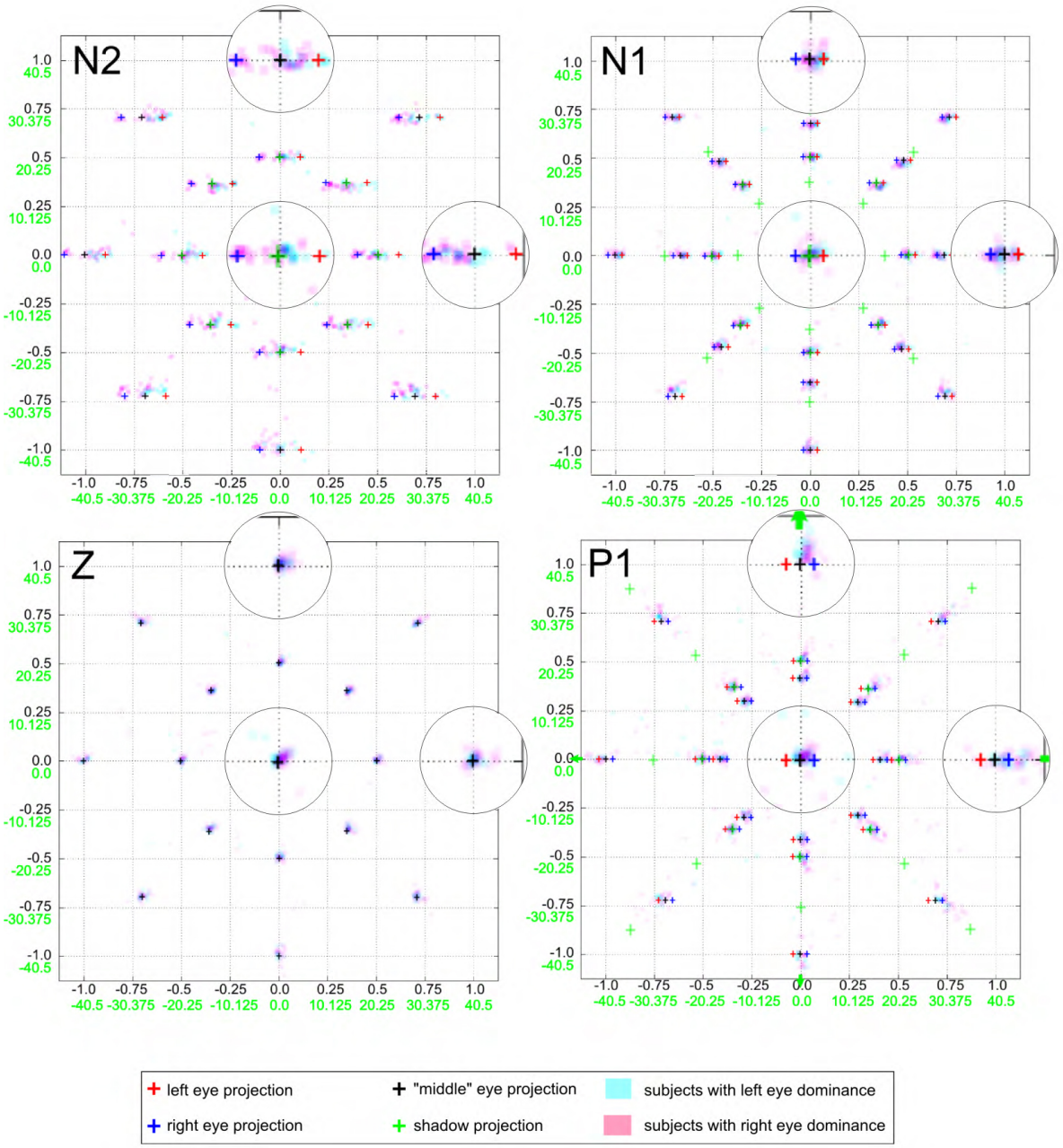

Figure 2.6: Individual touch results for all trials from the experiment: (top left) shows the touch locations for the subjects in condition N2, (top right) for condition N1, (bottom left) for condition Z and (bottom right) for condition P1. The crosshairs illustrate the position of the different touch targets: (blue) corresponds to dominant eye, (red) corresponds to non-dominant eye, (black) corresponds to the center, and (green) to the shadow projection of the stereoscopic object. The green numbers show the corresponding values in centimeters calculated for subjects with $180cm$ body height. The insets show zoomed by factor $\times 2$ views for some of the clusters.

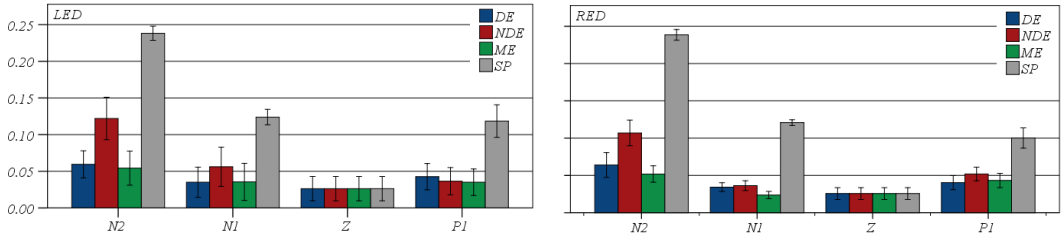

Figure 2.7: Mean distances from the target points for different parallax surfaces for subjects from (a) Group LED and (b) Group RED. The vertical bars show the standard error.

Touch Points

Since the objects in our experiment were arranged in concentric circles centered at the subject's eye level, we define the focus point on the projection wall as the origin of a 2D coordinate system, with the y-axis running from bottom to top, and the x-axis running from left to right. As units for both axes we have chosen the subject's maximal (convenient) reach distance $r = 0.6 \cdot l$. We express the coordinates of all touches performed by the subject in terms of this coordinate system. Since the coordinate systems used for the different subjects take into account the differing arm lengths and body heights of the subjects, the coordinates of the touch points are already normalized and can be compared directly among all subjects.

The individual touch locations for all trials are plotted in Figure 2.6. The crosshairs illustrate the positions of the different touch targets, i. e., blue corresponds to the projection for the right eye, red to the projection for the left eye, black to the midpoint between both projections (middle eye), and green to the orthogonal "shadow" projection of the stereoscopic object.

We have not found a significant difference in the data for male and female participants (two-sided t-test, $p = 0.932$ for the x-coordinate and $p = 0.637$ for the y-coordinate), so we have pooled the results for all subjects. We have calculated for each tested object the corresponding unified coordinates for the four considered touch targets, i. e., dominant eye (DE), non-dominant eye (NDE), middle eye (ME) and shadow projection (SP), and determined the distances between the performed touches and the corresponding target points using a 2D Euclidean metric. With a two-sided t-test we found a significant difference between the mean distances for subjects with left eye-dominance and for subjects with right-eye dominance ($T_{20} = 2.174, p = 0.042 < 0.05$). Mean distance for the left-dominant subjects was $0.094 \cdot r$ ($SD = 0.0284$), and for the right dominant subjects it was $0.075 \cdot r$ ($SD = 0.0135$). Thus, we split the results for the two groups, i. e., left eye dominance (LED) group and right eye dominance (RED) group, in the subsequent analysis.

We have determined an offset of $(0.00883 \cdot r, -0.00387 \cdot r)$ from the coordinate system's origin which was not significantly different from $(0, 0)$, (t-test, $p = 0.16$ for unified x and $p = 0.503$ for unified y). This indicated that there is no significant difference between the objects on the left, right, top or the bottom side.

We have calculated the mean distances to each target point for the four different parallax planes for each subject of the LED group. Figure 2.7 (a) shows a bar plot of the distances from the target points for different parallax surfaces for the group LED, the means and standard deviations are shown in Table 2.1. Those mean values were then analyzed with a factorial analysis of variance (ANOVA), testing the within-subjects effects of target point and stereoscopic parallax. The analysis revealed a significant main effect for the parallax ($F^3_{96} = 59.61$, $p < 0.01$) as well as for the target point ($F^3_{96} = 69.69$, $p < 0.01$). Post-hoc analysis with the Tukey test showed that subjects touched significantly closer to an object that is displayed on the surface with zero parallax compared to objects displayed with positive or negative parallax ($p < 0.01$ for P1, N1 and N2). Furthermore, there was a significant difference between the touch targets for objects displayed with strong negative parallax N2 and objects displayed with other parallaxes ($p < 0.01$ for P1, Z and N1). We have not found a significant difference between planes P1 and N1 ($p = 0.919$). The post-hoc analysis also showed that the touch points were significantly farther away from the SP target than from all other targets ($p < 0.00$ for DE, NDE and ME). Furthermore, subjects from group LED touched significantly further away from the NDE target in comparison to the targets DE ($p = 0.034 < 0.05$) or ME ($p = 0.01 < 0.05$), but significantly closer than to the target SP ($p < 0.01$). For the LED group, we have not found a significant difference between the two targets, which were closest to the subjects' touch points, i. e., DE and ME ($p = 0.973$).

Figure 2.7 (b) shows a bar plot of the distances from the target points for different parallax surfaces for the group RED, the means and standard deviations are shown in Table 2.2. The mean distances were then analyzed with a factorial analysis of variance (ANOVA), testing the within-subjects effects of target point and stereoscopic parallax.

Again, we calculated the mean distances to each target point for the four different parallax planes for each subject of the RED group and performed a factorial ANOVA to test the

	DE		NDE		ME		SP	
	mean	SD	mean	SD	mean	SD	mean	SD
P1	0.0427	0.0656	0.0366	0.0598	0.0351	0.0636	0.1185	0.0811
Z	0.0265	0.0630	0.0265	0.0630	0.0265	0.0630	0.0265	0.0630
N1	0.0351	0.0661	0.0563	0.0727	0.0358	0.0713	0.1240	0.0482
N2	0.0594	0.0510	0.1223	0.0678	0.0545	0.0618	0.2383	0.1042

Table 2.1: Mean distances (and standard deviation) from the target points for different parallax surfaces for subjects from group LED.

	DE		NDE		ME		SP	
	mean	SD	mean	SD	mean	SD	mean	SD
P1	0.0411	0.0534	0.0471	0.0527	0.0407	0.0536	0.1060	0.0704
Z	0.0259	0.0559	0.0259	0.0559	0.0259	0.0559	0.0259	0.0559
N1	0.0346	0.0498	0.0426	0.0513	0.0275	0.0513	0.1219	0.0466
N2	0.0626	0.0536	0.1118	0.0554	0.0527	0.0485	0.2383	0.1023

Table 2.2: Mean distances (and standard deviation) from the target points for different parallax surfaces for subjects from group RED.

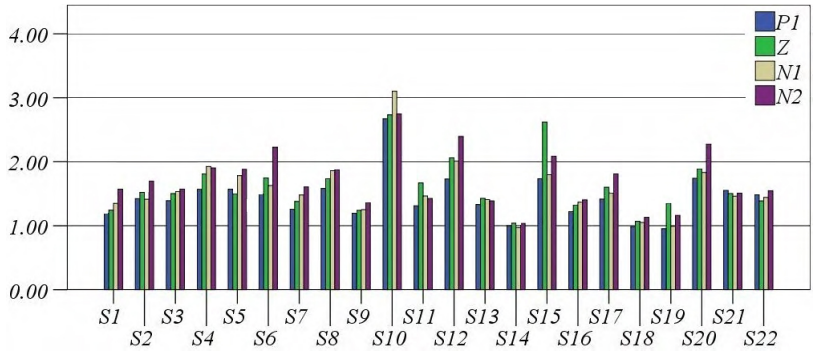

Figure 2.8: Performance times per subject and parallax.

within-subjects effects of target point and stereoscopic parallax. The analysis revealed a significant main effect for the parallax ($F^3_{224} = 230.68$, $p < 0.01$) as well as for the target point ($F^3_{224} = 254.19$, $p < 0.01$). Post-hoc analysis with the Tukey test showed that it was also significantly easier for subjects from group RED to touch an object with zero parallax compared to all other parallaxes ($p < 0.01$ for P1, N1 and N2), and there was a significant difference between strong negative parallax N2 and all other parallaxes ($p < 0.01$ for P1, Z and N1). As for the LED group, we have not found a significant difference between planes P1 and N1 ($p = 0.463$). Similar to subjects from group LED, subjects from group RED touched significantly farther away from the SP touch target in comparison to all other targets ($p < 0.01$ for DE, NDE and ME). Furthermore, subjects touched significantly farther away from the NDE target in comparison to the targets DE ($p < 0.01$) or ME ($p < 0.01$), but significantly closer than to the target SP ($p < 0.01$). As for the group LED, we found no significant difference between the ME and the DE targets ($p = 0.491$).

Performance Time

Figure 2.8 shows the mean time elapsed until a subject touched a corresponding object, for each subject and parallax planes P1, Z, N1, N2. We have analyzed the results with a one-way ANOVA, testing the within subject effect of stereoscopic parallax on the mean performance time. We have not found a significant main effect ($F^3_{18} = 1.489,\ p = 0.223$), i. e., the subjects' performance time is almost the same for objects on planes P1, Z, N1 and N2. The estimated mean value for the performance time on the parallax plane P1 is $1.446s$ ($SD = 0.3599$), for the Z plane is $1.608s$ ($SD = 0.4287$), for plane N1 $1.575s$ ($SD = 0.4467$) and for plane N2 $1.710s$ ($SD = 0.4359$).

Again, we have not found a significant difference between male and female subjects (two-sided t-test, $p = 0.07$). The mean time for the female subjects was $1.73s$ ($SD = 0.352$) and for the male subjects $1.91s$ ($SD = 0.532$). Nevertheless, since the objects on N1 and N2 were considerably closer to the subjects compared to the objects on Z, these results might show a degradation of the user's performance time for objects with negative parallax.

2.4.3 Discussion

In general, the results for the LED and RED group show the same qualitative behavior and differ only in quantity. Right-handed subjects with right eye dominance perform significantly more precise touch gestures than right-handed subjects with left eye dominance. However, subjects from both groups tend to choose the same strategy to select a stereoscopic object on a two-dimensional touch surface.

As it can be seen in Figure 2.6, the touch points for planes N2 (top left), N1 (top right) and P1 (bottom right) are more scattered than the touch points on the Z plane (bottom left), although the size of the projected images for objects on P1 is smaller than the size of the projections for objects on the Z plane. Furthermore, the touch points on planes N1 and P1 are comparably scattered, although the projected images for objects on N1 are greater then those on P1. This indicates that touching objects displayed with positive or negative stereoscopic parallax on a 2D surface induces more imprecision than touching objects with zero parallax. The touches on the N2 plane are more scattered compared to those on all other parallax planes and, the calculated distances to the target points are significantly larger than those for the planes N1, Z and P1. Thus, imprecision increases with stereoscopic parallax, in particular for objects displayed with negative parallax.

As described in the previous section, we have not found a significant difference between the per-subject performance times for different parallaxes. Nevertheless, Figure 2.8 shows that most of the users performed more slowly for objects on the N2 plane than for other objects. The inverse tendency can be seen for objects displayed stereoscopically with positive parallax, i. e., the objects on P1. An analysis of the video records that we made during the experiment revealed that for the objects on N2 and N1 most users perform a ”usual” point gesture until they reach the visual representation of the object and then move the finger slowly through it until it reaches the interactive surface, which may be an explanation for the increased performance times. In contrast, some of the users reported that they were ”surprised by the surface” while performing some of the touch gestures in order to select objects behind the surface. This also may have had an impact on the decreased performance times and precision, since in these cases, the gesture ended prematurely, without users fully executing the slower and more precise *correction phase*. Furthermore, since the motion of a user's arm during a touch gesture may differ very much among users and for different object positions, the prematurely ended gestures may have led to the ”random touches”, i. e., outliers, on P1, as may be seen in Figure 2.6.

None of the subjects complained about touch difficulties, for example, accidentally recognized touches, during the experiments. Most of the subjects have observed the parallax problem described in the introduction (see Figure 2.1) and reported that for objects displayed with negative stereoscopic parallax it was difficult to get a stereoscopic impression when touching the surface behind the object with the finger. None of the subjects evaluated this effect as a strong distraction from the interaction, and some of the subjects, in particular those with lower performance times, have not noticed it at all. Interestingly, some of the

subjects reported difficulties to merge the objects on the N2 plane, although they were within the wide accepted maximal distance for positive parallax. This may be due to the fact that the participants were relatively close to the projection wall and thus were more sensitive to small mismatches due to resolution or illumination constraints.

2.5 Design Implications

Even though our analyses show that the ME and DE targets are best guesses for the location of the on-surface touch targets for stereoscopic objects, the calculated mean distances to the actual touch points are still rather large. For instance, the mean distance between the DE targets and the corresponding touch points is 0.0656 (for group LED) and 0.0493 (for group RED), which corresponds to $2.65cm$ (group LED) respectively $1.99cm$ (group RED) for a user with a height of $180cm$. Furthermore, the video recordings of the subjects during the experiment reveal that during most of the trials they neither touched the DE nor the ME target, but rather a point "in-between" both touch targets.

We can express the position P_{IMD} of this new *intermediate* target point (IMD) as a linear blend between the positionas P_{DE} and P_{ME} of the DE and ME targets respectively:

$$P_{\text{IMD}} = P_{\text{ME}} + \alpha \cdot (P_{\text{DE}} - P_{\text{ME}})$$

The parameter $\alpha \in [0,\ 1]$ determines the position of the point IMD according to the segment (DE$-$ME). For instance, for $\alpha = 1$ the IMD coincides with DE, whereas for $\alpha = 0$ it coincides with ME. One can find the optimal value of α with an optimization algorithm, minimizing the mean distance between the touch points and the IMD target. Let $P_i \in \mathbb{R}^3$ be the position of the i-th tested object, with $i = 1, \ldots, n$, and $T_{ij} \in \mathbb{R}^2$ be the actual touch point of the j-th trial ($j = 1, \ldots, m$) for the object at P_i, and P_{IMD_i} is the (unknown) position of the intermediate target for the i-th object. Then the optimization could be expressed as:

$$\min_{\alpha \in [0,1]} \left(\frac{1}{m \cdot n} \sum_{i=1}^{n} \sum_{j=1}^{m} \| T_{ij} - P_{\text{IMD}_i} \| \right)$$

Using this equation the optimal α value for the LED group is 0.551 with mean error $\varepsilon = 0.0266$, i.e., $1.07cm$ for a subject with $180cm$ body height. For the RED group we have determined $\alpha = 0.165,\ \varepsilon = 0.0365$ ($1.47cm$), which suggests that the subjects in the RED group choose a slightly different strategy than the subjects from the LED group. The calculated mean distances to the actual touch points are in all cases considerably smaller than the mean distances to the DE or ME targets.

Apparently, the optimal α may be influenced by several parameters such as the parallax, the user's handedness, performance speeds and preferences. Nevertheless, the reported values could be used to optimize the selection of a stereoscopically rendered virtual object on an interactive surface if the user's eye dominance is known. We expect even greater improvements by using parallax and eyedness dependent α values, which will be addressed in future works.

2.5.1 Confirmatory Study

In this section we describe a confirmatory study in which we applied the results of our experiment in a real-world application. The test application has been developed in the scope of the AVIGLE project [zAV], which explores novel approaches to remote sensing using a swarm of *Miniature Unmanned Aerial Vehicles (MUAVs)*, equipped with different sensing and network technologies. The acquired images are sent in quasi-real time to a flight ground control station. The user can interact with this visualization, for instance, by changing the viewpoint to the virtual environment. In addition, she can define new positions for each MUAV moving its visual representation in the virtual environment (see Figure 2.9). Since MUAVs within a swarm usually fly at different altitudes, stereoscopic visualization is essential to provide additional depth cues to show the altitude of each MUAV to the operator (see Figure 2.9). In order to select the correct MUAV from the swarm, it is important to determine the exact touch target for each virtual MUAV as described above. The goal of the confirmatory study was to verify if operators of the AVIGLE system perform better with the touch targets that we determined in the experiments in comparison to the other approaches.

Procedure

8 expert operators of the system participated in this study (6 had right eye dominance and 2 had left eye dominance). In a within-subject design experiment, we placed the operators in front of the stereoscopic multi-touch surface used in the experiment (cf. Section 2.4.1). The visual stimulus was a typical scene of our application showing a view with a swarm of 12 stereoscopically displayed virtual MUAVs (see Figure 2.9). We tested a subset of 42 locations from our initial experiment for the swarm. The position of each MUAV within the swarm and its altitude was randomized in each trial with respect to the minimal and maximal inter-MUAV distances. In each trial a MUAV was highlighted and the operator's task was to select it. We gave visual feedback about the selection, so that the operator could retry until the highlighted MUAV was selected. We tested two different on-surface targets, i. e., DE and ME, against the IMD. In order to simplify the confirmatory study, we averaged the IMD across both groups, thus we used $\alpha = 0.4$ in all cases. The swarm's position and the on-surface targets were randomized and uniformly distributed. To determine if a MUAV had been selected we constructed a ray with origin at the position of the dominant eye (for DE), or at the camera's position (ME), or shifted by $0.4 \cdot (\text{IOD}/2)$ towards the dominant eye (IMD) and the actual touch point; IOD denotes the interocular distance. A collision test between a cone around the ray with radius $0.03 \cdot l$ ($1.21cm$) at the projection wall and the mesh of each drone was used to determine the selection. The radius $0.03 \cdot l$ ($1.21cm$) was half the standard deviation (cf. Section 2.4.2). We measured the number of errors in terms of the number of repetitions required to select the correct MUAV, as well as the time required to perform the task.

Figure 2.9: Multi-touch interaction with a swarm of virtual MUAVs flying over a virtual city model.

Results

The mean number of touches the operators required to select the correct MUAV was 2.15 ($SD = 2.291$) for the touch target DE, 2.10 ($SD = 1.951$) for the touch target ME, and 1.73 ($SD = 1.634$) for the touch target IMD. We have analyzed the mean number of touches for each target and subject with a one-way ANOVA over all trials. We have found a significant main effect ($F^2_{1005} = 4.47,\ p = 0.12$) of the touch target on the number of touches required to hit the correct MUAV. Post-hoc analysis with the Tukey test showed that operators required significantly less touches to hit the correct MUAV, when we used the IMD touch target instead of the touch targets DE ($p = 0.018 < 0.05$) or ME ($p = 0.042 < 0.05$). We have not found a significant difference between the number of touches required to hit the MUAV when the touch target was ME or DE. Operators required approximately 1.7 touches to select the highlighted MUAV using our approach. This large value is caused by the small radius of the touch target as explained above. However, in a swarm of several MUAVs, touch targets may overlap if their radii are chosen too large.

The mean time the operators required to select the correct MUAV was $2.77s$ ($SD = 2.819$) for the touch target DE, $3.93s$ ($SD = 7.836$) for the touch target ME, and $2.65s$ ($SD = 4.421$) for the touch target IMD derived from our experiments. We have analyzed the mean performance time for each target and subject with a one-way ANOVA over all trials. We have found a significant main effect ($F^2_{1005} = 5.687,\ p = 0.012$) of the touch target on the time required to hit the correct MUAV. Post-hoc analysis with the Tukey test showed that operators required significantly more time to hit the correct MUAV, when we used the ME touch target ($p < 0.05$ for both IMD and DE). We have not found a significant difference between the time required to hit the MUAV when the touch target was DE or IMD ($p = 0.958$).

2.6 Current Limitations and Future Work

In this chapter we have discussed the main problems of 3D touch interaction with stereoscopic visualizations and have enumerated three different strategies to handle the problem: moving the interactive surface, moving the touch points or using perceptual illusions, which will be further investigated in more detail in the following chapters. Furthermore we presented the results of the first steps to analyze the relation between 3D positions of stereoscopically rendered objects and the *on-surface touch point*, where the user touches the surface. Therefore, an experiment was performed in which we determined the positions of the users' touches for objects which are displayed with different parallaxes. The results of the experiment show that users tend to touch between the projections for the two eyes with an offset toward the dominant eye's projection. We gave guidelines to set the on-surface touch points for stereoscopically displayed objects on a multi-touch surface. These guidelines depend on the user's head position as well as eye dominance; we explained how both can be easily determined. We have verified these guidelines in a real-world application and showed the benefits in terms of task performance over other approaches. Our results give novel implications for the development of future touch-enabled interfaces which support stereoscopic visualization.

While these initial findings provide useful insights into how users touch 3D stereoscopically displayed objects on a 2D touch surface, further studies are required to fully understand users' strategies in such setups. First, the scope of the experiment can be expanded to include varying user positions and orientations, as well as objects displayed with larger parallaxes or different projection sizes. Furthermore, we will more deeply consider the impact of handedness as well as eye dominance. The question arises if the IMD point can be formulated to model all these factors. We will extend this research and consider also other stereoscopic multi-touch surfaces such as table-top or mobile devices.

The combination of multi-touch technology and stereoscopic display provides an enormous potential not only for simple selection tasks, but also for richer interaction such as 3D manipulations of or collaborative interactions with stereoscopically rendered virtual scenes. These and similar research questions and challenges will be addressed in the future.

3 Chapter 3

Imperceptible Motions

Reality is merely an illusion, albeit a persistent one.

(Albert Einstein)

3.1 Perceptual Illusions for 3D Touch Interaction

As mentioned in the Introduction, the main benefit of multi-touching stereoscopic objects is that it allows us to get closer to the basic goal of "natural" interaction by building upon skills which humans have developed in their everyday lives interacting with real world objects. In particular, the user perceives virtual objects stereoscopically, i .e., with their associated depth properties, while she is able to interact with those objects with her own hands and fingers and thus receives direct or indirect haptic feedback.

The lack of haptic feedback is a common issue for interaction with virtual content which may reduce the "naturalness" of the interaction and often increases the amount of overshooting errors and reduced precision [BKLP04, Min95]. Specialized hardware devices exist which use mechanical or ultrasound actuators, e .g., [CSL+13, KIP13, Rek13], to provide active haptic stimuli. Although these technologies can provide compelling stimulation, they are usually cumbersome to use and have a limited application scope [CSL+13].

In fully immersive or head-mounted display (HMD) environments *passive haptic* feedback, which is provided by physical props registered to virtual objects, has been shown to be beneficial [Ins01]. For instance, a user might touch a physical table while viewing its (potentially different) representation in the virtual environment. Nevertheless, until now only little effort has been undertaken to extend this approach into non-HMD, projection-based setups. Theoretically, the display itself might serve as physical prop and provide passive feedback for the objects visually aligned with its surface (as it is the case in 2D visualizations). At the same time the point of finger or palm contact with the display can be tracked and used as input for interaction, which adds a powerful extension to this approach. Going a step further, one may separate the touch and the visualization surfaces, e .g., using a *transparent prop* as pro-

posed by Schmalstieg [SES99], which considerably increases the interaction volume in which touch-based interaction is available.

An alternative approach would be to move the virtual objects to the display surface, but with the important requirement that all object or scene manipulations are applied imperceptibly for the user. While this requirement would lead to relatively shallow object distribution within the stereoscopic volume in front of and behind the display, it allows us to combine stereoscopic visualization with direct object manipulation and haptic feedback in a single interaction interface. Thus the sacrifice of available depth range allows us to greatly improve the user's interaction experience, providing her with haptic *and* stereoscopic cues, but without loosing the directness of the interaction. In a more general sense, our concept is following the *perceptual illusions* paradigm for interface design, as proposed by Steinicke and Bruder [BSWL12, SB13]. Interfaces based on this paradigm benefit from illusions which arise from misinterpretation of (deliberately manipulated) sensory information by the brain [SLE10]. In the domain of virtual reality the "redirected walking" technique is a remarkable example which is based on this paradigm [SBRH08, RKW01]. By ingenious manipulations of the virtual camera this technique reroutes the user to walk in circles while she believes devoutly to walk on a straight line. Although walking and pointing (or touching) are quite different activities, there are some similarities in the nature of the perceptual inconsistencies, which makes such perceptual illusions feasible in the context of 3D stereoscopic touch interaction. In particular, the approach is strongly motivated by the findings of many perception research groups, revealing that there is a certain amount of induced object or scene manipulations which (although perceivable) cannot be reliably detected by the human visual system (e. g. [BPMB05]). Thus either the entire scene or a single object could be manipulated with some technique such that this manipulation remains below the threshold of our attentional awareness. From these considerations the desire for *usable* manipulation techniques arises which may be applied to the virtual scene and benefit the user in the context of 3D stereoscopic touch interaction.

Since we are mainly targeting on augmenting pure virtual objects floating in front of or behind the display surface with passive tactile feedback, we suggest to manipulate the properties of either an object or the entire virtual scene in such a way, that:

(a) At the moment when the user reaches the display surface the intended object is aligned with this surface, such that tactile feedback is received (exactly) at this moment.

(b) The application of the manipulation technique remains imperceivable for the user, i. e., it remains below her perceptual detection threshold.

Obviously, manipulation may be applied during different phases of the overall interaction process, and different phases may require different types of manipulation techniques or different settings for the same manipulation technique in order to achieve the desired result.

In this chapter we investigate the applicability of the *perceptual illusions* paradigm to allow users to interact with stereoscopically displayed objects when the input is constrained to a 2D touch surface. Therefore we discuss a number of possible manipulation techniques and

Figure 3.1: Illustration of the user interaction states with a stereoscopic multi-touch enabled display.

describe a series of experiments, which we have conducted in order to determine the applicability of those techniques, which seemed most promising. In particular, we have evaluated the detection thresholds for the misalignment between visual and tactile contact as well as the user sensitivity to small induced depth shifts applied while the user reaches for an object or walks toward the display surface. Our findings show that there is a usable range stretching in front of and behind the display surface, in which both *scene* and *object shifts* can not be detected reliably, i .e., an object could be imperceptibly moved closer to the display surface while the user is walking toward the display and is reaching out to touch it, and thus at the moment of touch passive haptic feedback is provided.

3.2 User Interaction States

During observations of many users participating in numerous demonstrations and evaluations with the 3D touch setups in our laboratory, we were able to identify some typical user behavior pattern. Similar to interaction with large public displays, users change between different states of involvement [VB04]. Nevertheless, in contrast to public displays where the focus is on different ”levels” of user involvement and attracting the user’s attention is one major goal, in most non-public interfaces the user already intends to interact with the proposed setup or environment in order to fulfill her tasks. To generalize these empiric observations we adapt Norman’s interaction cycle [Nor98] resulting in the definition of three typical interaction states as shown in Figure 3.1.

In the *observation* state the user is at such a distance from the display that the whole scene is in her field of view. In this state often the goal of the intended interaction is formed and different strategies to achieve this goal are formulated. Thereafter, and after the most promising strategy is selected, the global task is subdivided into subtasks. The significance of this phase may vary widely with the setup, e .g., size, form, position and orientation of the display, as well as with the particularities of the specific task and interaction interface. For instance, with our approx. $3m$ wide stereoscopic interactive wall the user usually remains beyond arm-reach distance in order to keep track of the scene as a whole, i .e., to get the ”big picture”, and to identify new areas or objects for further local interaction. In contrast, with the 65" tabletop setup in our laboratory, where a large portion of the scene is within the field of view of the user, this phase is entered only once at the beginning of the interaction.

The user is in the *specification* state while she is within arm-reach distance from the surface but still not interacting. Thus, the objects or tasks to be performed have already been selected, but the corresponding actions have not yet been performed. We have observed that the user spends only a short period of time in this state, plans the local input action and speculates about the system's reaction. For small vertical displays and tabletops the user usually never returns to the observation phase, once she has entered the specification. In contrast, in large vertical display environments, where the display size makes it impossible for the user to see the entire scene at once when she is close enough to interact, the user may be forced to go back to the observation phase in order to recapitulate the current results and plan the next local interaction. In the observation state the user is, with our setup, approximately $1.5-2m$ away from the interactive surface, whereas during the specification state she is within $50-60cm$ distance from the screen. Thus, a key feature of the transition between the observation state and the specification state is that *real* walking (as opposed to virtual traveling metaphors) is involved.

Finally, in the *execution* state the user performs the actions planned in the specification state. Here it must be mentioned, that the execution itself is, in this sense, a process rather than a static state. With touch-based interfaces the user is applying an input action while simultaneously observing and evaluating the result of this action and correcting the input in a series of mutually-connected interactive *micro-cycles* or *tonic* action. Nevertheless, further subdivision of the interaction beyond this merely generalized *execution* state might quickly become a complex and controversial task, which is far beyond the scope of this thesis. Once the execution of the current action is finished, the user may return back to the specification and thereafter to the observation state to evaluate the results at a higher level with respect to the global task.

While the described user interaction states and the transitions between them are similar for different kinds of tasks and visualizations, the time spent in each state and the number of transitions between them heavily depends on the application scenario and setup. Nevertheless, there are some high level tendencies, as our observations have revealed. For instance, users frequently switch between specification state and execution state, while changes between observation state and specification state are rather rare or completely missing (indicated by the size of the arrows in Figure 3.1), depending on the display size, orientation and interaction goal. In tasks in which only local interaction is required or the entire display size is in the user's field of view, users usually do not need to switch back to the observation state at any time. In contrast, in front of a large vertical display and especially in scenarios where some interrelations between the scene objects exist, the users frequently step back to get the "big picture" after a partial task is considered finished. Furthermore, it is likely that the observed phases and user behavior are affected by the parameters of the particular visualization, i .e., brightness or contrast of the projection, the type of the presented virtual scene, etc. In particular, with vertical wall-size displays the user is more likely to step back at some point, if the scene contains objects rendered with negative parallax, as when there are only objects with zero or positive parallax.

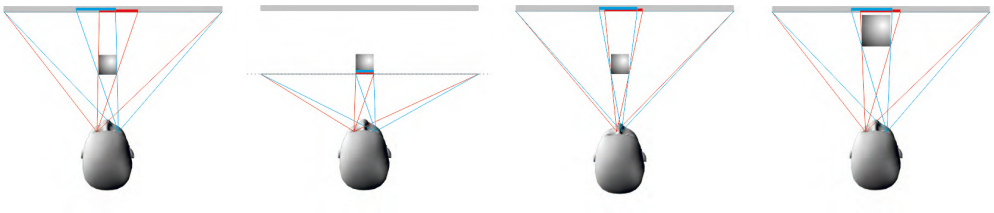

Figure 3.2: Illustration of different manipulation techniques: (a) neither the scene nor the visualization parameters are manipulated; (b) 3D scene rendered stereoscopically with manipulated focal distance; note how the projections for the left and for the right eye are aligned; (c) manipulated IOD, since the scene is rendered with smaller IOD, the projections for the left and for the right eye are closer to each other; (d) object shift – the object is shifted closer to the display surface and scaled accordingly, so that the size of the projection remains the same, while the stereoscopic disparity has changed.

The goal of this heuristic is not to provide a universal description of users' behavior while interacting with stereoscopic visualizations, but it illustrates many aspects involved in this process and underlines the states examined in our experiments.

3.3 Manipulation Techniques

In this section we are looking at some instances of the perceptual illusions paradigm, which might be useful for the problem at hand. Therefore we are considering techniques which manipulate the parameters of a particular object, of the virtual scene, or of the visualization itself, such that the object, which is intended to be touched, appears closer to the display surface after application of the technique. The application of a technique should be imperceptible for the user in a wide depth range. Thus we are looking for techniques, which will allow us to manipulate the perceived depth of the object of interest, while maximizing at the same time the available depth vicinity in front of and behind the display in which scene objects may be placed. In the following, possible manipulation techniques which likely satisfy these requirements are considered, and the constraints for their applicability are discussed.

Manipulation of the Visualization Parameters

The first group of manipulation techniques deals with manipulation of the parameters which are characteristic for the stereoscopic visualization, i .e., virtual inter-ocular distance (IOD) and focal length. One can align the object of interest to the plane of the display surface by adjusting the focal length of the virtual camera in such a way that the object in question moves to zero parallax (s. Figure 3.2(b)). In this case, all objects in front of the intended one will have negative parallax, all objects behind it will have positive parallax and all objects at the same depth will lie on the zero parallax plane. One advantage of this technique is that objects keep their relative distances to each other, since they are not moved within the scene. In addition, perspective distortion, lights or shadows remain unchanged. However, with objects

vastly scattered in depth, this technique could move the perceived depths of some objects too close or too far away from the observer, i.e., on strong parallax, causing uncomfortable viewing conditions. Furthermore, our preliminary tests have shown that misalignment of the depth of the zero parallax plane with the user's head-plane distance leads to substantially impaired viewing adaptation and thus to strong eye strains and exhaustion.

Another approach is to modify the inter-occular distance (IOD) between the camera for the left and the camera for the right eye, i.e., to modify the stereoscopic disparity of the scene objects as shown in Figure 3.2(c). Again, perspective distortion, lights or shadow cues remain unchanged. Nevertheless, gradual reduction of the IOD to 0 will effectively lead to monoscopic visualization, which contradicts the benefits of the interface itself. On the other hand, increasing the IOD beyond some value usually leads to merging problems and diplopia. Again, our observations have revealed that frequent variation of the IOD usually results in strongly disturbed viewing conditions, which quickly result in eye strains.

Manipulation of the Spatial Parameters

Instead of altering the stereoscopic parameters of the visualization, one could simply shift either the object of interest or the entire scene to the desired position, as illustrated in Figure 3.2(d). Shifting the entire scene has the advantage that the spatial relations between the objects remain unchanged. In particular, since the light sources are usually considered as part of the scene, lighting and shadows do not change either. Nevertheless, motion of an object close to the edge of the screen may easily be detected, since the bezel of the display provides a non-manipulative reference for the object's position. This technique may be especially suitable in large scale display setups, which usually cover more than 60° of the horizontal field of view of the user. Moving a single object can reduce this problem, especially if the object is far away from the display's edge. While changes in the object's shading and its shadow's position or form may reveal its motion, there is a wide range of applications, e.g., GUI elements or chemical visualizations, where the objects and lighting are primarily synthetic and the inter-object relation is not important. In applications where this is not the case, the visualization framework could still compensate for these changes by altering the shading algorithm or fitting the shadow volume calculations for this particular object. In addition, there is a wider range of manipulations applicable to a single object (s. Figure 3.2(d)). For instance, it could be moved along some curved path or along the line between its center and the position of the virtual camera, its size could be changed during the shift, etc. Because of this we consider investigation of object shifts as an important first step in this direction.

At first glance, the idea to imperceptibly shift a stereoscopic object or the scene in depth looks like contradicting its own benefits. On one hand, we claim that binocular disparity cues are precise enough to significantly increase the user's understanding for a 3D scene, i.e., the understanding "where the objects are" in the world and relative to each other. On the other hand, we suggest to either shift the virtual scene or to change the depth of one particular object and assume that the user's depth perception is imprecise enough to not recognize this.

The coherence of both assumptions lies in the understanding that human perception of self motion and position (whether we consider displacement of the person itself, e .g., locomotion, or motion of some body part) is mainly dominated by *exteroceptive* environmental stimuli, thus it is predominantly ruled by cues and landmarks extracted from the environment [Ber00]. Our *proprioception* gives us some sense of the relative position and motion of the parts of our body and lets us approximate the result of a particular effort. The results of such approximations are then fed in complex feed-back and feed-forward control paths and mixed together with signals resulting from the evaluation of the related exteroceptive stimuli, which results in generation of corrective muscle signals. Thus we do not have some "global positioning" sensation, but predominantly judge our own position, speed and direction based on the perceived environmental cues, which we have already classified (due to experience or learning) as static [Ber00]. While large discrepancies between proprioceptive and exteroceptive cues might be communicated to cognitive networks at higher hierarchical levels and thus become available to our awareness, many investigations have shown that there is a certain amount of mismatch, which is below the threshold of our conscious perception [BSWL12, BPMB05, CKC+10]. Therefore, in interfaces where real walking is involved it sounds reasonable that the virtual scene could be imperceptibly moved along or against the user's motion direction, such that a floating object is shifted onto the interactive surface potentially providing passive haptic feedback.

On the other hand, in many multi-touch interfaces the user remains static in front of the interactive surface. Nevertheless, the same considerations let us assume that it is possible to apply a similar approach while the user is reaching out to touch an object. The crucial point here is the moment in which the manipulation is applied. Since touching is only a small part of the interaction process, we can assume that the user already has built a *cognitive map* [Kit94] of the environment. In this case changing the position of an object would be perceived as object motion within a static scene rather than scene motion while the object is remaining static. Thus the spatial understanding of the scene would not be disturbed even if the object motion is detected. Furthermore, the total arm movement during reaching for an object consists of two distinct phases. During the *ballistic* phase the hand is moved close to the target, and during the subsequent *correction* phase the error between the hand or finger and the target is minimized under control of visual, haptic and proprioceptive feedback [LCE08]. Thus the mechanical control in the correction phase, i .e., which muscle-control signal must be applied and what would be the result, is only approximative [LCE08]. Therefore, we expect that if we slightly move the target within the correction phase, the arm control would change to accommodate to the new target position. Since reaching is a very low-level task, controlled through the dorsal visual pathway, we expect that the user will not consciously detect the motion, provided it remains within certain thresholds.

Considering the user interaction states and transitions between them, one can see that there are mainly two instances where a manipulation might be applied – (1) while the user remains in one particular state, (2) during transitions from one state to another. Though it is possible to manipulate an object's spatial parameters by applying temporal drifts, while

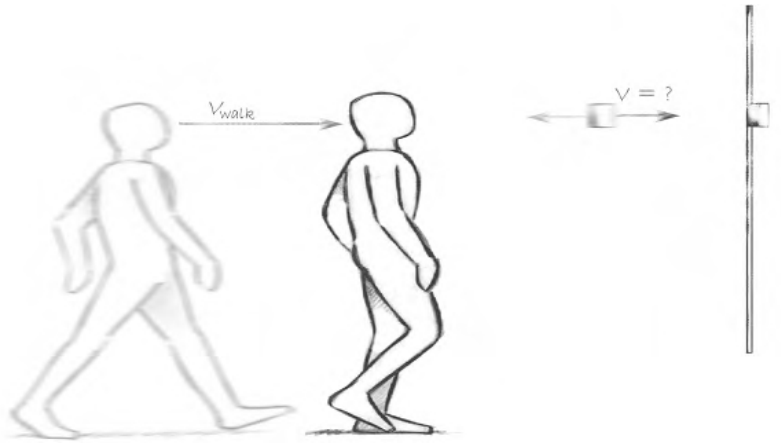

Figure 3.3: Illustration of object/scene shifts while walking. While the user is walking toward the display surface either an object or the entire virtual scene is shifted in or against the walking direction with a fraction of the walking speed.

the user's attention is focused on an object, it has been shown that the magnitude of such drifts has to be very small in order for the manipulation to remain imperceptible [Raz05]. This makes temporal drifts mostly unpractical. Thus, we could basically apply manipulations during transitions between states, which we consider in detail in the following sections.

3.4 Scene Shifts while Moving toward the Surface

First we will consider the transition between the observation state and the specification state, which is characteristic for interaction with large vertical display environments such as powerwalls, CAVEs, etc. As mentioned previously, a characteristic for the transition between these two states is that walking is involved, i.e., while in the observation state the user is, with our setup, about $1.5m$ – $2m$ away from the surface, such that the entire display is in her field of view, and she is at arm reach distance (about $0.5m$ – $0.6m$) when in the specification state. Thus the user has to make a couple of steps toward the display in order to switch from observation to specification state, and vice versa.

As mentioned previously, exteroceptive environmental stimuli usually dominate the proprioception. Moreover, it has been shown that visual information mostly dominates *extraretinal* cues, such as proprioception, vestibular signals, etc., in a way that humans usually experience difficulties to detect induced discrepancies between visually perceived motion and physical movement of their body [KBMF05, PWF08]. In this context, the question arises, if and how much a virtual scene can be imperceptibly shifted in depth during a user's transition from the observation state to the specification state. As illustrated in Figure 3.3, one can slightly translate the virtual scene in, for instance, the same direction as the user's motion and with speed proportional to the user's motion speed, while she is approaching the screen. Thus, an object of interest, which had negative parallax, may be shifted toward the interactive surface, where the user would receive passive haptic feedback if she touches it.

In most stereoscopic display setups the user's head motions in the real world are captured by a tracking system and mapped to translations (and rotations) of the virtual camera so that the virtual scene appears static from the user's point of view. Since humans usually tolerate a certain amount of instability of the virtual scene, we can describe our deliberately induced scene motion as instability with a translation shift $T_{shift} \in \mathbb{R}^3$, i.e., if $P \in \mathbb{R}^3$ is the *perceptually stable* position of an arbitrary object and $P_{shift} \in \mathbb{R}^3$ is the shifted position of the same object, then:

$$P_{shift} = P + T_{shift}$$

In most cases no scene shifts are intended, thus $T_{shift} = 0$. Nevertheless, due to latency, jitter, drift and the finite resolution of the real world position, measured by any tracking system, $T_{shift} = 0$ is merely a theoretical concept, thus one usually has:

$$\begin{aligned} T_{shift} &= \pm\varepsilon_{tracker} \\ &= \pm(\varepsilon_{lat} + \varepsilon_{drift} + \dots) \\ &\approx 0 \end{aligned}$$

In our setup we want to induce depth shifts in the same or in the opposite direction as the motion of the virtual camera, which are considerably larger than the tracker's imprecision $\varepsilon_{tracker}$. Therefore, we define the *scene shift factor* $\rho \in \mathbb{R}$ as the amount of virtual camera motion used to translate the scene in the same or in the opposite direction, i.e.,

$$T_{shift} = \rho \cdot T_{camera}$$

with $|\rho \cdot T_{camera}| \gg |\varepsilon_{tracker}|$.
In the most simple case the user moves orthogonally to the projection screen, and her motions are mapped one-to-one to virtual camera translations. In this case a shift factor of $\rho = 0.3$ means that, if the user walks $1m$ toward the projection screen, the scene will be translated $30cm$ in the *same direction*, while with $\rho = -0.3$ the scene will be translated $30cm$ *opposite* to the user's walking direction.

In order to prove the applicability of the technique, its acceptance by the user, and most importantly, to determine the range, in which applied shift factors ρ remain under a user's perceptual detection threshold, a psychological experiment was conducted.

3.4.1 Experiment: Discrimination of Scene Shifts

In this experiment we analyzed subjects' ability to detect induced scene motion while approaching a large stereoscopic projection wall. Therefore, subjects had to discriminate whether a stereoscopically displayed virtual object moved in the same direction or opposite to their motion. For the experiment setup we have used a 300×200 cm screen with passive-stereoscopic, circular polarized back projection for visualization. Two DLP projectors with a resolution of 1280×1024 pixels provided stereo images for the left and the right eye

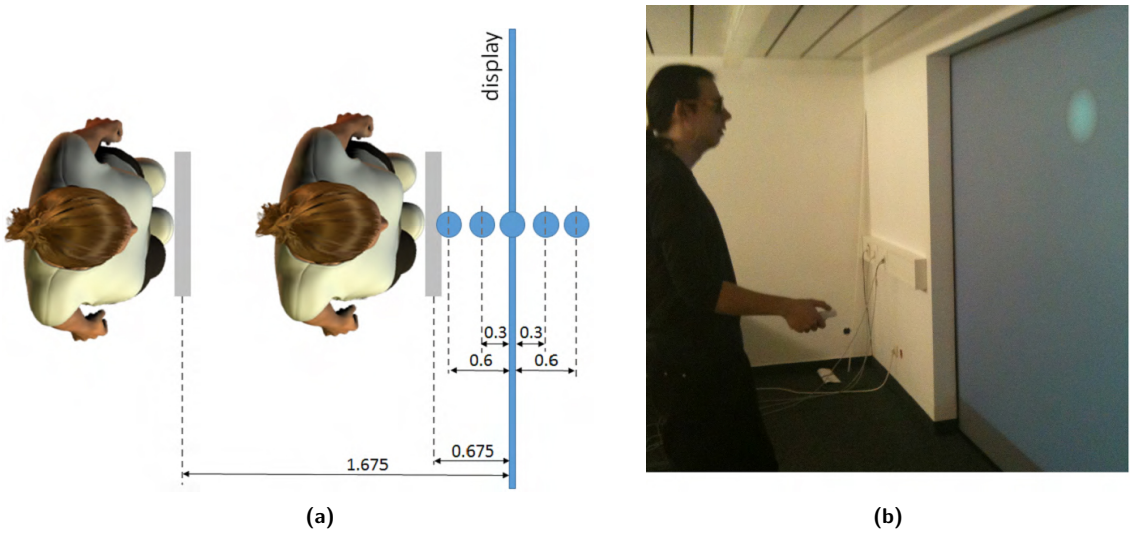

Figure 3.4: Experiment setup for the scene shift discrimination task; (a) schematic representation of the experiment setup and object start parallaxes; (b) illustration of a subject while performing the experiment task.

of the user. The virtual scene was rendered on an Intel Core i7 @ 2.66GHz processor with 4 GB RAM and nVidia GTX295 graphics card. We tracked the user's head position with an optical IR tracking system (InnoTeamS EOS 3D Tracking). The visualization setup was extended by Rear DI instrumentation in order to support multitouch interaction. Four IR illuminators were used for back-lighting the projection screen, and a digital video camera (PointGrey Dragonfly2) equipped with a wide-angle lens and a matching infrared band-pass filter and mounted at a distance of 3m from the screen was used to capture the reflected IR light. The camera captured an 8-bit monochrome video stream with resolution of 1024×768 pixels at $30fps$ ($2.95mm^2$ precision on the surface). Since our projection screen is made from a mat, diffusing material, we did not use an additional diffusing layer.

Participants

15 male and 4 female subjects (age 23-42, $\varnothing$: 26.9; height $1.54m$-$1.96m$, $\varnothing$: $1.80m$) participated in the experiment. Subjects were students or members of the computer science, mathematics and geoinformatics departments at the University of Münster. All had normal or corrected to normal vision, 11 wore glasses or contact lenses and none of the subjects reported known stereopsis disruption. 15 subjects had experience with stereoscopic projections, and 12 had already participated in a study in which stereoscopic projections were used; 7 had much video game experience, 10 some, and 2 none. All subjects were naïve to the experimental conditions. The total time per subject including pre-questionnaire, instructions, training, experiment, breaks, and debriefing took 45 minutes. Subjects were allowed to take breaks at any time.

Material and Methods

At the beginning of the experiment subjects judged the parallax of three small shapes displayed stereoscopically on the projection wall. We included this stereopsis test to confirm the subject's ability of binocular vision. If this test was accomplished successfully, a written task description and experiment walk-through was presented via slides on the projection wall.

The experiment layout and setup are illustrated in Figure 3.4. At the beginning of each trial, subjects were instructed to walk to the start position in front of the projection wall, which we marked with a white line on the ground. For visual stimuli we used a virtual scene that consisted of a single dark gray sphere projected at eye-height of the subject. To minimize ghosting artifacts of passive stereoscopic projection, we used a light gray color for the background. Once the virtual sphere was displayed, subjects had to walk forward to the projection wall until a written message indicated to stop. The walk distance in the real world was $1m$ in all trials. Subjects started $1.675m$ in front of the projection wall and stopped at their mean arm-reach distance. We determined the arm-reach distance as $0.675m$, i. e., the 3/8 part of the statistical median of the body height ($1.80m$) in our local area. In a *two-alternative forced-choice (2AFC)* task subjects had to judge with a Nintendo Wii remote controller if the virtual sphere moved in or opposite to their walking direction. The 'up' button on the controller indicated scene motion in the same direction as the subject, whereas the 'down' button indicated scene motion in the opposite direction. After subjects judged the perceived scene motion by pressing the corresponding button, we displayed a blank screen for $200ms$ as short interstimulus interval, followed by the written instruction to walk back to the start position to begin the next trial.

For the experiment we used the method of constant stimuli. In this method the applied shift factors $\rho \in \mathbb{R}$ as well as the scene's initial start positions are not related from one trial to the next, but presented randomly and uniformly distributed. We varied the factor ρ in the range between -0.3 and 0.3 in steps of 0.1, resulting in 7 different values of ρ. We tested five initial start positions of the stereoscopically displayed virtual sphere relative to the projection wall ($-60cm$, $-30cm$, $0cm$, $+30cm$ and $+60cm$). Since we have used a *right-handed* coordinate system, *negative* depth values represent *positive parallax* and vice versa. Each pair of start position and factor was presented exactly 5 times in randomized order, which results in a total of 175 trials per subject. Before the test trials started, 10 training trials in which we applied strong scene manipulations (factors $\rho = \pm 0.4$ and $\rho = \pm 0.5$) were presented to the subjects in order to ensure that they understood the task and received some initial training.

Results

In the analysis of the experimental results we have first used a repeated measurement ANOVA analysis to test the within subject effects of each tested factor, i .e., which factors had significant effect on the user estimations. Post-hoc analysis in this case would usually only show that there is no significant difference between subsequent values of a variable and that there is a significant difference between the others. For instance, the mean responses for $\rho = -0.3$

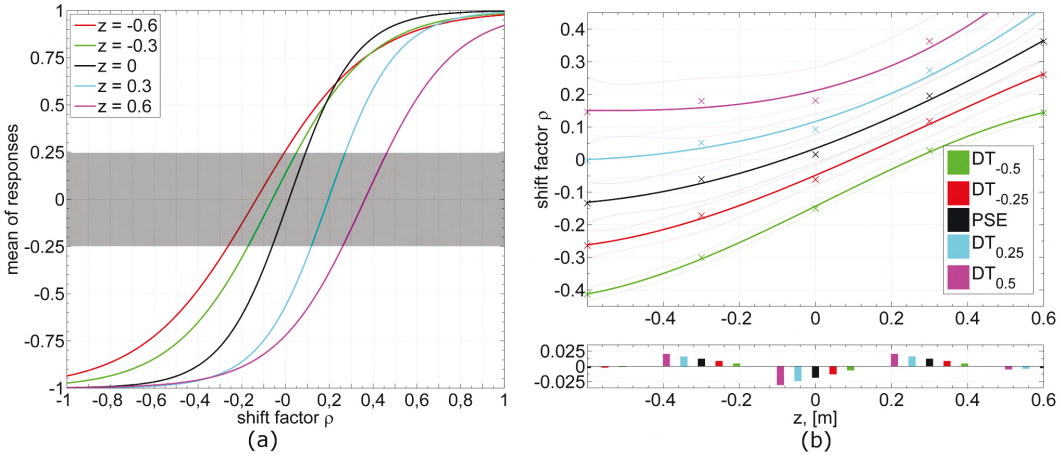

Figure 3.5: Experimental results for the discrimination task: (a) Fitted psychological sigmoid functions for each object start position z. The x-axis shows the applied shift factor ρ, the y-axis shows the mean responses. (b) Fitted polynomial functions for $DT_{\pm 0.50}(z)$, $DT_{\pm 0.25}(z)$ and $PSE(z)$ (top) and the fitting residuals (bottom). The x-axis shows the objects' start position z, the y-axis shows shift factors ρ. The solid curves show fitted polynomial functions and the light curves show the 75% confidence intervals.

differ significantly from those for shift factors $-0.1, 0, 0.1, 0.2$ and 0.3, but they do not differ significantly from those for $\rho = -0.2$. On the other hand, a multivariate logistic regression analysis relies on the implicit assumption that the factors are linearly dependent, which is rarely the case. Thus we used ANOVA to test which factors had effect and thereafter regression analysis on the significant effectors to determine the nature of these effects, i.e., to fit psychological sigmoid curves for the subject responses. These curves were then used to determine the absolute detection thresholds for the manipulation.

The subjects' mean responses were evaluated with 7×5 repeated measurement ANOVA, testing the within-subject effect of shift factor ρ and object start parallax z. Since the sphericity assumption was violated for parallax (Mauchly-Test $\chi^2 = 134.81,\ p \leq 0.01$) as well as for shift factor (M.-T. $\chi^2 = 124.11,\ p \leq 0.01$), the within subject effects were corrected with a Greenhouse-Geisser (G.-G.) corrector. We found a significant within-subject effect for both shift factor (G.-G. $F^{2.16}_{192.45} = 77.40,\ p \leq 0.01$) and parallax (G.-G. $F^{3.49}_{311.02} = 94.10,\ p \leq 0.01$). Therefore we split the results for different shift factors and parallaxes. The relation between mean subject estimations and shift factors were evaluated for different parallaxes with a binomial logistic regression analysis, fitting psychological sigmoid curves of the type $\frac{1}{1+e^{b \cdot \rho + c}}$ with parameters $a, b \in \mathbb{R}$. The estimated curve parameters and the regression statistics are summarized in Table 3.1 and the fitted curves are presented in Figure 3.5 (a). The x-axis shows the applied shift factors and the y-axis shows the probability for the user to estimate that the scene moved in the same direction as she walked scaled to fit the interval $[-1, 1]$. For instance, a y-value of -0.8 means that a user will estimate with 10% probability that the scene shifted in the same direction, i.e., that $\rho > 0$ and with 90% probability, that the scene shifted against her walking direction ($\rho < 0$). Similarly, a value $y = 0$ indicates 50/50

guessing in either direction. Usually a perceptual detection threshold is defined as the lowest intensity at which a stimulus can be detected at least 50% of the time, thus at least 75% of the answers were correct. Reflecting on the curves shown in Figure 3.5, one can see that the detection threshold for positive shift factors is the cut point of the curve with the line $y = -0.5$ and for negative shift factors with $y = +0.5$. We call these detection thresholds $DT_{+0.5}$ and $DT_{-0.5}$ respectively. Since some of the curves never reach these boundaries within the tested interval $\rho \in [-0.3, 0.3]$, we defined other, more restrictive, thresholds in which the shifts are detected at least 25% of the time, i.e., at least 62.5% of the subject estimations were correct. Obviously these are the detection thresholds $DT_{+0.25}$ and $DT_{-0.25}$. The estimated points of subjective equality (PSE) at $y = 0$ as well as the $DT_{\pm 0.5}$ and $DT_{\pm 0.25}$ for shift factors are summarized in Table 3.2 for the tested parallaxes. Figure 3.5 (b) shows a third-order polynomial interpolation of the relation between DT/PSE and parallax. Differences within the range defined by these thresholds cannot be detected reliably. For instance, for the $0\,cm$ start parallax subjects had problems to discriminate scene translations with shift factor ρ between -0.15 and 0.18. Thus subjects could not reliably detect if a virtual object initially aligned with the plane of the display surface moved $18\,cm$ in the same direction during $1\,m$ forward movement. Similarly, we could move the same virtual object up to $15\,cm$ against the user while she was walking $1\,m$ toward the display surface, and this motion was still indistinguishable in 75% of the cases. The possible object shifts for $1\,m$ subject motion are illustrated in Figure 3.6.

	model coefficients		model fitness			regression coefficients			
z, [m]	χ^2	p	χ^2	p	$\hat{R^2}$	b	p	c	p
-0.60	84.24	≤ 0.01	4.64	0.46	0.16	-3.931	≤ 0.01	0.526	≤ 0.01
-0.30	114.38	≤ 0.01	3.32	0.65	0.25	-4.579	≤ 0.01	0.277	≤ 0.01
0.0	209.13	≤ 0.01	4.05	0.54	0.36	-6.641	≤ 0.01	-0.104	$= 0.26$
0.30	163.17	≤ 0.01	2.38	0.79	0.31	-6.551	≤ 0.01	-1.280	≤ 0.01
0.60	78.13	≤ 0.01	4.47	0.48	0.18	-5.013	≤ 0.01	-1.817	≤ 0.01

Table 3.1: Table listing the regression coefficients and model goodness statistics from the logistic regression analysis: The first two columns show the results of the Omnibus test of the model coefficients, the third and fourth column show the Hosmer-Lemershow test of model fitness. The Nagelkerkes R^2, i.e., $\hat{R^2}$ are summarized in the fifth column. The regression coefficients b, c and their significance values are listed in the last 4 columns.

	opposite			same	
z, [m]	$DT_{-0.5}$	$DT_{-0.25}$	PSE	$DT_{0.25}$	$DT_{0.5}$
-0.60	-0.413	-0.264	-0.134	-0.004	0.146
-0.30	-0.300	-0.172	-0.060	0.051	0.179
0.0	-0.150	-0.061	0.016	0.093	0.181
0.30	0.028	0.117	0.195	0.273	0.363
0.60	0.143	0.261	0.362	0.464	0.582

Table 3.2: Table listing the detection thresholds $DT_{\pm 0.5}$ and $DT_{\pm 0.25}$, and the points of subjective equality *PSE* for each tested start parallax z.

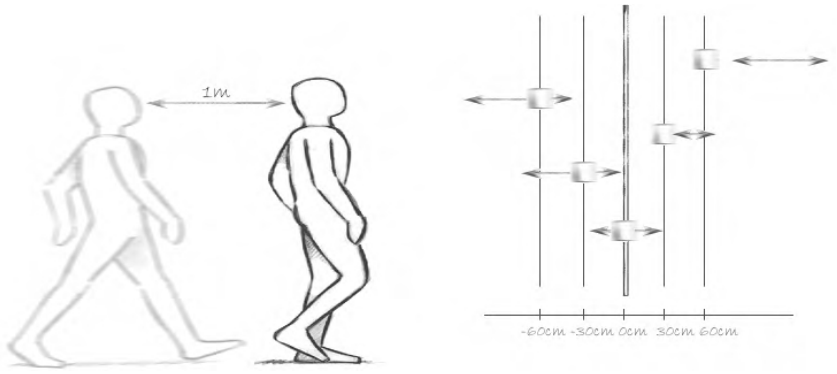

Figure 3.6: Illustration of the imperceptible scene shift ranges. The arrows indicate the maximal scene shift for $1m$ walking distance, which the user will not be able to reliably detect. The objects are distributed in depth according to their starting parallax.

3.4.2 Discussion

Our results show that subjects generally had problems to detect even large shifts of the stereoscopic depth of rendered objects during active movements, i. e., when approaching the projection wall by walking. In general, our results show smaller manipulation intervals than determined in similar experiments for HMD environments [SBJ+10]. This may be due to real-world references in our non-fully immersive setup as well as the short walking distances of about $1m$. Figure 3.5 (a) shows that for objects on the projection surface subjects were accurate at detecting scene motions corresponding to shift factors outside the interval between $\rho = -0.150$ and $\rho = 0.181$. For objects starting in front of the projection wall we observe a step-wise shift of the fitted psychophysical curves towards $\rho > 0$. The subjects rather show a significant bias towards underestimation of the motion speed of the virtual object relative to the observer's own motion. This result is in line with results found for underestimation of distances in studies conducted in HMD environments [SBJ+10]. However, we found this shift exclusively for objects displayed with negative parallax, which motivates that other factors may have influenced the results. For positive parallax subjects perceived the objects slightly moving opposite to their walking direction (with $\rho < 0$) as spatially stable. Compared to the results for objects in front of the projection wall, this result represents an overestimation of the subject's perceived self-motion relative to the virtual object. This difference to the results often found in fully-immersive environments may in part be caused by references to the real world in our projection-based experiment setup, such as the projection wall's bezel. While the estimated values may only apply for the simplified virtual scene in which the experiments were conducted, we hypothesize that more complex environments will differ only in quantity, i. e., detection thresholds, PSE, etc., and exhibit the same qualitative performance.

While the results of this experiment represent first steps towards touch interaction in stereoscopic projection environments, they are limited in various ways. For instance, the derived shift factors may be affected by the object's position in relation to the projection wall's bezel, since the bezel provides a non-manipulative reference to the user. Furthermore, the options to apply shift factors, while the user remains in the interaction area and only moves her hands,

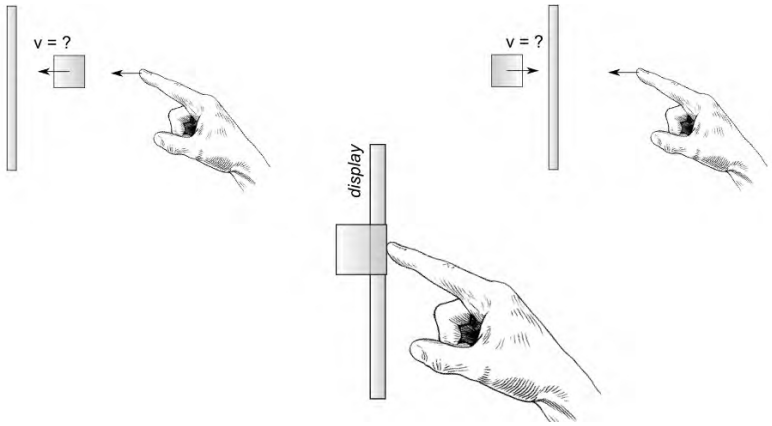

Figure 3.7: Illustration of object/scene shifts during touch. While the user is a touch gesture either an object or the entire virtual scene is shifted with or against her finger with a fraction of the fingers speed.

as well as rotational or curvature gains [SBJ+10] have not been studied sufficiently and will be addressed in future work. Nevertheless, from our initial application tests we believe that touch interaction has the potential to provide a vital enhancement of stereoscopic projection-based setups for a wide range of applications requiring touch interaction.

3.5 Object Shifts during Touch

As mentioned in Section 3.2 users spend most of their time in the *specification* and *interaction* states, which are in some setups the only reasonable user states, and frequently change from one state to the other. The specification might be considered as a form of *passive* interaction state, in which the actions necessary to fulfill the task are specified and the objects subject to these actions are identified. In the subsequent interaction state the user *actively* performs these actions, modifying this way the properties of the virtual scene or of a particular object, and compares the results with the results previously anticipated. Since touch-based interaction is in the focus of this thesis, the set of actions considered is dominated by point, touch and grasp gestures. In the specification phase the user identifies the next object to be touched or grasped and the specific type of touch or grasp gesture to be performed. In this context then the change from specification state to interaction state usually manifests itself by the user simply reaching out to touch or grasp the intended object. As in the previous section it seems worthwhile to investigate the possibility to imperceptibly manipulate the depth of a stereoscopic object while the user is reaching out to touch it, since this could allow us to shift that object to the surface and thus provide haptic feedback at the moment of touch. While it is likely that the possible magnitudes of such subtle manipulations are very small, there is a range of applications – discussed briefly in the following section – in which *shallow-depth 3D*, i .e., 3D interaction with limited depth, would be sufficient [HCC07]. In contrast to the scene manipulations while the user approaches the surface, as considered in Section 3.4, ma-

nipulations applied while the user is reaching out to touch an object are not limited to large vertical display setups. Thus, techniques relying on this kind of manipulation have potential to bring stereoscopic touch interaction to a larger set of hardware setups, including tabletops and tilt displays.

3.5.1 Application space

As indicated by Hancock et al. [HCC07] there are multiple application domains in which *shallow-depth 3D* would be sufficient. Assuming there is a usable range of imperceptible misalignments between visual and tactile touch (cf. Section 3.6), one could extend those applications with stereoscopic visualization without a need of complex 3D tracking of the user's finger. Through small induced object motions these ranges could be considerably increased. Although this comes at the cost of adding 3D finger tracking, in contrast to alternative techniques both direct object manipulation *and* haptic feedback are provided without additional instrumentation of the user (e .g., haptic gloves, phantoms, etc.).

For instance, a map viewer could render markers or widgets stereoscopically above the display in order to improve visibility and (especially if head tracking is supported) reduce occlusion artifacts. Those widgets, if rendered within some range, would still be accessible for touch interaction. Going a step further, one could overlay the map itself on a (possibly flattened) height model in order to improve spatial understanding of the representation. Many applications, such as the 3D desktops with stacked items on a tabletop setup, or graph visualizations in which highlighted nodes are rendered with different depths, may benefit from the same approach. The combination of stereoscopic touch interaction with the tangible views paradigm [STSD10] might also lead to a range of valuable interfaces. For instance, in a medical visualization a stereoscopic representation on the top of the tangible prop would support the tracking of long structures (e .g., veins, nerves) while touch interaction could be used to change the visualization properties at the same time (e .g., zooming, transparency).

In general, there is a wide range of applications, especially in urban planning or data visualization domains, in which shallow-depth 3D is sufficient, but may still benefit from additional stereoscopic cues and more natural interaction interfaces.

3.5.2 Manipulation of Stereoscopic Objects

In order to move an object to the display surface one could shift either only the object of interest or the entire scene to the desired position. As already discussed, shifting the entire scene has the advantage that the spatial relations between the objects remain unchanged. In particular, since the light sources are usually considered as part of the scene, lighting and shadows do not change either. However, our initial evaluations – in a psychological experiment similar to the one described in the previous section, but with shift factors applied while the user was reaching for an object – revealed a significant reduction of the perceptual detection thresholds. In particular, subjects were quite accurate in detecting shift factors outside the range $[-0.05; 0.07]$ for all tested parallaxes on a tabletop display. For instance, if we start

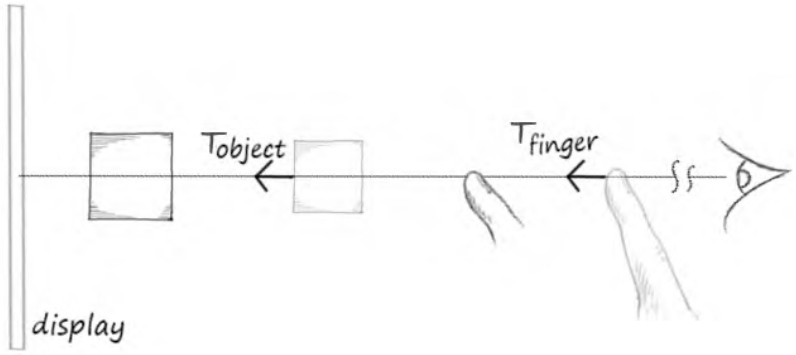

Figure 3.8: Illustration of the *scaled shift* manipulation technique. The object of interest is moved along the view line between its center and the camera position and scaled at the same time.

the manipulation $10cm$ before the user's finger reaches the object, we could only move the object $0.5cm$ against or $0.7cm$ with the finger, which is far below the requirements of most applications and – as discussed in Section 3.6 – might be achieved without any manipulation. Thus, although this technique proved suitable on a large display, which covers more than 60° of the user's horizontal field of view while the user walks toward the display, it seems to be inappropriate for manipulation of the stereoscopic depth while performing a touch gesture, especially in setups with limited display size, e .g., tabletops or desktop displays.

Moving a single object can reduce this problem, in particular if the object is far away from the display edge. Moreover, there is a wider range of manipulations that can be applied to a single object. For instance, an object could be moved along some curved path or along the line between its center and the position of the virtual camera, its size could be changed during the shift, etc. Nevertheless, since changes in the object's shading and its shadow's position and form may reveal the motion, the application should compensate for this. Therefore, we consider investigation of object shifts as an important first step in this direction and discuss a particular technique in more detail in the following.

Scaled Shift Technique

In order to be able to imperceptibly shift an object to the display surface we chose to move it along the line between its origin and the position of the virtual camera. Here, we consider the intermediate point between the cameras for the left and for the right eye as camera position. While [VSBH11] indicates that the intermediate eye point might be suboptimal and proposes a gradual shift (α shift) toward the user's dominant eye, the grade of this shift and the parameters on which it may rely are only vaguely defined [BSS13]. Thus we chose the intermediate eye point to confine variations due to inappropriate selection of the α shifts. While this ensures constant orientation of the object during translation, it still changes the size of its projection onto the image plane. Thus we have decided to simultaneously adjust the object's scale factor during translation relative to its motion. This manipulation technique, which we call *scaled shift technique*, is illustrated in Figure 3.8.

The technique reduces the number of motion cues on which the user may rely while still allowing us to use a number of visually different objects (e. g., different forms, textures, etc.). This is particularly important because imperceptible scene or object manipulations should only be applied in the correction phase due to the fact that the user focuses the object intensively during reaching in the ballistic phase, but pays less attention to the object during error correction and refinement in the correction phase. Indeed, once initiated the ballistic phase is carried out without further assistance from the visual system, which is usually scanning for changes in the scene [Car77]. In the correction phase the vision is switched back and forth between the object and the user's finger, which allows manipulations [Car77]. Since the correction phase is entered shortly before reaching the object, stronger manipulations are desired to move objects with strong parallax to the display surface. To provide smooth, undetectable manipulation we move the object depending on the motion of the user's fingertip. Thus, if $T_{finger} \in \mathbb{R}^3$ and $T_{object} \in \mathbb{R}^3$ represent the translations of the user's finger and the object of interest, respectively, we define an *object shift factor* σ as the relation between the magnitudes of these translations, i .e.,

$$t_{object} = \sigma \cdot t_{finger}$$

with $t_{object} = \|T_{object}\|_2$ and $t_{finger} = \|T_{finger}\|_2$. Similar to the ρ-shifts described in the previous section positive values ($\sigma > 0$) move the object of interest in the same direction as the finger, while for negative values ($\sigma < 0$) the object is moved in the opposite direction. For example, with shift factor $\sigma = 0.5$ the object is moved with half of the speed of the pointing finger in the *same direction* as the finger, while with $\sigma = -0.5$ the object is moved in the *opposite direction*. If $P_{old} \in \mathbb{R}^3$ and $P_{new} \in \mathbb{R}^3$ denote the object's position before and after the translation and $P_{camera} \in \mathbb{R}^3$ the camera position, we obtain

$$P_{new} = P_{old} + \sigma \cdot t_{finger} \cdot \nu$$

with direction vector $\nu = \dfrac{P_{camera} - P_{old}}{\|P_{camera} - P_{old}\|_2}$. The new scale factor s_{new} of the object could then be calculated as

$$s_{new} = s_{old} \cdot \frac{d_{new}}{d_{old}}$$

with $d_{old} = \|P_{old} - P_{camera}\|_2$ denoting the old distance between the object and the camera and $d_{new} = \|P_{new} - P_{camera}\|_2$ the new distance between the translated object and the camera. We assume that the correction phase starts when the user's fingertip is $10cm$ away from the object. Dvorkin et al. [DKK07] have shown that for virtual objects the correction phase is usually starting sooner than for real objects, which implies that our approximation may be too restrictive. However, precise determination of the start of the correction phase is beyond the scope of this work. Again the user's ability to detect induced object manipulations was tested in a psychological experiment, as described below.

3.5.3 Experiment: Discrimination of Object Shifts with the Scaled Shift Technique

In this experiment we examined participants' ability to detect induced σ shifts as described in the previous section. In a *2AFC* task subjects had to decide whether an object appeared to be moving toward or against their fingertip while reaching out to touch this object.

For this experiment we have formulated the following hypotheses:

- $H_0^{(1)}$: The point of subjective equality (PSE) is at $\sigma = 0$ and does not change with the parallax.
- $H_0^{(2)}$: The on-screen position of the object has little or no effect on the detection thresholds for σ, if the objects are reasonably far from the display edges.
- $H_0^{(3)}$: The subject's handedness has little or no effect on the detection thresholds for σ.

Experimental Setup

For the experiment setup illustrated in Figure 3.9 we used a $120Hz$ frame sequential stereoscopic visualization on a $22''$ Samsung SyncMaster display with nVidia's 3D Vision shutter glasses and an nVidia Quadro 4000 graphic card. The screen size was $47.9cm \times 29.5cm$, and the resolution was 1680×1050 on the upright positioned display. The scene was rendered on a PC with Intel Core i7 processor, and $8GB$ of RAM using two-pass, perspective corrected, on-axis stereo rendering, and eye-convergence on the display surface. We occluded noticeable objects with a black board and performed the experiment in a dark room to avoid distraction. Input was performed with a Belkin n52te Speedpad and a mouse fixed at a position convenient for the dominant hand of the user. A subject's fingertip was tracked with our in-house tracker, based on the Microsoft Kinect. Our tests showed that the precision of the tracker is better than $5mm$, and its accuracy is better than $1mm$.

Participants

20 persons were invited to the experiment, but two of them failed the stereoscopy test and were excluded from participation. 17 male and 1 female subjects (age $20-36$, $\varnothing : 25.8$, $SD : 3.79$) participated in the experiment. All were students or professionals from the department of Computer Science at the University of Münster and most of them reported to have experience with stereoscopic content, mostly due to 3D movies. 14 have reported to be right-handed and 4 to be left-handed. All subjects had normal or corrected to normal vision; 8 wore corrective glasses or contact lenses which they had to use during the experiment. All subjects were naïve to the experimental conditions. The total time per subject including pre-questionnaire, instructions, training, experiment, breaks, and debriefing took about 45 minutes. Subjects could take breaks at any time. In addition, after every 50 trials subjects had to take a 2 minutes break in order to minimize errors due to exhaustion or poor concentration.

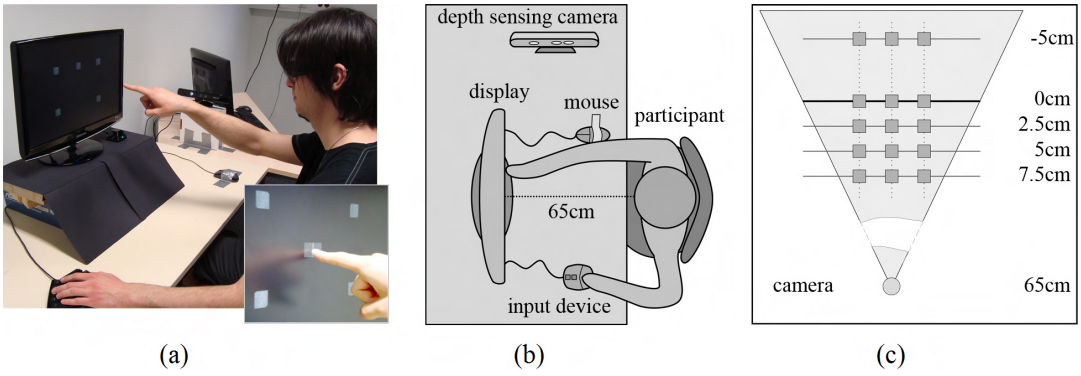

Figure 3.9: Illustration of the experiment setup for the scaled shift technique: (a) participant performing a touch gesture during the experiment; (b) experiment setup; (c) top-down view of object distribution.

Materials and Methods

Again a *within-subject design* was used, in which subjects had to decide in a *2AFC* task whether a highlighted object moved in the same direction as their fingertip or against it, while they were performing a touch gesture.

At the beginning of the experiment each subject completed a simple stereoscopy test with our apparatus. Subjects were seated such that the center of the screen was at their eye level at a distance of approximately $65cm$ (s. Figure 3.9). They were instructed to remain in this position during the entire experiment and to perform all trials with their dominant hand.

As visual stimuli (s. Figure 3.9 (a)) we used $2cm$ large boxes having a light grey texture with random non-repetitive pattern on black background, which ensured sufficient stereoscopic and perspective cues. Four additional static boxes having the same size, but slightly darker texture, were displayed on the zero parallax plane as reference with screen-space (origin top-left) coordinates $(0.25, 0.25)$, $(0.25, 0.75)$, $(0.75, 0.25)$ and $(0.75, 0.75)$.

The applied *shift factors* varied from -1.0 to 1.0 in steps of 0.5, resulting in five different values. To test the effect of different object positions and depths we varied the object's *screen space coordinates* using unified positions $(0.25, 0.5)$, $(0.5, 0.5)$, $(0.75, 0.5)$, $(0.5, 0.25)$ and $(0.5, 0.75)$, as well as the object's *parallax* using the depth planes $-5cm$, $0cm$, $2.5cm$, $5cm$ and $7.5cm$ (s. Figure 3.9). We mainly considered negative parallaxes, since objects floating in front of the display surface introduce the main problem for touch interaction. Again, we used a right-handed coordinate system, where *negative* depth values represent *positive parallax* and vice versa. With the method of constant stimuli the applied shift factors as well as the object positions and parallaxes were not related from one trial to another, but presented randomly and uniformly distributed. Each combination of shift factor, position and parallax was presented exactly two times, resulting in a total of 250 trials per participant. Additional 10 training trials with strong shift factors, ± 1.0 and ± 1.5, were added at the beginning of the experiment in order to ensure that participants understood the task and received some initial training. Subjects started each trial by pressing the left button of a fixed mouse. Then an object was displayed, and the subject had to reach out and touch it with the pointing

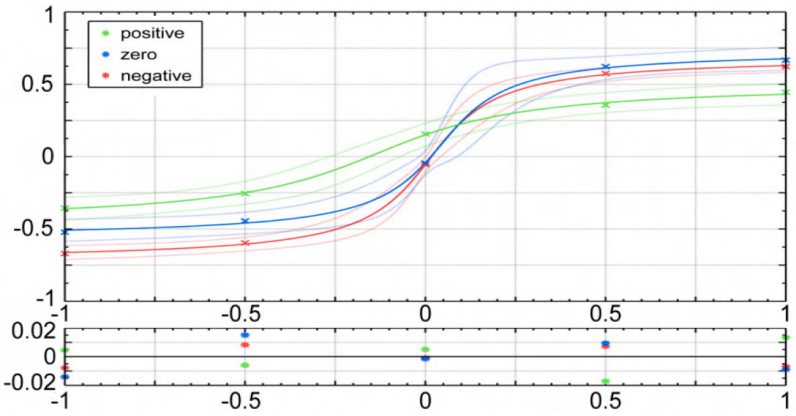

Figure 3.10: Experiment results of the discrimination task with the scaled shift technique: Fitted psychological functions for each parallax (top) and the fitting residuals (bottom). The X-axis shows the applied shift factor σ, the Y-axis shows the mean responses. The thin lines show the 95% confidence interval and the crosses the means of subject responses.

finger of her dominant hand. Once the subject's finger reached the display surface the object disappeared, and a written instruction forced the subject to decide whether the object moved in the same direction as her finger or in the opposite direction. Input was performed with the Speedpad using the 'up' button to indicate object motion in the same direction as the movement of the fingertip and the 'down' button for object motion in the opposite direction.

Results

We excluded the data of two subjects, since we found strong deviations in their results (more than twice the standard error over all participants) in mean responses for shifts, parallaxes and screen positions.

Since we have not found a significant difference between left-handed and right-handed subjects (two-sided t-test, $T_{16} = -0.39; p = 0.702$), we pooled the results over all subjects. The subject mean responses were evaluated with $5 \times 5 \times 5$ repeated measurement ANOVA with G.-G. correction, testing the within-subject effect of shift factor, object size and parallax. We found a significant within-subject effect for shift factor (G.-G. $F^{1.82}_{30.87} = 37.19,\ p \leq 0.01$), but not for position (G.-G. $F^{2.98}_{50.69} = 2.50,\ p = 0.07$) and parallax (G.-G. $F^{1.47}_{25.00} = 1.30,\ p = 0.28$). Nevertheless, the combined effect of parallax and shift factor was significant (G.-G.

Parallax	$DT_{0.5}$	$DT_{0.25}$	PSE	$DT_{0.25}$	$DT_{0.5}$
	opposite			same	
positive	–	−0.50	−0.16	0.14	–
zero	−0.40	−0.13	0.02	0.13	0.33
negative	−0.76	−0.09	0.02	0.13	0.40

Table 3.3: Estimated detection thresholds $DT_{\pm 0.25}$, $DT_{\pm 0.5}$ and points of subjective equality *PSE* for different parallaxes, when using the scaled shift technique.

$F_{95.72}^{5.63} = 3.07,\ p \leq 0.01$), while no significant effect of position and shift were found (G.-G. $F_{117.90}^{6.93} = 0.69,\ p = 0.68$). Therefore we split the results for different shift factors and parallaxes. The relation between mean subject estimations and shift factors was evaluated for different parallaxes with a parametric non-linear regression analysis, fitting psychological curves of the type $f(z) = a \cdot \arctan(b \cdot z + c) + d$ with real $a,\ b,\ c$ and d with the non-linear Levenberg-Marquard least square iteration algorithm.

Figure 3.10 presents the fitted curves and the residuals. The x-axis shows the applied shift factor σ, the y-axis shows the mean responses, where $+1$ indicates a subject's judgment that an object moved in the same direction as the finger and -1 indicates judgment for the opposite direction. The solid curves show fitted sigmoid functions for positive (purple), negative (cyan) and zero (orange) parallax, and the dotted curves show the 95% confidence intervals. The crosses represent the means of subject responses, while the stars represent the fitting residuals. The standard error of the regression ($RMSE$) for objects starting on positive parallax was 0.015 with summed square of residuals $SSE = 2.27 \times 10^{-4}$ and degrees of freedom adjusted R^2, i.e., $\hat{R}^2 = 0.999$. For objects starting on the zero parallax plane the $RMSE$ was 0.024 with $SSE = 5.98 \times 10^{-4}$ and $\hat{R}^2 = 0.998$, and for objects starting with negative parallax the $RMSE$ was 0.024 with $SSE = 5.57 \times 10^{-4}$ and $\hat{R}^2 = 0.996$. These values and the random nature of the residuals show that the model functions are good fits of the data and explain more than 99% of the variance. The estimated PSE, and the $DT_{\pm 0.5}$ and $DT_{\pm 0.25}$ for positive and negative shifts are summarized in Table 3.3. The detection thresholds show that subjects were accurate in detection of object shifts outside the interval $[-0.4, 0.33]$ for objects starting with zero parallax and outside the interval $[-0.76, 0.4]$ for objects starting with negative parallax. Participants were not able to detect shifts for objects starting with positive parallax with a probability of at least 75% in either of the two directions, but the $DT_{0.25}$ thresholds show that they had at least some good impression of the motion outside the interval $[-0.5, 0.14]$.

Discussion

Our results support the initial hypothesis $H_0^{(3)}$ that the detection of induced object shifts does not depend on the user's handedness. While the hypothesis $H_0^{(2)}$ has also been supported, it is (at the current state) not clear, if it still holds for objects closer to the bezel of the display or in densely populated environments with multiple objects close to each other.

Contrary to our initial hypothesis $H_0^{(1)}$, participants judged objects starting on positive parallax and moving against their fingertip as static and thus underestimated the distances to those objects. The PSE for shift factors of objects starting on the zero or negative parallax planes is nearly zero, which indicates that subjects are more sensitive to the motion directions in these cases. In contrast to the steeply ascending curves for non-positive parallaxes, the flat curve for mean results of objects starting on positive parallax hints to detection thresholds outside the range of $[-1.0, 1.0]$. This may be explained with the smaller absolute object shift during reaching for objects on positive parallax. Indeed, since we start manipulating the object's position when the user's finger is approximately $10cm$ away from the object,

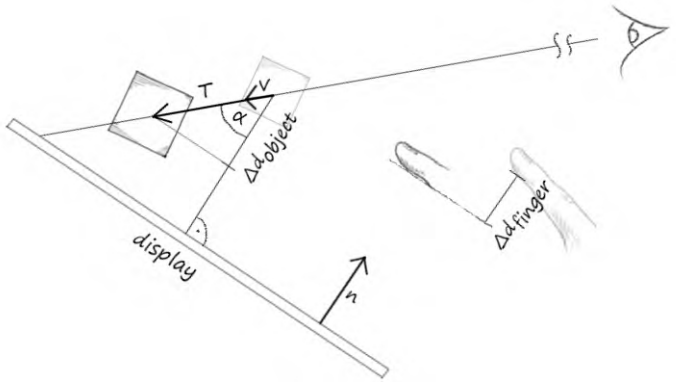

Figure 3.11: Illustration of the generalized scaled shift technique. The object to be manipulated is moved along the ray from the camera to its own position, such that its distance to the screen surface is proportional to the finger's depth.

the maximal hand motion until the finger reaches the display surface is only $5cm$. While the relative motion of the object is the same in all cases, the object is manipulated longer if starting on a negative or zero parallax. Thus the effective object motion even with strong shift factors is smaller compared to the absolute object motion for the other parallaxes.

3.5.4 Generalized Scaled Shift Technique

While the scaled shift technique compensates for many cues, which could reveal the object motion, it relies on the implicit assumption that in the correction phase the user's point of view, the object and the finger lie on the same line and the finger is moving approximately along the normal of the display surface (cf. Figure 3.8). While this assumption is reasonable for vertical displays, especially if the display surface is centered to the user's head, it is difficult to generalize the results for horizontal or tilt displays, where the user's hand is usually approaching the object from one of its sides (s. Figure 3.11). Therefore we have generalized the technique for arbitrary hand motions and thus for different display orientations. In our technique (illustrated in Figure 3.11), which we call *generalized scaled shift* technique, we ignore the exact position and motion vector of the finger and take into account only its orthogonal distance d_{finger} to the display surface. The object in question is then moved along the vector defined by its position and the position of the virtual camera, $\nu \in \mathbb{R}^3, \|\nu\|_2 = 1$, such that its depth change is a fraction of the finger's depth change, thus:

$$\Delta d_{object} = \sigma \cdot \Delta d_{finger}$$

with *shift factor* $\sigma \in \mathbb{R}$ as in the previous section.

Thus, if $\Delta d_{finger} = d^{old}_{finger} - d^{new}_{finger}$ is the relative depth motion of the user's finger, the total object translation $T \in \mathbb{R}^3$ could be expressed as:

$$T = \frac{\sigma \cdot \Delta d_{finger}}{\cos(\alpha)} \cdot \nu$$

where $\cos(\alpha)$ (cf. Figure 3.11) could be expressed as dot product of the motion vector ν and the display normal vector n, i. e., $\cos(\alpha) = -\nu \cdot n$. Since we are using a right-handed coordinate system, centered in the middle of the display, one has $n = (0 \;\; 0 \;\; 1)^T$, thus with $\nu = \frac{P_{object} - P_{camera}}{\|P_{object} - P_{camera}\|_2} = (v_x \;\; v_y \;\; v_z)^T$, the above equation could be expressed as:

$$T = \frac{\sigma \cdot \Delta d_{finger}}{-\nu_z} \cdot \nu$$

After unwinding the normalization we obtain

$$T = \frac{\sigma \cdot \Delta d_{finger}}{d_{camera} - d_{object}} \cdot (P_{object} - P_{camera})$$

with P_{object} and P_{camera} representing the 3D positions of the object and the virtual camera, respectively. The right side of the equation consists of two main components. The left term defines the amount of object motion as part of the camera to object vector and is only dependent on the depth relations between the finger, the object and the virtual view point. Since the objects and view point positions are known a-priori, one only needs to determine the proximity of the finger to the display surface in order to apply the technique. This greatly reduces the requirements of the finger tracking hardware, since only the finger depth, i .e., its distance to the display, has to be tracked precisely, which could be achieved with different techniques, e .g., [DVS+12, NS03].

The right term defines the scale and the direction of the object motion and captures the dependency on the head-tracking technique, if used. Indeed, in a head-tracked setup the position of the virtual view point is constantly adjusted to the user's head position, thus P_{camera} and d_{camera} in the above equation might change between frames. This may lead to small sidewards offsets, which may reveal the manipulation on a static background. As in the original scaled shift technique, the size of the object is adjusted to ensure constant projection size.

3.5.5 Experiment: Discrimination of Object Shifts with the Generalized Scaled Shift Technique.

In order to examine the participants' ability to detect induced object shifts σ with the generalized scaled shift technique in head tracked environments with different display configurations, a within-subject design experiment was conducted. In a *2AFC* task the subjects had to decide whether an object appeared to be moving toward or against their fingertip while performing a touch gesture.

Partially following 3.5.3 we formulated the following initial hypotheses:

- $H_0^{(1)}$: The point of subjective equality (PSE) is at $\sigma = 0$ for objects with zero or negative parallax and negative $\sigma < 0$ for objects with positive parallax (i. e., behind the display surface).
- $H_0^{(2)}$: The display's tilt angle has no significant effect on the detection thresholds.

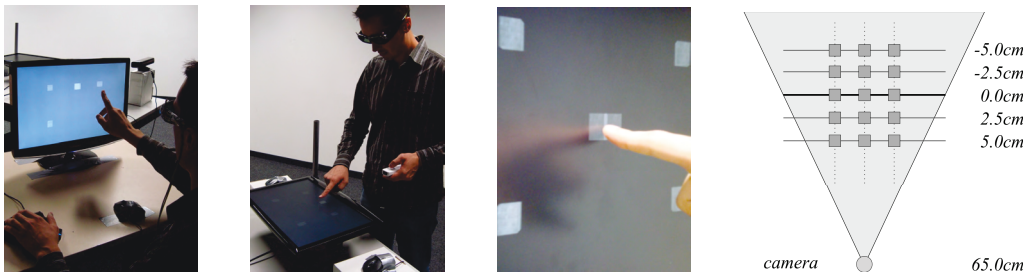

Figure 3.12: Illustration of the experimental setup; (a) in the VERTICAL setup the display was in an upright position; (b) in the HORIZONTAL setup the display was tilted by 90° degrees and in the TILT setup by 45° degrees (not shown in the figure); (c) the manipulated object was surrounded by some static objects; (d) top-down view of objects' arrangement.

Participants

29 male and 4 female students from our department (age $23-56$, M: 28.2, SD: 3.21) participated in the experiment. All subjects were naïve to the experimental conditions. 20 subjects were right-handed and 13 were left-handed. All subjects had normal or corrected to normal vision.

Experimental Setup

The experiment was conducted in 3 setups which were similar to the one described in Section 3.5.3. Again, we used a $120Hz$ frame sequential stereoscopic visualization on a $22''$ Samsung SyncMaster displays with nVidia's 3D Vision shutter glasses. The size of each screen was $47.9cm \times 29.5cm$, and the screen resolution was 1680×1050. Noticeable objects in front of the participants were occluded with black board and the experiment was performed in a dark room to avoid distraction. Input was performed with either a Belkin Nostromo n52te Speedpad or Nintendo's Wii controller and a fixed mouse. As in the previous experiment a subject's fingertip was tracked with our in-house tracker, based on the Kinect depth camera.

The displays in the three setups were arranged such that the subjects did not disturb each other during the experiment. In the first setup (VERTICAL), the display was in an upright position and the user performed input with the Belkin Nostromo n52te Speedpad (cf. Figure 3.12 (a)). The display in the second setup (TILT) was tilted by 45° degrees and in the third setup (HORIZONTAL) by 90° degrees (cf. Figure 3.12 (b)). In those setups input was provided with the Wii controller.

We used head-tracking in every setup. A subject's head was tracked with the WorldViz PPT optical tracker, which provides millimeter precision and sub-millimeter accuracy. We have tracked two markers per subject and adjusted the virtual cameras for the left and for the right eye relative to the positions of those markers. An additional offset relative to the subject's interocular distance was added to adjust the tracking data to the exact eye position for each subject.

Materials and Methods

In a *within-subject design* setting, subjects had to decide in a *2AFC* task whether a highlighted object moved in the same direction as their fingertip or against it, while they were performing a touch gesture. A subject had to start each trial by pressing the left button of a fixed mouse, which guaranteed consistent start position of the user's hand during the experiment. Afterward 5 objects, one of which was highlighted, were displayed, and the subject had to reach out and touch the highlighted one with the index finger of her dominant hand. Shifts were applied only to this object, while the other objects remained static and were aligned with the zero parallax plane (s. Figure 3.12 (c)). Once the subject's finger reached the display surface the object disappeared, and a written instruction on the display forced the subject to decide whether the object moved in the same direction as her finger or in the opposite direction. These instruction screens were displayed for at least $200ms$ to ensure the afterimages on both eyes were deleted. As visual stimuli we used a textured box in order to provide a sufficient amount of depth perception and stereoscopic disparity cues (s. Figure 3.12 (c)). The box was displayed with a light grey texture having a random, non-repetitive pattern on black background in order to reduce cross-talk artifacts. The applied *shift factors* varied from -0.5 to 0.5 in steps of 0.125, resulting in nine different values. To test the effect of different parallaxes, we varied the objects' *depth* from -5.0 to 5.0 in steps of 2.5 resulting in five parallax planes shown in Figure 3.12 (d). Negative depth values represent positive parallax and vice versa. Trials were divided into 3 blocks – one for each setup, and each participant had to perform all blocks. The same set of shift factors and parallaxes was tested in each block. For the VERTICAL and TILT setups users were seated at a convenient distance (approx. $65cm$) in front of the display, as shown in Figure 3.12 (a). For the HORIZONTAL setup users performed the trial while standing (cf. Figure 3.12 (b)). The order in which the users had to perform the blocks was randomized and counter-balanced to avoid ordering effects. Each combination of shift factor and parallax was presented exactly 5 times in randomized and uniformly distributed order, resulting in a total of 225 trials per participant. Additional 10 training trials were added at the beginning of each block in order to ensure that participants understood the task and received some initial training. The total time per subject including pre-questionnaire, instructions, training, experiment, breaks, and debriefing took about 60 minutes. Subjects could take a break at any time. In addition, after every 75 trials subjects had to take a break of two minutes in order to minimize errors due to exhaustion or poor concentration.

Results

The data of two subjects were excluded from evaluation, since they either misunderstood the task or mixed the buttons. Post-interrogation with one of the subjects revealed that he judged the objects' parallax rather than their motion. The second subject admitted that he was not able to detect any motion in most of the trials and deliberately pressed the left mouse button consistently. We did not find a significant difference between the mean responses for

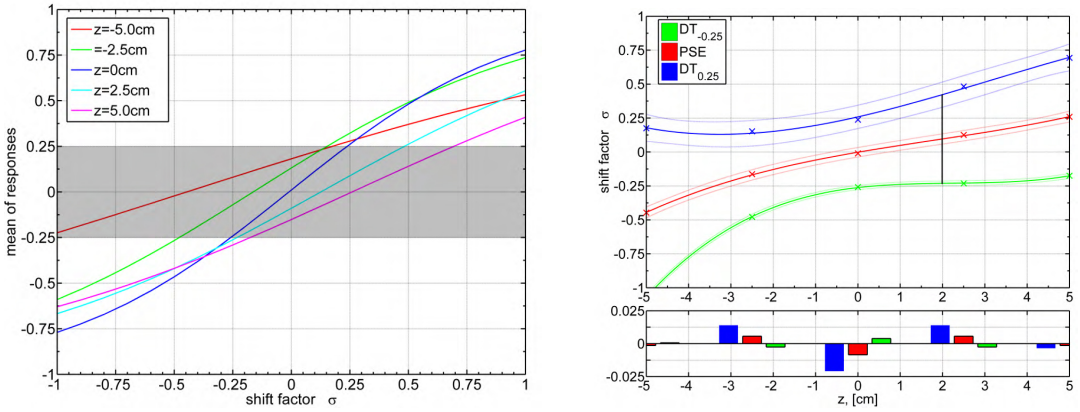

Figure 3.13: Experiment results for the generalized scaled shift technique: (a) Fitted psychological functions for each parallax. The x-axis shows the applied shift factor σ, the y-axis shows the mean responses. (b) Fitted polynomial functions for $\sigma_{max}(z)$, $DT_{\pm 0.25}(z)$ and $PSE(z)$ (top) and the fitting residuals (bottom). The x-axis shows the object's depth position z, the y-axis shows shift factors σ. The solid curves show fitted polynomial functions and the dotted curves show the 75% confidence intervals.

left-handed and right-handed subjects (two-sided t-test; $T_{29} = 0.39, p = 0.702$), thus we have pooled the results over all subjects. The subjects' mean responses were evaluated with $3\times5\times9$ repeated measurement ANOVA with G.-G. correction, testing the within subject effect of display tilt angle, parallax and shift factor. We found a significant within subject effect for shift factor (G.-G. $F^{1.49}_{44.76} = 30.81,\ p < 0.01$) and for parallax (G.-G. $F^{1.24}_{37.17} = 6.13,\ p < 0.01$). The sphericity assumption for tilt angle was not violated, but the result was not significant $F^2_{60} = 1.63, p = 0.21$. The combined effect of parallax and shift factor was also found to be significant (G.-G. $F^{13.34}_{400.24} = 3.21,\ p \leq 0.01$), while no significant effects of tilt angle and shift were found (G.-G. $F^{8.62}_{258.54} = 1.38,\ p = 0.144$). Therefore we split the results for different shift factors and parallaxes. The relation between mean subject estimations and shift factors was evaluated for different parallaxes with a logistic regression analysis, fitting psychological sigmoid curves of the type $\frac{1}{1+e^{-(b\cdot\sigma+c)}}$ for each parallax, with $b, c \in \mathbb{R}$. The estimated curve parameters and the regression statistics are summarized in Table 3.4 and the fitted curves are presented in Figure 3.13(a).

No residuals outside the twice standard error interval were found by the analysis. The fitted curves in Figure 3.13(a) are scaled for the presentation in the interval $[-1; 1]$. The x-axis shows the applied shift factor σ, the y-axis shows the mean responses, where $+1$ indicates a subject's judgment that an object moved in the same direction as the finger and -1 indicates judgment for the opposite direction. The solid curves show fitted sigmoid functions for strong positive (red), positive (green), zero (blue), negative (cyan) and strong negative (magenta) parallaxes. The estimated points of subjective equality (PSE), the $DT_{\pm 0.5}$ and $DT_{\pm 0.25}$ for positive and negative parallax are summarized in Table 3.5. The detection thresholds show that subjects were accurate in detection of object shifts outside the interval $[-0.26, 0.24]$ for

objects starting with zero parallax and outside the interval $[-0.18, 0.48]$ for objects starting with negative parallax. Within the tested ranges participants were not able to detect negative shifts for objects starting with strong positive parallax with a probability of at least 62.5%. Interestingly, while the model evaluation and goodness of fit coefficients of the regression model for objects on zero parallax confirmed the fitness of the model, the model constant term $c = 0.021$ was not found to be significant and the effect of shift factor $b = 2.059$ was found to be stronger as by the other parallaxes.

In order to evaluate the relations between object start parallax and the detection thresholds $DT_{\pm 0.25}$ and PSE, we have fitted third order polynomials $P(z) = az^3+bz^2+cz+d,\ \ a, b, c, d \in \mathbb{R}$ to the values. Table 3.6 shows the polynomial coefficients as well as the goodness of fit estimations. As one can see from this table, the polynomials are a good fit to the data and explain more than 99% of the variance. The resulting curves as well as the fitting residuals and 95% confidence intervals for new estimations are shown in Figure 3.13(b). The x-axis shows the object's depth position z, the y-axis shows shift factors σ. The solid curves show fitted polynomial functions for $DT_{-0.25}(z)$ (green), $DT_{0.25}(z)$ (blue) and $PSE(z)$ (red), and the dotted curves show the 75% confidence intervals. The crosses represent the means of subject responses, while the stars represent the fitting residuals.

Discussion

Our results support the initial hypothesis $H_0^{(2)}$ that the detection of induced object shifts does not depend on the display tilt angle. While the hypothesis $H_0^{(1)}$ has also been partially supported, i. e., the PSE for zero parallax is at $\sigma = 0$ for objects on the zero parallax plane and negative for objects with positive parallax, the results for objects starting with negative parallax differ considerably from those estimated in our previous experiment. In particular, subjects judged static objects with positive parallax as moving with their finger and thus underestimated the distances to those objects. In contrast, static objects starting with negative parallax were judged as moving against the user's finger, which indicates that the subjects overestimated the distances in these cases. The PSE for shift factors of objects starting on the zero plane is not significantly different from zero ($p = 0.103$ for the constant term of the

	model coefficients		model fitness			regression coefficients			
z, [cm]	χ^2	p	χ^2	p	$\hat{R}^2$	b	p	c	p
−5.0	70.26	≤ 0.01	3.19	0.87	0.022	0.822	≤ 0.01	0.367	≤ 0.01
−2.5	265.71	≤ 0.01	11.80	0.11	0.082	1.620	≤ 0.01	0.264	≤ 0.01
0.0	299.03	≤ 0.01	9.36	0.10	0.103	2.059	≤ 0.01	0.059	> 0.05
2.5	211.45	≤ 0.01	13.04	0.07	0.066	1.431	≤ 0.01	−0.180	≤ 0.01
5.0	142.98	≤ 0.01	2.99	0.89	0.045	1.176	≤ 0.01	−0.305	≤ 0.01

Table 3.4: Table listing the regression coefficients and model goodness statistics from the logistic regression analysis: The first two columns show the results of the Omnibus test of the model coefficients, the third and fourth column show the Hosmer-Lemershow test of model fitness. The Nagelkerkes R^2, i .e., $\hat{R}^2$ are summarized in the fifth column. The regression coefficients b, c and their significance values are listed in the last 4 columns.

Parallax	$DT_{-0.5}$	$DT_{-0.25}$	PSE	$DT_{0.25}$	$DT_{0.5}$
strong positive, $z = -5.0$	-	-1.07	-0.45	0.18	0.82
positive, $z = -2.5$	-0.72	-0.48	-0.16	0.15	0.51
zero, $z = 0.0$	-0.56	-0.26	-0.01	0.24	0.52
negative, $z = 2.5$	-0.60	-0.23	0.13	0.48	0.88
strong negative, $z = 5.0$	-0.63	-0.18	0.26	0.69	-

Table 3.5: $DT_{\pm 0.5}$, $DT_{\pm 0.25}$ and PSE for the shift discrimination task.

fitted sigmoid, cf. Table 3.4), which indicates that subjects are more sensitive to the motion direction in these cases. As shown in Figure 3.13(a), the flat sigmoid curve for objects starting on the $z = -5.0$ plane hints even the tighter detection threshold $DT_{-0.25}$ outside the tested interval of $[-0.5, 0.5]$. Again, this may be explained with the smaller absolute object shift while reaching for objects on positive parallax. For instance, a shift factor of 1.0 translates an object starting on the strong positive parallax plane a total of $5cm$, while the same shift factor translates an object starting on the zero parallax plane $10cm$.

Considering Figure 3.13(b) one can see that a vertical line segment between $DT_{\pm 0.25}$ defines the interval of possible shift factors, which may be applied to an object with given parallax. While for the proposed technique we only consider shifts, which move the objects closer to the surface (positive for objects with negative parallax and vice versa), one may consider moving the objects in either directions. A free-hand interaction technique may, for example, manipulate objects with negative parallax to move closer to the hand, while the user is grasping for them, thus reducing the overall magnitude of the hand motion. Alternatively a "smart" technique could selectively move an object closer or further away from the user's finger or from the interactive surface, depending on its accessibility or appropriateness for the current task. Overall, there is a range of possible applications for these and similar kinds of imperceptible object motions, which might be usable for interactive applications.

3.5.6 General Discussion and Design Implications

The results of both experiments show that there is a usable range of detection thresholds, which could be used to enable touch interaction with stereoscopically rendered objects, with little or no impact on the depth perception or touch performance. Indeed, the correction

	z^3	z^2	z^1	z^0	$RMSE$	SSE	$\hat{R}^2$
$DT_{-0.25}$	2.1×10^{-3}	-0.014	0.036	-0.262	0.005	2.91×10^{-5}	0.999
PSE	6.8×10^{-4}	-3.6×10^{-3}	0.053	-1.6×10^{-3}	0.011	1.41×10^{-4}	0.998
$DT_{0.25}$	-7.5×10^{-4}	7.1×10^{-3}	0.070	0.259	0.029	8.53×10^{-4}	0.984

Table 3.6: Coefficients and goodness of fit estimations of the fitted polynomials for $DT_{\pm 0.25}(z)$ and $PSE(z)$. The first four columns show the polynomial coefficients; the standard error of the regression ($RMSE$), the summed square of residuals SSE and the degrees of freedom adjusted R^2, i.e., $\hat{R}^2$, are shown in the last 3 columns.

phase is only a small part of the interaction process itself and during task formulation, task planning and the ballistic phase users usually focus attention on the visual scene. Thus, we can expect that stereoscopy would be beneficial for the overall interaction process, and that it will not be disturbed by the manipulations.

On the practical side, the subjects' inability to discriminate small induced object manipulations is quite interesting since it allows users to interact with objects within shallow depth directly, and without noticeable impact on their performance, provided the accuracy is adjusted according to [VSBH11].

As shown in the next section, users are quite inaccurate in determining an object's depth at the end of the correction phase if tangible feedback is missing. Our experiments provide options to extend the so defined depth vicinity in which the user cannot determine if she first touched the display or the object. In particular, objects with negative or zero parallax, i. e., in front of or on the display surface, could be moved with the user's finger by about 33% of the total finger motion without the user noticing it. For instance, starting the manipulation when the pointing finger is at about $10cm$ in front of the screen for an object at depth $3.3cm$ will allow us to move the object exactly onto the display surface at the moment of touch contact, while this motion will remain imperceptible for most users. Although this comes at the cost of additional hardware equipment, it opens a range of new opportunities. For instance, combined with the approach proposed by Wilson and Benko [WB10], one could use stereoscopic rendering, thus augment non-planar content on the available surfaces or add objects floating in front of or beyond those surfaces. By shifting the virtual objects or application of some morphing technique on more complex surfaces, one could enable touch interaction while still providing passive haptic feedback in such a setup. Using solely one of the scaled shift techniques with manipulations starting $10cm$ before the intended object or point is reached, the available interaction space could be seamlessly extended to a volume between the paralax planes at $\pm 1.3cm$.

Furthermore, the behavior of the PSE might indicate that one *needs to add some object manipulations* in order to enhance the interaction. In particular, since users apparently judge static objects as moving, one may need to move those objects to make them appear more static. As our initial user evaluation revealed, this was in fact the case with our setup (this is described in detail in Chapter 4.). Some of the test persons were initially confused when they were instructed to touch an object which is floating in front of the screen. Nevertheless, after a few test trials all of them adapted to the task and none found it unnatural or inconvenient. Usually more confusing for the users was the forced choice on objects' parallax or motion. Most of the users commented that in many cases they were not able to determine the right answer and pressed a button on gut instinct. Interestingly, those "instinctive" decisions were surprisingly consistent across the participants as the confidence intervals in Figures 3.10 and 3.13(a) confirm. This might be explained with the fact that disparity cues are used for planing a pointing gesture on a very basic cognitive level. Therefore it might be possible to detect such thresholds, that are independent from the cultural or educational background of the user.

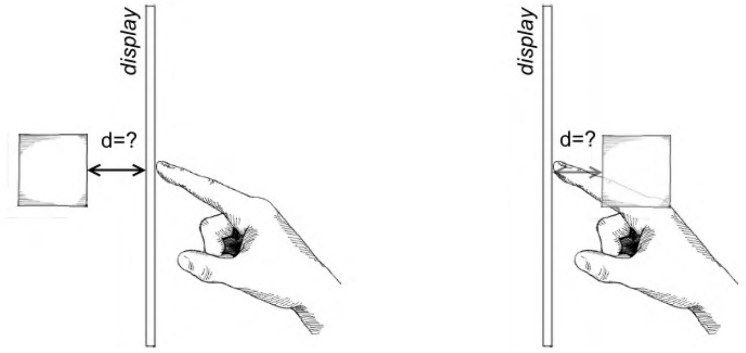

Figure 3.14: Illustration of misalignment between visually perceived and tactually felt contact with a virtual object.

3.6 Discrimination of Stereoscopic Depth

In the last few sections we have investigated the human ability to discriminate induced depth shifts of either a single virtual object or the entire scene and have discussed a range of potential applications which may benefit from such techniques. Here we want to investigate the human ability to discriminate depth misalignment between visually perceived and tactually felt contact with a virtual object, without manipulating its properties in any way (illustrated in Figure 3.14). While this would supposedly reduce the depth of the available interaction volume, it also allows touch-based stereoscopic interaction without any further hardware devices. Moreover, existing 2D interaction techniques might be reused in stereoscopically rendered shallow depth 3D environments without modification, while the user may still benefit from the additional visual cues. One of the interesting results of the experiment reported in Section 2 was that users have quickly adopted to the task of touching stereoscopic objects which were floating in front of or behind the display. Indeed, while a few of them initially found the task description awkward, most of the comments given after the experiments indicated that zero, small positive and small negative parallaxes could not be reliably distinguished when performing a touch gesture. One obvious reason for this is that the display surface has neither visual representation nor associated meaning in a stereoscopic context. On the other hand, it also means that there is some spatial displacement between the point at which the user sees that she is touching the object and the point at which she feels the touch. Interestingly, most of the users found "nothing wrong" with the interaction itself and considered a touch gesture finished when haptic feedback was received, although in some cases they passed with their pointing finger through the visual representation of the object (for objects displayed with negative parallax) or never reached this visual representation (for positive parallax, cf. Figure 3.14). While this might be a remarkable exception of the general rule that vision usually dominates extraretinal cues, one must take into account the fact that the visual cues are in this case contradictory, too. Indeed, when reaching out to touch an object rendered with negative parallax, at some moment the binocular disparity cues are suggesting that the user's finger is behind the object. At the same time the finger is occluding parts of the object, since the display surface is behind it. This ambiguity is further

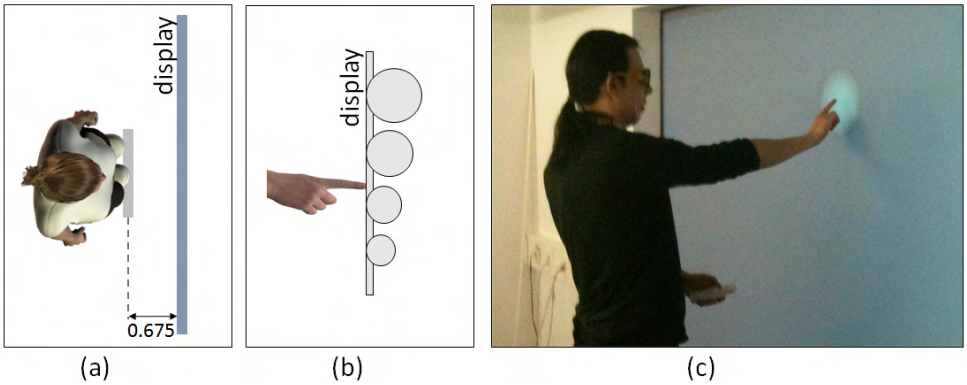

Figure 3.15: Experiment setup for discrimination of stereoscopic depth in large stereoscopic display environments; (a) the experiment setup; (b) illustration of object alignment to a parallax plane (here: the zero parallax plane)(c) participant judged object parallax after a touch was detected.

enhanced by the missing haptic feedback which is expected when touching an object. Thus, while the user's finger has already reached the object and even gone beyond the point of initially expected contact, the user is misinterpreting the visual cues and usually continues the touch until either the discrepancies become too large or until some additional, clearly distinguishable cue (such as the sense of touching something) prevails, solving the ambiguity in either direction. While of particular interest, the question at which depth the discrepancy becomes too large and thus available for our attention is not easily answered. Depending on the object's size, visual appearance, or on-screen position, as well as on the interaction context and the touch-environment settings, e .g., display size, tilt angle or user position, the importance of a single cue might significantly differ.

In this section we report the results of two experiments designed to evaluate the user's ability to discriminate between visually perceived and tactually felt point of touch contact with a virtual object. In the first experiment we have tested the interaction with large stereoscopic walls, while in the second a desktop setup was used. While considerably different in value, the results show similar qualitative behavior and undoubtedly support our initial assumption, that there is a usable depth range for shallow depth interaction. In addition, the estimated values allow us to enlarge the interaction volume defined by the detection thresholds for the scaled shift techniques, since the objects do not need to be perfectly aligned with the display surface at the moment of touch.

3.6.1 Experiment: Discrimination of Stereoscopic Depth in Large Stereoscopic Display Environments

The aim of this experiment was to analyze how sensitive subjects are to a small discrepancy between visual and haptic depth cues while performing touch gestures on a large stereoscopic display. Therefore we evaluated subjects' ability to determine the exact point of contact with an object projected with different stereoscopic parallaxes on our multi-touch wall. The

experiment was performed using the same hardware setup as in the experiment described in Section 3.4.1. For this experiment we have formulated the following initial hypotheses:

- ${H_0^{(1)}}_{wall}$: The size of the object does not affect the perceptual detection thresholds.
- ${H_0^{(2)}}_{wall}$: The detection thresholds are smaller for objects rendered stereoscopically with positive parallax than those for objects rendered with negative parallax.

Participants

18 of the 19 subjects who participated in the experiment described in Section 3.4.1 also participated in this experiment. The total time per subject including pre-questionnaire, instructions, training, experiment, breaks, and debriefing took 30 minutes. Subjects were allowed to take breaks at any time.

Material and Methods

We presented a written task description and experiment walk-through via slides on the projection wall. A gray sphere projected stereoscopically on the touch-surface (cf. Figure 3.15 (c)) was used as visual stimulus. Subjects were positioned at arm-reach distance from the projection wall and were instructed to perform touch gestures with their dominant hand, while remaining in place (cf. Figure 3.15 (a)). The subjects' task was to touch the virtual sphere projected on the multi-touch wall and to judge in a 2AFC task if they first touched the projection wall or penetrated the sphere's surface while performing the touch gesture. After subjects judged the perceived stereoscopic depth by pressing the corresponding button on a Wii controller, we displayed a blank screen for $200ms$ as short interstimulus interval. As experimental conditions we varied the position of the sphere, so that the point of the sphere's surface closest to the subject was displayed stereoscopically behind the interaction surface, in front of it or exactly on it, as illustrated in Figure 3.15 (b). We have tested 5 positions (sphere's surface displayed -20cm and -10cm behind the projection wall, +20cm and +10cm in front, and 0cm on the projection wall). Additionally, we varied the sphere's size using a radius of 10cm, 8cm, 6cm or 4cm. The sphere's position and size were not related from one trial to the next, but presented randomly and uniformly distributed. Each subject tested each of the pairs of position and size 5 times, resulting in a total of 100 trials. Before the test trials started we presented 10 randomly chosen test trials to the subjects to provide training and ensure that they understood the task.

Results

We found no significant difference between results for the different sizes of the spheres so we pooled these responses. Figure 3.16 plots the mean probability for a subject's judgment of having touched the projection wall first ('up' button) against the tested distance between the sphere's surface and the projection plane. The x-axis shows the distance between the sphere's surface and the projection plane, the y-axis shows the probability for 'up' responses on the

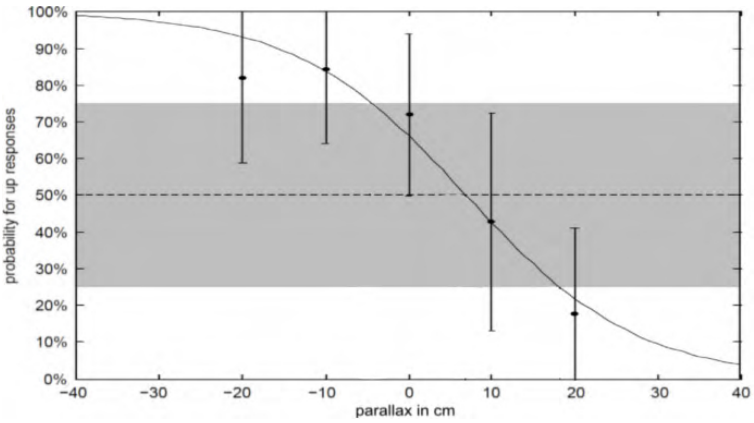

Figure 3.16: Experiment results for discrimination of stereoscopic depth in large stereoscopic display environments. The solid line represent the fitted psychological function for the discrimination task. The x-axis shows the tested parallax, the y-axis shows the mean responses. The vertical bars show the standard error.

Wii remote controller, i .e., the judgment of having touched the projection wall first and not the sphere. The solid line shows the fitted psychometric function of the form $f(x) = \frac{1}{1+e^{a \cdot x+b}}$ with real numbers a and b. The vertical bars show the standard error. From the psychometric function we determined a slight bias for the $PSE = 6.92cm$. Detection thresholds of 75% were reached at distances of $-4.5cm$ for 'up' responses and at $+18.5cm$ for 'down' responses, although the standard error is quite high in this experiment.

3.6.2 Experiment: Discrimination of Stereoscopic Depth in Desktop Environments

As in the experiment described in the previous section, the aim of this experiment was to analyze how sensitive subjects are to a slight discrepancy of visual and haptic depth cues while performing touch gestures. Nevertheless, this experiment was conducted for a desktop setup with considerably smaller objects, such that the on-screen surface of the object is comparable to the projected surface of the user's finger. Furthermore, since the display size is considerably smaller than in the previous experiment, environmental stimuli might have significant impact on the user's estimations. The experiment was performed using the same hardware setup as in the experiment described in Section 3.5.3. All subjects who participated in the experiment described in Section 3.5.3 also participated in this experiment. The total time per subject including pre-questionnaire, instructions, training, experiment, breaks, and debriefing took about 30 minutes. Subjects could take breaks at any time. In addition, after each 50 trials subjects had to take a 2 minutes break in order to minimize errors due to exhaustion or poor concentration. As in the previous experiment we hypothesize that the detection thresholds for positive parallax will be smaller than those for negative parallax, but also expect some correlation with the object size. Therefore for this experiment the hypotheses are formulated as follows:

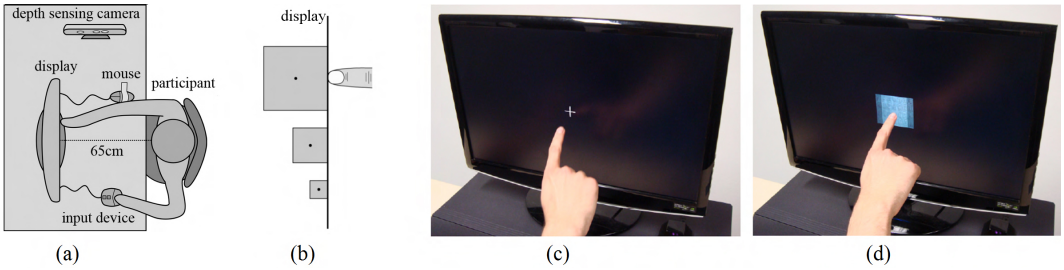

Figure 3.17: Experiment setup for discrimination of stereoscopic depth in desktop environments; (a) the experiment setup; (b) illustration of object alignment to a parallax plane (here: the zero parallax plane); (c) participant's hand motion, a white cross in the middle of the screen indicates the point to be touched; (d) participant judged object parallax after a touch was detected.

- $H_0^{(1)}{}_{desktop}$: Smaller objects, with size comparable to the finger width, will have larger detection threshold intervals compared to objects considerably larger than the user's finger.

- $H_0^{(2)}{}_{desktop}$: The detection thresholds are smaller for objects rendered stereoscopically with positive parallax, than those for objects rendered with negative parallax.

Materials and Methods

In a *within-subject designed* experiment, participants had to decide in a *2AFC* task whether objects appeared in front of the screen surface or behind it after performing a touch gesture.

At the beginning of the experiment each subject completed a simple stereoscopy test with our apparatus. Subjects were seated such that the center of the screen was at their eye level at a distance of approximately $65cm$ as illustrated in Figure 3.17 (a). They were instructed to remain in this position during the entire experiment and to perform all trials with their dominant hand.

Subjects indicated the start of each trial by pressing the left mouse button, which guaranteed a consistent start position of the touch gesture throughout the experiment. After a trial was started, a white cross in the middle of the screen was displayed to indicate the position the user has to touch. Once touch on the cross was detected, an object was displayed for one second with a vertical offset of $1cm$ to avoid complete occlusion by the finger (cf. Figure 3.17 (c) and (d)). Then a written description, displayed for at least $200ms$, forced the subject to decide with the Speedpad whether the object's surface appeared to be in front of the display surface or behind it. After all trials were finished, subjects were shortly interviewed to provide their subjective impressions or remarks.

As visual stimuli we used a textured box in order to provide sufficient depth perception and stereoscopic disparity cues. The box was displayed with a light gray texture having a random, non-repetitive pattern on black background in order to reduce cross-talk artifacts. To test the effect of object size, boxes with edge lengths of $2cm$, $3.5cm$ and $5cm$ were displayed. We varied the distance between the display's and object's surfaces in the interval between $-10cm$

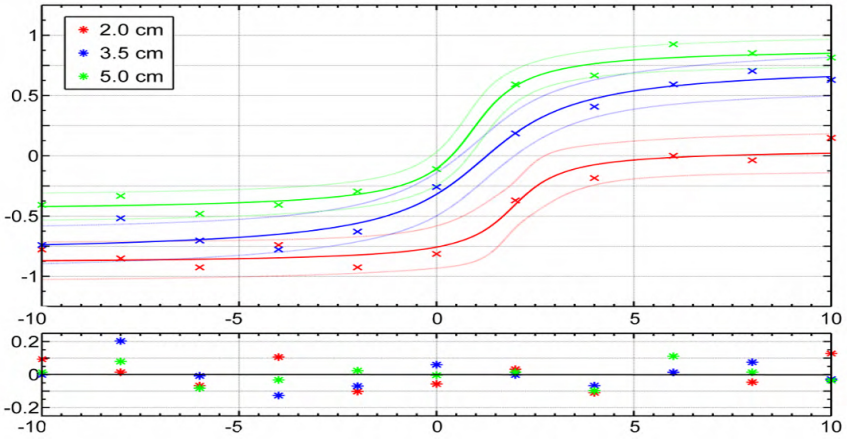

Figure 3.18: Experimental results for discrimination of stereoscopic depth in desktop environments; X-axis shows the depth of the object's surface in cm, Y-axis shows the mean responses for different object sizes; (top) solid curves show fitted psychological sigmoid functions, the light curves show the 95% confidence intervals and crosses show the means of subject responses; (bottom) the fitting residuals

and $10cm$ with steps of $2cm$, resulting in a total of 11 parallax planes. Since we have used a *right-handed* coordinate system, *negative* depth values represent *positive parallax* and vice versa. Box positions were adjusted in depth such that the foremost surface was aligned to the specific parallax, i.e., positioning of the viewable surface instead of the center, as illustrated in Figure 3.17 (b). For this experiment the *method of constant stimuli* was used, i.e., the sizes and positions of the objects were not related from one trial to another, but presented randomly and uniformly distributed. Each position-size pair was presented exactly three times, resulting in a total of 99 trials per subject. Additional 5 training trials with strong parallaxes ($\pm 10cm$ and $\pm 15cm$) were added at the beginning of the experiment in order to ensure that participants understood the task and received some initial training.

Results

The data of one subject was excluded from evaluation since post interrogation with the subject revealed that he had misunderstood the task, evaluating the object's size rather then parallax.

The subject mean responses were evaluated with 3×11 repeated measurement ANOVA with Greenhouse-Geisser (G.-G.) correction, testing the within-subject effect of object size

Size $[cm]$	$DT_{0.5}$ behind, $[cm]$	$DT_{0.25}$ behind, $[cm]$	PSE $[cm]$	$DT_{0.25}$ in front of, $[cm]$	$DT_{0.5}$ in front of, $[cm]$
2.0	1.69	2.54	7.02	–	–
3.5	−1.27	0.33	1.28	2.29	4.35
5.0	–	−0.94	0.38	0.99	1.67

Table 3.7: Detection Thresholds $DT_{\pm 0.25}$ and $DT_{\pm 0.5}$, and Points of Subjective Equality (PSE) for tactile and visual touch discrimination task.

and parallax. We found a significant within-subject effect for object size (G.-G. $F^{1.32}_{22.48} = 17.09$, $p < 0.01$) as well as for parallax (G.-G. $F^{2.76}_{47.01} = 30.49$, $p < 0.01$) and therefore split the results for different object sizes and parallaxes. The relation between mean subject estimations and parallax were evaluated for different object sizes with a parametric non-linear regression analysis, fitting psychological curves of the type $f(z) = a \cdot \arctan(b \cdot z + c) + d$ with real a, b, c and d with the non-linear Levenberg-Marquard least square iteration algorithm.

Figure 3.18 presents the fitted curves (top) and the residuals (bottom). The X-axis shows the start parallaxes, i.e., the distance between the object's front surface and the display surface, with positive values indicating objects in front of the screen, i .e., with negative parallax, and negative values indicating objects behind the screen, i .e., with positive parallax. The Y-axis shows the mean responses, where the judgment that an object appears behind the screen surface is valued as -1, and the judgment that an object appears in front of the screen surface is valued as 1. The solid curves show fitted sigmoid functions for object sizes $2cm$ (red), $3.5cm$ (blue) and $5cm$ (green), and the dotted curves show the 95% confidence intervals. The crosses represent the means of subject responses, while the stars represent the fitting residuals. The standard error of the regression ($RMSE$) was 0.111 for objects with edge length of $2.0cm$ with summed square of residuals $SSE = 0.085$ and degrees of freedom adjusted R^2, i.e., $\hat{R}^2 = 0.929$. For object size $3.5cm$ the $RMSE$ was 0.108 with $SSE = 0.083$ and $\hat{R}^2 = 0.968$, and for object size $5.0cm$ the $RMSE$ was 0.079 with $SSE = 0.045$ and $\hat{R}^2 = 0.981$. These values and the random residuals show that the model functions are good fits of the data and explain more than 90% of the variance. The estimated points of subjective equality (PSE), and the $DT_{\pm 0.5}$ and $DT_{\pm 0.25}$ for positive and negative parallax are summarized in Table 3.7. The detection thresholds in Table 3.7 show that subjects were accurate for parallaxes outside the interval $-1.27cm$ and $4.35cm$ for objects of size $3.5cm$. Participants detected positive parallaxes for $2cm$ objects below $2.54cm$ with a probability of at least 75%. In contrast they detected negative parallaxes for $5cm$ objects starting from $0.99cm$ with a probability of at least 75%. Participants were not able to accurately detect negative parallaxes for $2cm$ large objects and positive parallaxes for $5cm$ large objects. The PSE for $2cm$ large objects shows a strong offset of $7.02cm$, while the PSE for $3.5cm$ large objects shows a smaller offset of 1.28cm. Objects with $5cm$ size have a PSE of 0.38cm.

3.6.3 Discussion

Consistent with Chan at al. [CKC+10] the first experiment provides strong evidence for the fact that if tangible feedback is missing, humans are quite inaccurate in determining an object's depth at the end of the correction phase of a touch gesture. Interestingly, the effect of the object's size was different in both experiments. While no significant effect was found for the large display setup, the object's size had a strong correlation with the users ability to discriminate depth misalignments for the desktop setup. In particular, subjects had problems in detecting positive parallaxes for large objects and negative parallaxes for small objects. The curves in Figure 3.18 show convergence below 75% detection in these cases. For

instance, small objects ($2cm$) on the zero parallax plane were perceived behind the screen, but the same objects having negative parallax could not be discriminated clearly by subjects ($PSE = 5.9cm$). In contrast, subjects' detection for larger objects ($5cm$) was quite accurate for objects having negative or zero parallax, with PSE nearly zero ($PSE = 0.29cm$), while for objects behind the screen the subjects were somewhat uncertain.

One can explain the difference in the results of the two experiments with occlusion of objects by the finger. Objects with size of $2cm$ were in large part occluded by the finger such that depth impression was disturbed on negative parallaxes. This led to more uncertainty in discrimination for these objects, even though we applied small vertical offsets for objects displayed under the finger to reduce this effect. With increased object size stereoscopic perception had more surface to rely on, providing a more accurate detection even if a subject's finger occluded a part of the surface.

On the practical side, the subjects' inability to discriminate depth misaligments motivates the possibility to design 3D interfaces without modifications of existing interaction techniques or the hardware. For instance, an interface designer could place some small widgets or anchors in front of the screen in order to improve their visibility and accessibility. Users could then directly interact with those widgets without noticeable impact on their performance, provided the accuracy is adjusted according to [VSBH11]. The detection thresholds define the narrow depth vicinity in which such objects could be scattered, i. e., if object size is under $3.5cm$, they may be placed at most at $4.35cm$ in front of the screen surface, but only at $1.3cm$ behind it. As discussed previously, this depth vicinity could be further extended by application of imperceptible object shifts.

3.7 Conclusion

In this chapter we have investigated the benefits and the limitations of using the "perceptual illusions" design paradigm to enable touch interaction with stereoscopically rendered objects. Therefore, we have first generalized our empiric observations on how users interact with stereoscopic visualizations and formulated three typical interaction states: (1) the *observation* state, where the goal of the intended interaction is formed and different strategies to achieve this goal are formulated; (2) the *specification* state, where the objects or tasks to be performed have already been selected, but the corresponding actions have not yet been performed and (3) the *execution* state, where the user performs the actions planned in the specification state. This general concept and our initial considerations about the possible types of manipulations, which can be imperceptibly applied to an object or to the entire scene, provided the groundwork for a series of psychological experiments, where a set of manipulations and their magnitudes were estimated. Our results show that there is a usable range of manipulations which could be applied to either an object or to the entire virtual scene, without the user noticing this. For instance, in a large virtual display environment, e .g., a CAVE or PowerWall, one could use the moment in which the user is approaching the display surface to imperceptibly shift the scene in the same direction and then could

further manipulate the position of the intended object, with either the original *scaled shift* technique or with its generalized variation. The results reported in the last section allow us to further relax the requirements for the manipulation, i .e., the virtual object does not need to be exactly aligned with the display surface, since small displacements between the visually perceived object depth and the moment in which the user receives haptic feedback could not be reliably detected.

The reported results provide the basis for natural interaction with stereoscopic objects. The user neither needs to switch between 2D and 3D interaction metaphors, nor needs to use cumbersome instrumentation in order to interact and could simply touch a virtual object floating in front of or behind the display surface. The system will imperceptibly manipulate some parameters of the visualization, which redirects the users finger to the display surface, where tactile feedback is provided. Nevertheless, this enhanced user experience comes at the cost of the available depth, where the objects might be placed, since even strong manipulations allow to shift the object by maximal $15cm$. However, there are many applications where the notion of *shallow depth* 3D is sufficient, but may still benefit from the enhanced visual perception provided by the stereoscopic visualizations.

While promising, the results reported in this Chapter are still first steps toward enabling ”real world” touch interfaces based on perceptual illusions. For instance, it is currently not clear how the proximity to the display’s bezel, the viewing angle and the speed of the touch gesture will affect the detection thresholds, and further research is needed to address these questions. Furthermore, one of the main drawbacks of the proposed techniques – the fact that the intended object has to be known in advance – is often a non-trivial task. A method to relax this requirement is discussed in the next chapter.

4 Chapter 4 Object Attracting Shift

Always think of what is useful and not what is beautiful. Beauty will come on its own accord.

(Nikolai Gogol)

4.1 Designing Interfaces with Scaled Shifts

While the results presented in the previous chapter motivate the applicability of the generalized scaled shift technique, there are a number of additional problems which arise when designing a practical interaction metaphor based on this technique. For instance, if objects are close to each other or overlapping, moving only one of them will be instantly noticed by the user. Especially in cases of textured objects in dense visualizations, moving one of them will gradually cover a larger or smaller part of the texture pattern on the other object, which may easily be detected. In addition, with multiple layered objects application of the scaled shift technique may lead to objects overlapping in depth or even switching their z-order. These and similar problems are inherent to all interfaces, in which a 3D interaction space is compressed to a 2D surface. Nevertheless, many investigations (most prominently [OS05, TS07]) have shown that 2D interaction with the top-most or with the closest visual surface in a 3D scene usually outperforms pure 3D interaction, which is the basis of the "magic" 3D interaction paradigm [TS07]. Furthermore, in a head-tracked setup the user will most probably adjust her point of view and reduce the touch gesture to a 2D pointing task instead of touching a fully occluded object. The "magic" 3D interaction paradigm becomes even more important in the context of shallow depth visualizations.

A more significant problem with the interaction technique used in our experiment is the fact that the intended object should be known exactly prior to the application of the technique. Depending on the application this is often a non-trivial precondition. Relaxing the condition that the object be known exactly to an approximation of some volume in which it may reside simplifies the problem significantly. In this case, one could use a heuristic (e. g. [JWBF01])

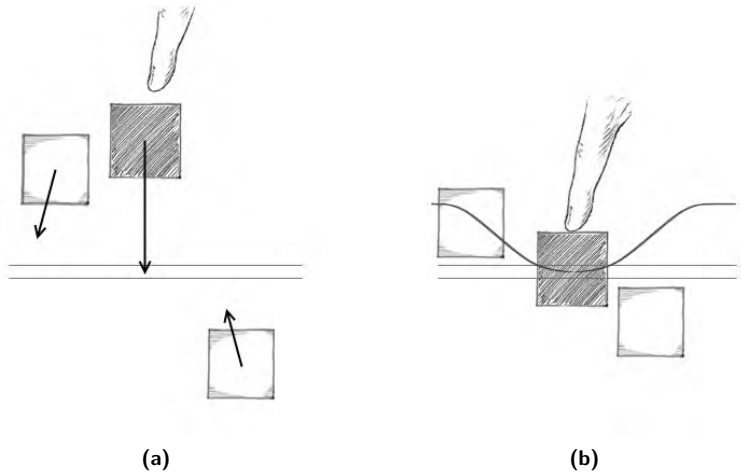

Figure 4.1: Illustration of the attracting shift technique; (a) objects close to the intended one are also moved, but with reduced shift factors; (b) at the end of the touch gesture the intended object is aligned with the zero parallax plane. The curve illustrate the distribution of relative shifts to nearby objects.

based on the last few positions of the finger and the current head orientation in order to determine safely and sufficiently early a volume in which the intended object resides [NS03, SD12].

In this chapter we present the *Object Attracting Shift Technique*, which is designed to alleviate these and similar problems. The conducted preliminary user evaluation has approved the applicability of the technique in more elaborate application scenarios, and it has confirmed our initial intuition that the application has to slightly shift virtual objects in order to make them appear perceptually static.

4.2 Object Attracting Shift Technique

Combining the results of the experiments reported in the previous chapter with the above considerations we have developed a new technique, which we call *attracting shift*. The core idea of the attracting shift technique, illustrated in Figure 4.1, is that a *generalized scaled shift* with maximal shift factor σ_{max} is applied to the object, which the user is intending to touch, and to all objects around this one manipulations with decreasing shift factors are applied. Thus figuratively the intended object (the *attractor*) to which the strongest shift factor is applied "attracts" all objects around itself.

With $P_0 \in \mathbb{R}^3, P_0 = (x_0\ y_0\ z_0)^T$ denoting the position of the attractor and $P \in \mathbb{R}^3, P = (x\ y\ z)^T$ denoting the position of an arbitrary object around P_0 we can then express the shift factor σ as

$$\sigma = \sigma_{max}(z) \cdot \alpha(r_{xy})$$

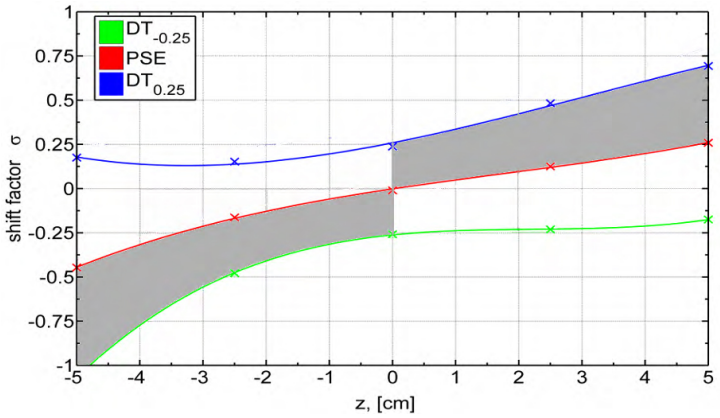

Figure 4.2: Illustration of the available shift factor space $\sigma_{max}(z)$. As in Figure 3.13 the solid lines show the fitted polynomial functions for $DT_{\pm 0.25}(z)$ and $PSE(z)$ for the generalized scaled shift technique. The shaded area represents the available shift factors, which could be applied to imperceptibly manipulate an object.

with $r_{xy} = \sqrt{(x-x_0)^2 + (y-y_0)^2}$ denoting the projected distance between P_0 and P. In the above equation $\sigma_{max}(d)$ denotes the maximal shift factor, which could be imperceptibly applied to an object starting at distance d from the display surface. The function $\alpha(r_{xy})$ determines the amount of maximal shift to be applied to an object depending on its distance from the attractor, i. e., the intended object. Reflecting on the form of $\alpha(r_{xy})$ one can see that it has to be (strictly) decreasing, with maximum of 1 at $r_{xy} = 0$. For our purpose, we consider the Gaussian curve as a reasonable initial choice. Thus we have

$$\sigma = \sigma_{max} \cdot e^{-1.39 \frac{r_{xy}^2}{R^2}}$$

with cut radius $R \in \mathbb{R}$ denoting the radius at which the strength of the applied shift factors falls below 25%.

As discussed in Chapter 3, the maximal shift factor, which can be imperceptibly applied to an object starting at distance z from the display surface, is defined by the detection thresholds $DT_{\pm 0.25}(z)$ (cf. Figure 3.13(b)). For instance, for objects starting $2cm$ in front of the surface, i. e., with negative parallax, one could apply shift factors between -0.23 and 0.42 without the user noticing the motion. Since we are currently only interested in moving the objects closer to the display surface, we only consider positive shift factors for objects with negative parallax and negative shift factors for objects with positive parallax. This new shift factor space is illustrated by the shaded area in Figure 4.2. Objects aligned with the display surface should not be manipulated. Thus the maximal applicable shift factor σ_{max} could be defined as:

$$\sigma_{max}(z) = \begin{cases} DT_{0.25}(z) & z > 0 \\ 0 & z = 0 \\ DT_{-0.25}(z) & z < 0 \end{cases}$$

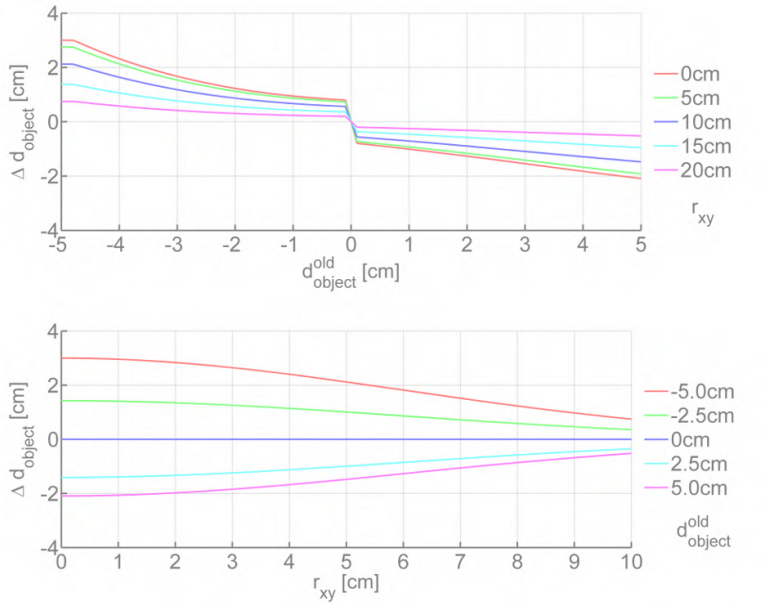

Figure 4.3: Illustration of the absolute object motion for finger motion $\Delta d_{finger} = 3cm$ as function of object's start parallax (top) or distance to the attractor (bottom).

Considering this definition of $\sigma_{max}(z)$ and the detection thresholds $DT_{\pm 0.25}$ reported in the previous chapter one can see that unscaled application of σ_{max} allows shifting an object to the opposite side of the display and that even the partial application of σ_{max}-shifts may align an object to the display surface. Thus the precise knowledge of the intended object is not needed. The system only has to determine and manipulate *some* object near to the intended one. The nearby objects will then be attracted by this one, albeit with smaller shift factors, and will probably end up aligned with the display or very close to it, too. As shown in Figure 4.2, the detection thresholds are decreasing when objects get closer to the display surface. Thus manipulating an object farther alway from the display surface will push all near objects closer to the surface. Recalling from Section 3.5.4 that $\Delta d_{object} = \sigma \cdot \Delta d_{finger}$, we could then express the new depth of an arbitrary object as function of its depth, distance to the attractor and finger motion. An illustration of the object displacement for fixed $\Delta d_{finger} = 3cm$ and varying d_{object} or r_{xy} is shown in Figure 4.3. As one can see in these figures, objects never switch their z-order, and the absolute object motions are sufficient to bring them to the zero parallax plane (and beyond, if desired).

4.3 Preliminary User Evaluation

In this study we compared the *object attracting shift* technique for different values of the cut radius R with a static condition in which no objects were moved.

In addition, as discussed previously, our results show that it might be beneficial to apply small object shifts according to the PSE values estimated earlier to aid users' perception. Indeed, as shown in Figure 4.2, subjects judged static objects with positive parallax ($z < 0$)

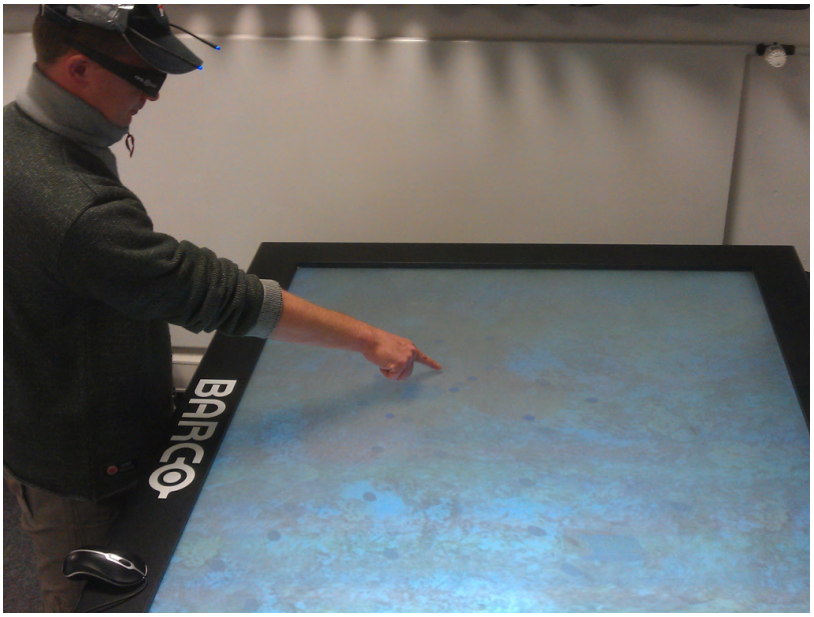

Figure 4.4: Participant performing a touch gesture during the preliminary evaluation of the attracting shift technique.

as moving against their finger, while static objects with negative parallax ($z > 0$) were judged as moving with their finger. In order to evaluate if we could compensate for this estimation, we tested a modified version of the object attracting shift technique, which uses the values of $PSE\,(z)$ instead of $\sigma_{max}\,(z)$, i. e., $\sigma = PSE\,(z) \cdot \alpha\,(r_{xy})$.

4.3.1 Participants

11 male and 2 female subjects (age $22 - 37$, $\varnothing$: 26.9, σ: 4.05) participated in the experiment. All were students or professionals from our department and most of them reported to have experience with stereoscopic content, mostly due to 3D cinema. All subjects were naïve to the experimental conditions. 12 subjects were right-handed and one was left-handed. All subjects had normal or corrected to normal vision.

4.3.2 Materials and Methods

For this study a $120Hz$ frame sequential stereoscopic visualization on a $50''$ tabletop display with resolution 1024×768 and DLP-Link shutter glasses were used (s. Figure 4.4). Subject's fingertip was tracked with our in-house tracker, based on the Kinect depth camera, and head-tracking with a $8 \times$ PPT tracking system was enabled.

We divided the study into two blocks. In the first block the original version of the attracting shift technique with different values of the cut radius R was evaluated against a static condition, in which no manipulations were applied. Subjects started each trial by pressing the left button of a fixed mouse. Then subjects had to sequentially touch 3 randomly highlighted scene objects with different parallaxes in two different conditions. In one of these conditions

the attracting shift technique with randomly chosen R from the pool $0, 5, 10, 15, 20cm$ was applied to the objects, while in the other one the objects remained static. Afterwards written instruction on the screen prompted the user to decide in which condition the objects appeared to move. The order in which the conditions were presented were randomly distributed and counterbalanced to avoid ordering effects. Each R was tested exactly 25 times in randomized order resulting in a total of 150 trials per subject; additional 10 training trials were added at the beginning of this block.

Since the applied shift factors were always within the detection threshold ranges defined previously, we explicitly instructed the participants to pay more attention to the path and speed of the touch gesture than to the object motions. Thus, when performing a touch gesture the subject had to evaluate if her finger reached the object slightly earlier or later than expected, which was then estimated as object motion, or if the path and speed of the gesture were exactly as expected. Our initial evaluation showed that this formulation of the task considerably increases the hit ratios of the participants, compared to the pure evaluation whether an object moved or not, where the hit ratios were 0.50 at mean.

In the second block of the user evaluation we applied the modified attracting shift technique with a fixed $R = 10cm$. The same procedure as in the first block was used to test the effect of PSE compensation. In this block 50 trials were performed by each participant.

As visual stimuli we used an artificial geo-spatial scenario. On a map model of our region rendered with stereoscopic depth-map we placed multiple 3D models (e. g., buildings, cars, planes) at different altitudes, which the participants had to touch within the experiments.

4.3.3 Results and Discussion

For the first block of the experiment, we evaluated the participants' mean hit rates, i. e., the relative fraction of correct answers, with a one-way ANOVA, testing the between group effect of cut-radius R. The effect of R was found to be significant with $F_{40}^{4} = 7.1, p < 0.01$ (Levene test with $p = 0.62$). Post-hoc analysis with the Tukey test showed that the mean hit rate for $R = 5cm$ ($M = 0.49$, $SD = 0.17$) was significantly lower than the hit rate for $R = 0cm$ ($M = 0.60$, $SD = 0.16$) with $p < 0.05$ as well as the hit rates for $R = 10cm$ ($M = 0.65$, $SD = 0.16$) with $p < 0.01$, for $R = 15cm$ ($M = 0.61$, $SD = 0.15$) with $p < 0.01$ and for $R = 20cm$ ($M = 0.58$, $SD = 0.17$) with $p < 0.05$. We did not find significant differences between the other values of R. The results confirm our initial design considerations that a shift application to a single object will be easier to detect in real world scenarios, especially in densely populated 3D scenes. Contrary to our initial expectation the mean hit rates rise again for cut radii greater than $5cm$. This makes the value $R = 5cm$, in which the mean participant's hit rate was 0.49, the perfect choice for the attracting shift technique. Nevertheless, it is currently not clear how this value depends on the scene or the setup's parameters.

The mean hit rate over all participants in the second block was $M = 0.449$ with $SD = 0.087$. In the second block a students T-test of the participants' mean hit rates against a constant

value of 0.5 showed that the mean hit rate ($M = 0.448$, $SD = 0.09$) is significantly lower than 0.5 with $T_{12} = 2.36, p = 0.032 < 0.05$. The result confirmed our initial intuition that application of compensating shift factors would aid the users' perception. In particular, the mean hit rate under 0.5 indicates that the users have perceived the manipulated scene as more static than the not-manipulated one. While the difference of only about 0.05, i. e., 5%, might seem small, it may be an indication of considerable differences in planning and execution of a touch gesture in real and virtual environments, and it may increase under different conditions (e. g., touch speed, object size).

None of the subjects reported to have problems with the stereoscopic impression, and most of them described the presented content as "realistic". Upon request, after all trials were completed, most of the subjects were able to determine the exact depth of an object and some were surprised, how the manipulation redirected their finger to the surface. Though we explained that we slightly move the objects during the touch gesture, some of the participants were confused, since they never "saw" objects changing their depth position. To summarize, our evaluation approved the applicability of the attracting shift technique for real world, shallow depth scenarios, providing a user with the ability to directly touch stereoscopically rendered objects without loosing the advantages of common 2D touch interfaces.

4.4 Limitations and Design Implications

In the previous chapter we approved that application of certain undetectable object manipulations aid user's perception while reaching for virtual objects and may redirect her finger to the display surface, where tactile feedback is provided. With the attracting-shift technique we have relaxed the inherent pre-condition that the intended object should be known in advance, which makes the technique far more practical. In contrast to alternative techniques that may be used for interaction with stereoscopic content, which either lack haptic feedback (e. g., free-hand, in-air interaction) or require additional instrumentation (e .g., tangible views), the proposed approach brings together direct-touch interaction with haptic feedback at the sacrifice of depth.

Talking about interaction with stereoscopic content one is tempted to fit the approach in the 3D user interface domain; however, an interface designer should not forget that the presented technique per se is inherently two-dimensional. Thus there is no way for the system to determine at the end of the touch which object was intended, if multiple objects were displayed on top of each other. Nevertheless, as discussed earlier, "magic 3D" and "shallow depth" are two powerful notions, suitable in many application scenarios.

Considering Figure 4.2 again, one can see that a vertical line segment between $DT_{+0.25}$ and $DT_{-0.25}$ defines the interval of possible shift factors, which may be applied to an object with given parallax. While for the attracting shift technique we only consider shifts which move the objects closer to the surface (positive for objects with negative parallax and vice versa), one may consider moving the objects in either direction. A free-hand interaction technique may – for example – manipulate objects with negative parallax to move closer to

the hand while the user is grasping for them and thus reduce the overall magnitude of the hand motion. Alternatively a "smart" technique could selectively move an object closer or farther away from the user's finger or from the interactive surface, depending on its accessibility or appropriateness for the current task. Overall, there is a range of possible applications for this and similar kinds of imperceptible object motions, which might be usable for interaction design.

On the practical side, the formulation of the scaled shift technique as a function of only the depth of the subject's finger and the extent of the $DT_{\pm 0.25}$ ranges for different parallaxes make it possible to reduce the required precision of the finger tracker. Indeed, since the $DT_{\pm 0.25}$ ranges allow shifting an object to the opposite side of the display, one can use slightly smaller values as σ_{max} for manipulation, such that incorrectly large values of Δd_{finger} are still imperceptible. The manipulation of the object should then be stopped, when its depth reaches zero. In contrast to the precision, the tracker's speed should be as high as possible. In particular, since the typical duration of a touch gesture is between 0.5 and 1.5 seconds, and the technique is applied at the very end of this gesture, one has to provide at least 60fps for reasonably smooth object motions. Otherwise, the motion of the object will be either jerky, which could then be easily detected, or no shift could be applied at all.

5 Chapter 5

Multi-Touch supported Navigation in Virtual Environments

In wisdom gathered over time I have found that every experience is a form of exploration.

(Ansel Adams)

5.1 Interaction with Stereoscopic Objects beyond the "Shallow" Depth

In the previous chapter we have evaluated the user's ability to discriminate some small induced object shifts, and proposed a practical interaction metaphor based on our results. The main objective was to enable direct touch interaction with haptic feedback by sacrificing interaction volume. Nevertheless, in many cases the available interaction volume is defined by the specific requirements of the application and cannot (or should not) be modified. Yet, it has to be available for natural interaction. An example of such application scenarios are the egocentric full-sized immersive virtual environments (IVEs). For instance, an architect might want to walk along the streets of a newly designed city complex, a customer might want to get a better impression of a house she is willing to buy, or an engineer might want to inspect some mechanical part in real size.

The main point in the *move the surface* concept is that one can decouple the interactive surface from the display and move it freely in the 3D volume above the display. One possibility to realize this is to use a multi-touch enabled transparent prop [SES99, VSBH10a], which can be aligned with a floating object and used as input to interact with this object in place. Thus the user interacts "directly" with the object through the prop and receives haptic feedback. Nevertheless, since the objects aligned with the prop are projected with very large disparity, the users often have considerable problems to maintain the fusion of the images for the left and the right eye. This is further impaired by even very small scratches on the prop's surface,

which may distract the eye accommodation from the display surface of the prop. Another recently published alternative in this context is to use an opaque prop and a top projection exactly on the surface of this prop, i. e., to use tangible views [STSD10]. Nevertheless, to the best of our knowledge the "tangible views" have not yet been considered with stereoscopic projections, which will undoubtedly unveil a set of new challenges for the application of these devices.

With the "move the touches" paradigm the touches are moved into the 3D space above or below the display surface by using the on-surface 2D positions of multiple touch points to calculate a 3D position of a distant "cursor" [BF07, SVH11]. As with the touch precision (cf. Chapter 2) the approach tries to shift the problem into the interface design space by defining the *stereo touch* as distinct input modality. Examples of interface techniques based on this approach are the *balloon selection* metaphor [BF07], the *triangle cursor* [SVH11], the *fishnet* metaphor [DFK12] and many more [CDH11, HBCdlR11, SGH+12]. While with these interfaces 2D interaction is either not supported or has to be realized with a different set of techniques (which leads to frequent switching between different interaction modes), they usually provide suitable solutions for manipulation tasks in the 3D volume above a tabletop surface. However, until now challenges and limitations of multi-touch interaction in the context of 3D *navigation* have rarely been considered. Recent developments in the area of interactive surfaces enable the construction of low-cost multi-touch displays and relatively inexpensive sensor technology to detect foot gestures, which allows to explore these input modalities for interactive 3D environments.

In this chapter we demonstrate how multi-touch hand gestures in combination with foot gestures can be used to perform navigation tasks in (semi-) immersive virtual environments in the context of Geographic Information Systems (GIS), which are well suited as a complex testbed for evaluation of user interfaces based on multi-modal input. Therefore, we have developed a navigation device based on the Nintendo Balance Board and a transparent FTIR-based [Han05] multi-touch surface for navigation in 3D geospatial data from an egocentric perspective, as well as a *Worlds In Miniature (WIM)* [SCP95] approach for wayfinding. As a proof-of-concept we simulate a Human-Transport vehicle with a physically inspired steering technique (illustrated in Figure 5.1(b)). In addition, multi-touch input is used to manipulate the WIM. Our preliminary user evaluation has verified that the combination of multi-touch hand and foot gestures finds acceptance and is beneficial for the user, thus it has the potential to enable more natural and powerful interfaces for traveling and navigation in virtual environments.

5.2 Navigation in Virtual Environments

Navigation is one of the most basic and common interaction tasks in virtual environments (VEs). Many applications require intuitive metaphors and techniques to explore the data displayed in a certain domain. For example, 3D geospatial data has grown in popularity and has been used widely in many different application domains in recent years. While

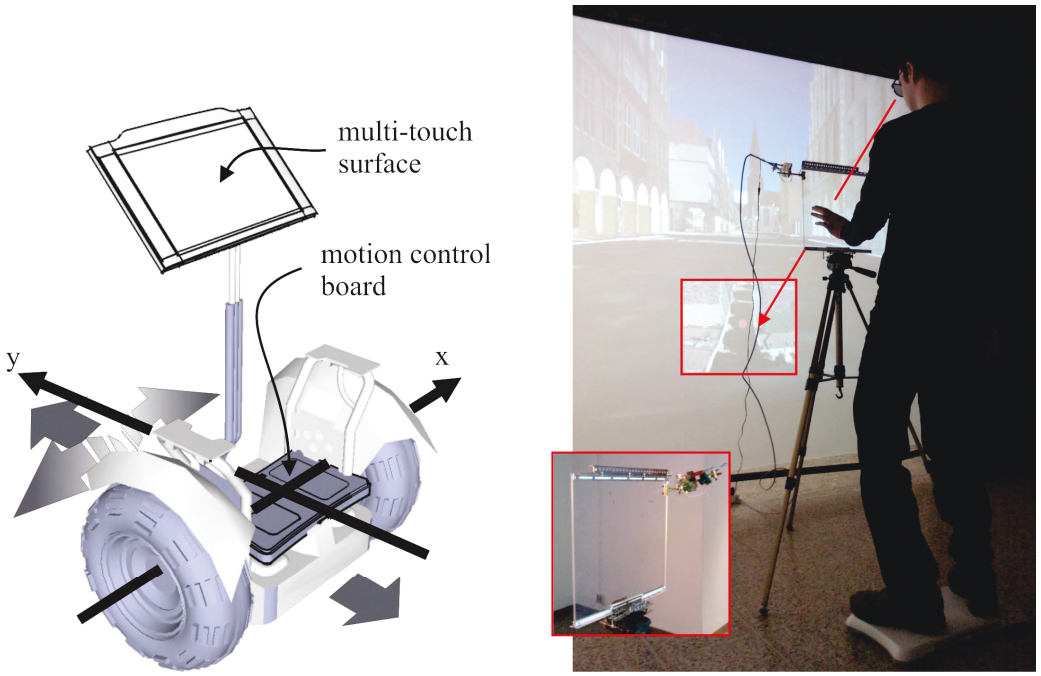

Figure 5.1: The multi-touch enabled human-transporter metaphor. (a) Illustration of the metaphor. The user can travel by shifting her weight. The multi-touch enabled transparent surface can be used for additional interaction. (b) Subject using the multi-touch enabled Human-Transporter metaphor to travel through a virtual 3D city model displayed on a stereoscopic back-projection wall. The setup consists of a transparent FTIR-based multi-touch surface statically mounted on a camera tripod for hand input, and a Nintendo Balance Board for foot input.

generating, processing and visualizing these complex data sets has been addressed through many sophisticated algorithms, current navigation and exploration techniques are often not sufficient for such complex environments [BCW+06]. Navigation is often referred to as the combination of *wayfinding* and *travel* [DP02]. Wayfinding denotes the *cognitive* aspects of navigation in which users have to build a cognitive, mental representation of the environment. It is thus used to determine how to get from one location to another, but does not involve the actual movement, whereas travel refers to the *physical* aspects of navigation.

In order to explore data from an egocentric point of view, it has been argued, that walking is one of the most intuitive and natural traveling technique [UAW+99]. However, real walking introduces problems in setups with limited walking space, such as CAVEs, PowerWalls or large public displays. Locomotor simulators [IHT06], omni-directional treadmills [STU07] and "redirected walking" [RKW01, SB13] provide certain solutions in this context, but often require a complex setup and can be exhausting during long term use. Similarly, many 3D input devices for IVEs are complex and exhausting to use, may divert the user's focus from her primary task [BCW+06], or can result in the user losing orientation due to unnatural motion techniques or unintended input actions.

Instead of using solely 3D hand-based input or physical locomotion systems, we propose to use a combination of hand and foot gestures for 3D traveling. Hand gestures allow precise input regarding point and area information. However, it is difficult to input continuous data with one or two hands for a long period of time. Foot interaction, in contrast, can provide continuous input by just shifting the body weight on the respective foot. Since humans primarily use their feet to travel in real life, such a foot gesture has the potential advantage of being more intuitive in the sense that it approximates a highly innate metaphor. Instead of using a 3D input device to specify, for example, the direction or the speed of the travel, we use a multi-touch enabled prop, which gives the user passive tactile feedback during the interaction and allows her to rest her arms. Multi-touch interaction has shown great potential for exploring complex content in an easy and natural manner. The geospatial domain provides a rich and interesting testbed for multi-touch applications because the command and control of geographic space (at different scales) as well as selection, modification and annotation of geospatial data are complicated tasks and have a high potential to benefit from novel interaction paradigms.

Interaction metaphors for exploration of IVEs at different scales have been exhaustively investigated in the last decades [BKH97, RSH05, WCF+05]. Some of the proposed techniques require the user to locomote through the real world and map her movements to translation and rotation of the virtual camera [IRA07, SB13]. Another class of approaches makes use of 2D or 3D input devices to specify motion parameters like direction, speed, start and stop [BKH97, FPW+00, RSH05]. The challenge to enable unlimited and efficient navigation in IVEs is further addressed by a set of techniques, which prevent user's displacements in the real world. Some remarkable examples of such techniques are presented and compared by Usoh et al. [UAW+99], which have demonstrated the benefits of natural navigation techniques in particular for egocentric navigation scenarios.

The Balance Board introduced by Nintendo in the second half of 2007 is shaped like a household body scale and contains multiple pressure sensors that are used to measure the user's center of balance and weight. Nintendo introduced several metaphors to use this board for traveling in games like a surfboard, magic carpet, or 2D transporter, and it has already been applied in some research projects. For instance, De Haan et al. [dHGP08] have applied the board to define a 3 DOF input device, which they used to implement 3D rotation or basic navigation techniques. Schöning et al. [SHR+08] have examined simultaneous usage of hands and feet to manipulate two-dimensional GIS data sets. By separating the tasks which have to be performed by hands or feet, they were able to achieve an improvement of the user's performance due to the parallel input from multiple channels. Hilsendeger et al. [HBTF09] proposed a navigation technique similar to a Human-Transporter. In their research different transfer functions for steering control, as well as differences between speed and acceleration control are evaluated. Their choice of the transfer function and control options is based on empirical consideration and lacks physical background.

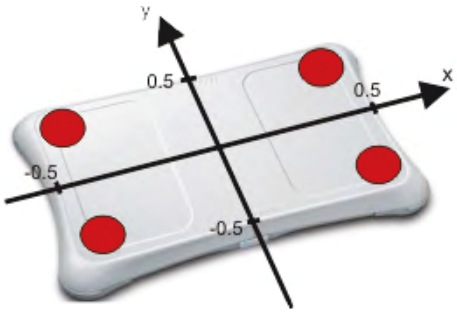

Figure 5.2: The Balance Board with the four pressure sensors at the corners.

5.3 Implementation of a Virtual Human-Transport Vehicle

In this section we explain how we implemented the Human-Transporter metaphor by a combination of hand and foot gestures for 3D traveling. We use the multi-touch surface for discrete, constrained input gestures and the Balance Board for continuous input to navigate.

5.3.1 Hardware Setup

The hardware setup (shown in Figure 5.1(b)) consists of a Balance Board and a transparent FTIR-based multi-touch surface. The multi-touch surface is an acrylic plate with a set of IR LEDs and a wide-angle (107°) camera mounted on its side. The camera is mounted outside the view frustum defined by the border of the acrylic plate and the user's head. Hence, the camera itself does not occlude objects behind the transparent surface. For finger detection ReacTIVision's[1] TUIO server was used. In order to reduce the fatigue for the user, it is mounted on a common camera tripod. The Balance Board (see Figure 5.2) features a bluetooth connection and contains multiple pressure sensors that are used to measure the user's center of balance as well as her weight. The center is determined by the intersection between an imaginary line drawn vertically through the center of mass and the surface of the Balance Board.

5.3.2 Simulation of a Human Transporter

As mentioned above, we use the Balance Board for navigation in a virtual world. The steering and speed control are inspired by the Human-Transporter vehicle and are based on the simulated setup illustrated in Figure 5.1(a). Leaning forward or backward leads to forward/backward motion of the vehicle, while leaning left or right turns it in this direction. In order to implement such a control a 2D projection of the user's center of gravity, i. e., her center of balance $C \in \mathbb{R}^2$, is used to move a *uniform* mass across the board. This way we minimize the impact of different user weights.

[1] http://reactivision.sourceforge.net/

Speed Control For speed control we use the y-component C_y of the user's center of balance. Moving the uniform mass along the y-axis of the board produces a rotational moment, which is proportional to the distance between its center of gravity and the vehicle's wheel axis. Since our coordinate system is aligned with the wheel-axis, the rotation moment is given by:

$$M(C_y) = -C_y \cdot F = -m \cdot g \cdot C_y$$

where m is the weight of the uniform mass, $g \approx 9.81 m/s^2$ is Earth's gravitational acceleration and $F = m \cdot g$ denotes the gravity force of the uniform mass. Applying reverse torque $M_R = -M$ to the wheels in order to compensate the declination of the platform results in forward (or backward for negative C_y) motion, since the wheels are not statically bound to the ground. In order to keep the calculations simple, we define the torque-to-speed function as $\omega = k \cdot M$ with a transmission coefficient $k \in \mathbb{R}$. Thus the speed for the left and the right wheels ω_R, ω_L is equal and given by:

$$\omega_R(C_y) = \omega_L(C_y) = k(C_y \cdot F) = k \cdot m \cdot g \cdot C_y \tag{5.1}$$

Steering Control By leaning left or right the user moves her center of balance along the vehicle's x-axis. This leads to different weight distribution among the two wheels. A weight applied to the board produces pressure on the wheels and thus increases the rotational friction between the wheels and the ground. Changing the weight distribution among the wheels will result in different friction and thus in different rotational speed of the two wheels, i. e., the wheel to which greater weight is applied will rotate slower than the other and a turn in this direction will be the result. For the calculation of the rotational friction force we use the simplified equation:

$$F_{fr} = \pm\ 0.01 \cdot k_{fr} \cdot F$$

where k_{fr} denotes the friction coefficient between the wheels and the ground. Here F denotes the force applied perpendicularly to the two surfaces in contact. In our case this is the gravity force of the weight applied to the wheel. The multiplication constant 0.01 is the empirically observed relation between rotational and sliding friction forces. Since the friction force takes effect against the motion direction and on the edge of the wheels, the rotational moment resulting from it is given by:

$$M_{fr} = -sign(C_y) \cdot 0.01 \cdot k_{fr} \cdot r \cdot F$$

where r denotes the radius of the wheel. If the vehicle's platform is a unit square, we can calculate the weight exerted on each wheel. With m as the weight of the uniform mass and C_x as the x-component of the user's center of balance we have:

$$m_R(C_x) = (0.5 + C_x) \cdot m$$
$$m_L(C_x) = (0.5 - C_x) \cdot m$$

Here m_R is the weight exerted on the right wheel, and m_L is the weight exerted on the left wheel. Finally, the rotational speed loss induced by the friction is:

$$\omega_{fr_R}(C_x) = -sign(C_y) \cdot 0.01 \cdot k_{fr} \cdot r \cdot m \cdot g \cdot (0.5 + C_x) \tag{5.2}$$

$$\omega_{fr_L}(C_x) = -sign(C_y) \cdot 0.01 \cdot k_{fr} \cdot r \cdot m \cdot g \cdot (0.5 - C_x) \tag{5.3}$$

The combination of (5.1) and (5.2) gives the final result for the wheel speed:

$$\begin{aligned}\omega_R(C) &= sign(C_y) \cdot m \cdot g \cdot (k \cdot |C_y| \; - \; 0.01 \cdot k_{fr} \cdot r \cdot (0.5 + C_x)) \\ &\approx sign(C_y) \cdot m \cdot g \cdot (k \cdot |C_y| + 0.01 \cdot k_{fr} \cdot C_x)\end{aligned}$$

Similarly combining (5.1) and (5.3) gives:

$$\omega_L(C) \approx sign(C_y) \cdot m \cdot g \cdot (k \cdot |C_y| - 0.01 \cdot k_{fr} \cdot C_x)$$

The last equations show that the rotational speed of each wheel is controlled by the y-component of the user's center of gravity while the x-component adds a negative correction to it, i. e., it acts as braking.

5.3.3 Flyer Metaphor

It is an interesting observation that while only a few persons have ever seen a fight from a pilot's point of view, literally everybody anticipates within a fraction of a second, how the flyer navigation works. For completeness it will be described here shortly. Flyer navigation is a typical steering exploration technique [BKLP04]. To implement the flyer navigation one needs an ordinary two-dimensional pointing device, such as a mouse, and some sort of action triggers, for instance (mouse) buttons. The pointing device is used to specify the look direction of the camera and the triggers to start/stop the flight. The flight itself consists of moving the camera's position along its direction by some constant distance at equally-spaced time points. Modification of the distance leads to modification of the flying speed. The subjective feeling of the user is that she is flying in the direction at which the visible representation of the pointing device points. In some implementations the roll and yaw rotations are bound to each other, simulating the physical aircraft motion (i .e., banking turn[2]).

This navigation metaphor is especially useful for exploration purposes, because it gives a user the opportunity to view a large area of the world, by flying high, and at the same time enables to a certain extend a detailed exploration of some area or object by flying closer to it, i .e., the metaphor provides multi-scale exploration with logical and understandable control. One of the main advantage of the flyer navigation in the context of virtual explorations is its smooth transition from one place to another. The camera keeps moving as long as the user holds the trigger. The user needs only to move the pointer in some way to change the area

[2] `http://www.grc.nasa.gov/WWW/K-12/airplane/turns.html`

explored, and to get smooth camera motion. Such a controllable, continuous, visual data stream supports the operator to form a mental representation of the 3D space [BP08], and the view of the same scene through a smoothly changing perspective results in a better spatial image of the scene in the user's cognition [BP08]. The drawback of this type of navigation is that it is not possible to make rapid turns or detailed exploration in an intuitive way. In particular, it is difficult to navigate backwards. Another drawback is that the camera motion and orientation are too constrained by each other, making it impossible to fly in some direction while looking into another, thus preventing a rear look at any object.

Since the flyer metaphor has been proven to be very helpful for spatial orientation and performing way planning/finding tasks, we have decided to implement it with the same hardware setup as for the human transporter. Here we use the Balance Board as "pointing device" and control the start and stop of the interaction implicitly by the flyer speed. Nevertheless, because of the positive feedback which we have received in earlier work, we have decided to keep the steering concept of the Human Transporter unchanged for the implementation of the Flyer metaphor and to extend it in order to provide the additional degrees of freedom needed by a flying metaphor (i .e., changing the height of the virtual camera). Reflecting on the formulas in the previous section, one will see that if the user presses her toes on one side of the Balance Board and her heel on the diagonally opposite side, the vehicle will (a bit counter-intuitive) not move, because the weight distribution between the two wheels remains the same and the forward and backward rotational moments compensate each other. Thus, we can use this particular gesture for changing the current height of the virtual camera. By utilizing this unused foot-gesture for height control and keeping the overall steering unchanged, we provide a user with the option of using both metaphors without switching between one another. In fact the Human Transporter metaphor becomes a subset of the Flyer metaphor in which the height of the virtual camera is fixed to a constant value.

5.3.4 Transparent Multi-touch Surface

As mentioned above, we make use of a multi-touch surface, which consists of an acrylic plate with a set of IR LEDs and a wide-angle (107°) camera mounted on its side. Since we can track both the user's head as well as the multi-touch surface in our setup, the transparency of the surface allows to display objects on a projection screen behind the surface in such a way that they appear either behind, attached to or in front of the multi-touch surface. In order to reduce the fatigue for the user, the surface is mounted on a common camera tripod.

WIM Interface In order to support the user's orientation in the VE we provide a WIM view, which the user can control using multi-touch gestures. The WIM view is displayed as inset in the viewport that is used to render the egocentric view of the VE, which the user perceives on the projection wall. The position of this viewport is calculated separately for the left and the right eye according to the frustum defined by the user's head and the multi-touch surface in such a way, that the WIM appears attached to the multi-touch surface (illustrated in Figure 5.1(b)). The WIM view itself is created by rendering the VE from the viewpoint

Figure 5.3: Illustration of the WIM interaction: a) Single finger pan gestures map to azimuth and elevation of a virtual trackball control centered on the user's current position. b) Pinch and rotate gestures define scaling and rotation around the main-cameras up-axis.

of an additional (slave) camera, placed with an offset relative to the egocentric camera and directed to it (s. Figure 5.3). The scale of the WIM and the position of the slave camera can be manipulated using multi-touch gestures, such as pan, pinch, rotation, etc.

Viewpoint Control Using the interactive surface the user can steer the virtual slave camera and modify the scale of the WIM view via single- and double-touch gestures. These gestures affect only the slave camera used to render the WIM, whereas the view on the projection wall is not altered. On the other hand, when using the Balance Board to move the egocentric viewpoint through the VE, the focus point for the WIM changes accordingly. This allows users to obtain additional information about the environment via the WIM while traveling using the Balance Board. Therefore, we map single finger pan gestures to azimuth and elevation of a virtual trackball control centered at the user's current position as used to render the egocentric view on the projection wall (Figure 5.3(a)). If two touch points are detected on the surface, we can define a line between them. Changing the position of one or both touch points defines a new line with different length and/or orientation. The difference of the lengths of these two lines and the angle between them is used for pinch and rotate gestures. The rotate gesture is used to rotate the slave camera around the main camera's up axis (Figure 5.3(b)). The pinch gesture is used to scale the WIM, providing users with the ability to get a detailed view of the current surroundings in an arbitrary direction or a broader overview of the environment. The transmission coefficient of the Human-Transporter, i.e., its speed sensitivity, is adjusted with the scaling factor of the WIM to provide an adequate motion speed according to the WIM view.

We also implemented long distance travel via the WIM. A user has to double-tap at the desired position in the WIM view, which places the main camera used for the projection wall at the corresponding position in the VE and aligns the WIM accordingly. Since direct "teleportation" often leads for the user to momentary loss of orientation, we have rather implemented a slow-in-slow-out type of space jump between the current and the desired location (as proposed in [WHB06]). This aims to reduce this effect and to give some time to

Figure 5.4: A screenshot of the virtual 3D city model of the city of Münster, used as a VE in the evaluation study.

the user to adjust to the new surroundings. An alternative approach could be to calculate a reasonable trajectory between the two locations and to trigger a non-interactive flight between them. This could lead to an additional advantage since the user can form a cognitive map between the original and the target position. Nevertheless, long distance traveling is usually far more involved and thus beyond the scope of these evaluations, but it may be addressed in future works.

5.4 Preliminary Evaluation

In order to analyze the benefits and drawbacks of our proposed setup we performed a simple evaluation in which we presented the hardware setup and the metaphor in a typical VR environment.

5.4.1 Participants

A total of 11 male (age 23 – 55, $\varnothing$: 29.27, height $172cm$ – $190cm$, $\varnothing$: 184.27) subjects participated in this test. Subjects were students or members of the computer science department of the University of Münster. Two of them had no relevant 3D game experience, and the rest had some. All subjects had normal or corrected to normal vision. All subjects were naïve to the experimental conditions and had never used the device before. The total time per subject including pre-questionnaire, training, experiment, breaks, and debriefing took 30 minutes. Subjects were allowed to take breaks at any time.

5.4.2 Procedure

The subjects did not receive any instructions about the underlying locomotion or steering concept. We simply told them that they have to use the device in order to navigate through a virtual 3D city model. Therefore, we used a visualization application which supports the display of large scale city environments (cf. Figure 5.4). At first, subjects had to step on the Wii Balance Board, and we allowed them to navigate for 1 minute in the virtual 3D city model without any instructions. Afterwards, we asked the subjects to navigate to certain locations in the city model. Subjects had to navigate from the start position via two stopovers to the end position. The length of the overall path was about $2.5km$. The maximum speed supported during this experiment was about $10km/h$. Since the target locations were global landmarks and all subjects were familiar with the displayed city environment, the way-finding component in this task was very simple. Therefore, subjects could concentrate on the traveling task with the metaphor. Nevertheless, since the participants have not been living in the city for the same time and have different basic orientation skills, we have not measured absolute performance parameters, such as time to completion, but rather constrained the maximal time for completion to 10 minutes. We tested two different conditions in a within-subject design, i.e., all subjects performed the test with and without a WIM. Half of the subjects performed the experiment first with the WIM and then without a WIM, half of the subjects performed the test in reversed order. The results for those participants who were not able to perform the task within the 10 minutes constraint either for the WIM or the non-WIM condition were not taken into consideration. After subjects had successfully completed the task, we interviewed them about their experiences during the test. Among other aspects they had to evaluate easiness to navigate and to orient, the locomotion and way-finding speed, precision, intuitiveness, learnability and fatigue. The subjects had to rate all aspects on a five point Likert-Scale, where 5 refers to positive evaluation, and 1 corresponds to negative evaluation.

5.4.3 Results

Overall, the feedback of the subjects was very positive. All 11 participants were able to perform the task within 10 minutes for the WIM condition, and one has failed for the non-WIM. Figure 5.5 shows the average values pooled over all subjects for both conditions, i.e., with and without WIM. We have further tested the results for significance using a two-sided t-test. The plotted results show that for all considered aspects, the condition in which the WIM is provided outperforms the non-WIM condition, except for level of fatigue, which is equal for both conditions. One of the most remarkable differences for the two conditions is the easiness to orientate. The result for the WIM condition (average of 3.7) is significantly higher than for the non-WIM condition (which was 2.1) with $p < 0.02$. Furthermore, the way-finding speed rises on average by 1.2 points from 2.6 to 3.8 when providing a WIM view, the t-test has shown for this result a significance level of $p < 0.02$. For all the other parameters no significant difference has been found. On average subjects evaluated the easiness to navigate

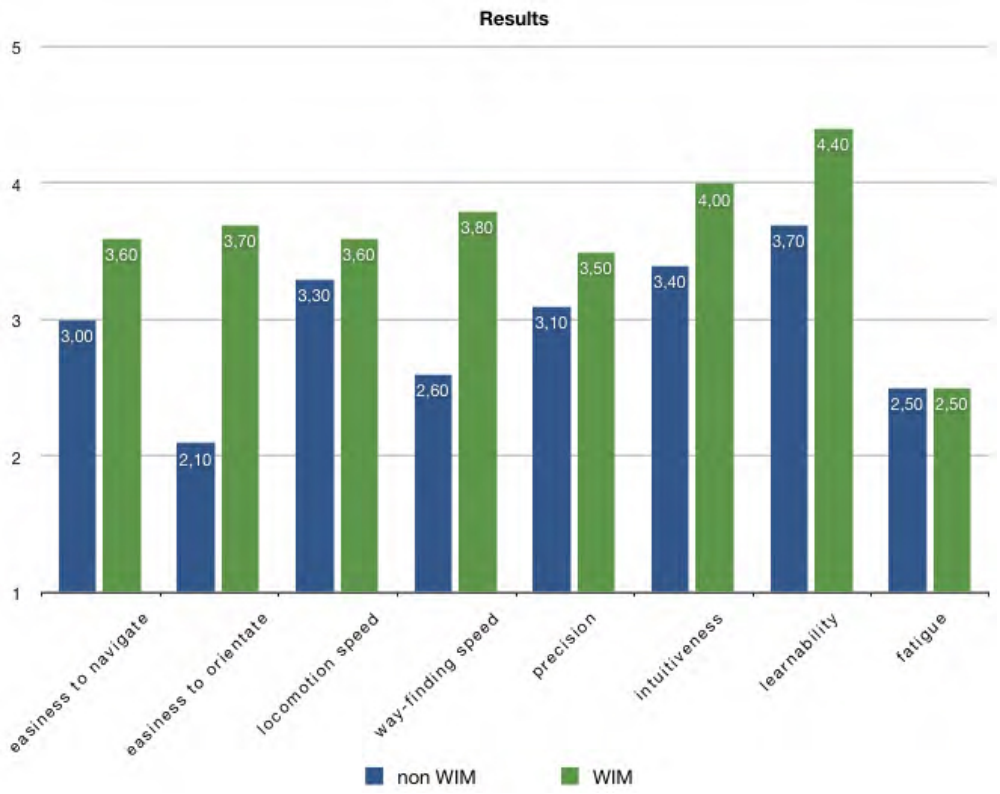

Figure 5.5: The average scores for the evaluated parameters under two conditions: one with the WIM metaphor enabled and one with the WIM disabled.

under the non-WIM condition with 3.0, whereas subjects rated this aspect with 3.6, if a WIM is provided. The locomotion speed and the precision of the metaphor are also barely affected by the usage of a WIM view (locomotion speed: 3.3 vs. 3.6, precision: 3.1 vs. 3.5). Interestingly, intuitiveness and learnability of the traveling technique are also evaluated better for the WIM condition, though no significant difference has been found. Intuitiveness grows on average from 3.4 for the non-WIM to 4.0 ($p = 0.14$) for the WIM condition, and learnability from 3.7 to 4.4 ($p = 0.19$). Contrary to our initial expectation, fatigue of the navigation technique is evaluated on average to be relatively high (though not affected by the WIM). For both conditions all subjects remarked that after a short adaptation they were able to navigate easily through the virtual world, and they found the steering very natural. Furthermore, during the experiment all subjects have tried out different options without being required to do so. Nevertheless, most subjects had particular problems to make a turn in place or within short distance. Furthermore, some of them remarked that it was difficult to maintain direction for a long time or to stay in place without moving in any direction. This could indicate a need for nonlinear speed control in our metaphor.

5.5 Conclusion and Future Work

In this chapter we have discussed the benefits and the limitations of multi-touch technologies in the context of navigation interfaces in IVEs and have introduced a novel navigation metaphor based on a device combination of a Nintendo Balance Board and a transparent FTIR-based multi-touch surface.

Overall, the feedback of the subjects was very positive and most of them were amused during the experiment, motivated to try out different options without being required to do so. Nevertheless, some users have reported problems with the navigation control. In particular, it was very difficult to make a turn in place or within short distances. Furthermore, some of the subjects remarked that it was difficult to maintain direction for a long time. While these are more or less minor problems, which might be easily fixed by some small adjustments of the foot gestures and/or the constants in the underlying equations, most users also found the technique to be very exhaustive for mainly two reasons: (a) the physical exhaustion due to the fact that the user has to lean forwards or backwards but keep her balance, which caused continuous muscle tension and (b) the perceptual exhaustion due to the fact that it was very difficult to merge the two images with such a large binocular parallax. The physical exhaustion problem might be addressed with a more elaborated mechanical setup, which compensates for the user's shifted center of balance, as the original Human Transporter device does. Another option would be to switch to isometric leaning [WL12], or to use some non-linear transfer function [HBTF09] for speed and direction control.

The perceptual exhaustion can not be alleviated with such techniques. The core of the problem is in particular the distance between the projection surface, where the user's eyes are accommodated, and the transparent props, where the user's eyes converge. There are many investigations of this so called "accommodation-convergence" problem (e.g., [DRE+11]) which show that the convenient stereoscopic parallax is at most 1/4 of the viewing distance. Nevertheless, with a transparent prop the user is usually (with our CAVE setup) at $1.5m$ distance from the display and the transparent props at arm-reach distance (about $0.6m$). Thus the parallax needed for the projection to appear attached with the prop is far beyond the maximal convenient viewing volume, which results in merging problems, eye strains and exhaustion. One option to handle this is to use tangible views [STSD10] with stereoscopic projection, but there was no appropriate hardware available at the time of writing. Another maybe more interesting option is to reduce the stereoscopic parallax of the visualization for the prop and relay more heavily on (possibly manipulated) motion parallax and perspective cues, which might "fake" the perceptional impression.

Despite the discussed drawbacks, the positive results of the preliminary usability test motivated us to further develop the proposed metaphor. In the future we want to further improve the hardware setup such that it compensates for some of the drawbacks. Moreover, we want to incorporate further interaction metaphors, which make use of the multi-touch capability of the transparent surface, and further investigate the combination of a WIM metaphor and multi-touch input.

6 Chapter 6

The VINS Framework

> Technology is nothing. What's important is that you have a faith in people, that they're basically good and smart, and if you give them tools, they'll do wonderful things with them.
>
> *(Steve Jobs)*

6.1 Design Challenges for the Interactive Graphics Frameworks

The field of interactive graphics encompasses research on various aspects of psychology and computer science, including perception and cognition, the development of multi-modal interaction, computer graphics techniques as well as hardware technology. In most interactive visualizations a computer-generated graphical environment is presented to the user, and her actions are captured with a variety of hardware devices. The most sophisticated HCI interfaces are provided by the IVEs, where usually tracked head movements are mapped to camera motions in the virtual world. In addition, many applications include sophisticated auditory and haptic rendering. Thus HCI interfaces are often complex hardware and software systems that require application developers to be knowledgeable in different areas of computer science, engineering and psychology, and to integrate or implement libraries or frameworks in extensive software engineering projects.

Nevertheless, graphical environments, hardware configurations and programming languages differ significantly between and within research groups, being constantly adapted for shifted research requirements and often replaced due to rapid developments in the computer graphics or electronic engineering communities. In addition, interaction techniques are often implemented as prototypes for a specific laboratory setup and are therefore tightly coupled to particular hardware configurations and rendering frameworks, often based on a research group's locally developed libraries and depending on the group's experience and preferences. For these reasons, novel interactive applications usually lack portability and reusability, which

hinders collaborative work and progress in the field of complex interaction techniques, e. g., making it nearly impossible to develop interaction code collaboratively with multiple research groups and laboratories or to exchange readily developed code.

In the domain of IVEs, where such effects are most obvious, this situation can be observed in numerous demonstrations. Usually, these systems are based on immersive or semi-immersive displays and tracking systems combining head tracking and view-dependent rendering with virtual object interaction via various input devices. While such kinds of multimodal user interfaces provide compelling immersive experiences, they often lack state-of-the-art rendering technologies, i. e., the visual appearance of the VE is often antiquated in contrast to current efforts in the game or movie industry and thus does not reflect the perceptual importance of visual stimulation in multimodal environments. Often this can be traced to developers integrating hardware technology and interaction concepts designed for locally developed graphics libraries based directly on *OpenGL* or *DirectX*, or open source graphics engines such as *OGRE*, *OSG* or *IrrLicht*. Many VR libraries and toolkits have been developed on top of these rendering engines, which allow to abstract the hardware interface from the application, but cause significant re-implementation when porting a VR application to another graphics engine.

The same situation may be observed with the support for emerging input devices. While virtually all frameworks, whether fully integrated or modular, implement a hardware abstraction layer to wrap the data from different input devices into higher level events, extending those layers to support new devices or event types is usually a complex and time consuming task. Furthermore, because of the event-oriented design of most interaction frameworks, it is usually very difficult and sometimes even impossible to find a proper integration of streaming device output, e. g., a sound-stream from a microphone or a depth image stream from Microsoft's Kinect.

As a result, migration of interaction techniques from one environment to another usually leads to dropping support for many of the implementations, causing transfer of interaction techniques between researchers to be limited to general descriptions of the concepts which are sufficiently simple for easy reimplementation but may also cause loss of features on the target system.

In this chapter we introduce VINS[1] (*Virtual Interaction Namespace*), a framework which provides seamless access to a dedicated distributed shared memory space designed to support modular and reusable design of interactive systems. With VINS an interaction metaphor, whether it is implemented as function or class in the main application thread, uses its own thread or runs as its own process on another computer, can be transferred from one application to another without modifications. Conceptually, the VINS shared memory space consists of very small data blocks called *variables*, which are hierarchically grouped into *namespaces*. The functional blocks of the application, e. g., threads or processes on the same or different computers, can create, read and update variables in this data space and thus exchange internal states and results in a seamless and modular way, allowing easy transfer of interaction code between different graphics or game engines, developer teams and research groups. Since the

[1] `http://vins.uni.muenster.de/`

API of the VINS framework is kept as minimalistic as possible (single library and header files, and an end user API consisting of 3 classes with few methods), it could easily be ported to another programming or scripting language; it is currently implemented in C++.

6.2 Related Systems

Many software systems and toolkits have been proposed to support the development of interactive graphics applications, e. g., [AKO95, BJH+01, Gro05, KASK02]. These systems usually provide interfaces for specification of VEs or interaction, abstracting hardware device handling from the layout and the dynamics of the virtual scene. However, many of these systems do not provide sufficient modularity for rapid integration of the currently most advanced rendering systems and cannot be integrated easily in existing hardware or software environments.

Examples of such interactive frameworks that provide high-level interfaces for developers are *VPL's Body Electric* [AKO95] and *SGI's Open Inventor* [Gro05]. These systems allow users to specify relations between virtual objects and input or output devices in a dataflow diagram editor, but have limited program modularity [BJH+01] and in particular do not provide a dedicated interface for integrating existing interaction techniques. Various other solutions provide rapid prototyping environments for creating interactive computer graphics applications without requiring a strong technical background by abstracting control of graphics and rendering [PBC+95]. Many VR libraries are also available, but most of them focus primarily on specific display technologies or applications, making it difficult to share interaction techniques [CN95]. For instance, some libraries are based on specific rendering platforms [BLRS98, Tra99a] or focus on particular areas of virtual or augmented reality, such as the Studierstube project [SFH+02] or ARToolkit [SGLS93].

Integrated software systems, such as VRJuggler [BJH+01], Vizard [zVIb] or Virtools [zVIa], allow building high-end VR applications and have been designed to overcome the drawbacks of some of the previous systems. For instance, the DIVERSE framework [KASK02] provides an elegant solution for abstracting the particular display configurations, interaction techniques and devices. By dynamically loading and unloading pre-compiled shared modules, the framework allows to port an application from one hardware setup to another with minimal or no modifications and to exchange readily programmed DIVERSE modules. Many of these systems allow to choose from multiple wrapped rendering frameworks or provide plugins for different libraries. For instance, DIVERSE provides a selection of rendering back-ends based on OpenGL Performer [SGI04], VTK [zVT] or directly on OpenGL. However, some experience with low-level programming in these environments is required for being able to write a plugin for state-of-the-art graphics rendering or game development environments. Moreover, developed interaction techniques often cannot be shared with other research groups due to incompatible versions, licensing issues or customizations.

As discussed above, HCI developments often lack standardization of the system components and architecture, but incorporate abstractions of hardware device handling, e. g., [THS+01].

Some research was recently conducted on flexible abstraction for interaction metaphors or interface definitions, such as the VITAL [CK10] abstraction layer, or InTml [FBB+08] or CHASM [WB08] systems. In addition, some recent reviews on VR software architecture [Ste08, TJV+10, WL10] provide valuable guidelines for building re-usable and flexible VR applications. In contrast, we are interested in designing an infrastructure to connect independent (already developed) software modules in a seamless data pool, which would foster modular design and re-usability and allow application developers to easily share their work.

The idea of sharing memory space or *computer resources* in general between multiple computers connected in a network cluster is not new. Maybe the most prominent example for such a system is the MOSIX [zMO] project, which originated in the year 1977 (later branched to openMosix which continued as LinuxPMI [zLI]). MOSIX is a UNIX/Linux based operating system which allows multiple computers in a cluster to be abstracted as a single "super-computer" and seamlessly share all available resources between them. Although appealing, the idea to connect all computers in a HCI laboratory to a single super-computer and let the operating system manage communication and distribution issues is most often not suitable for state of the art interactive applications. Distributed VR frameworks such as FlowVR [AR06] or Avocado [Tra99b] usually provide more adequate solutions in this context. FlowVR, for example, abstracts functional blocks into *modules*, which work in their own threads or processes without being aware of each other. The modules exchange high level messages and meta-data with light-weight *daemons* installed on each node of the cluster, which then forward those messages to other modules if needed. The messages could also be preprocessed and modified while passing through *filters* between the modules. Even though such systems provide a superior solution for implementation of distributed, high performance VR applications, their message-oriented communication approach makes the definition of flexible and extendible I/O semantics a challenging task. Another interesting example in this context is the DIVERSE framework [KASK02], which uses the shared memory space not only for communication, but also loads its system modules in this space. In particular, this allows to change or reconfigure the system on the fly. For example one may redirect an input handler from a particular hardware device to some "fake" input provider, which benefits application development and debugging considerably. Furthermore, since DIVERSE's display formats are also handled in the same way, existing applications may be reconfigured for arbitrary (supported) display setups without the need to change anything in the application itself. Nevertheless, the building blocks of the framework are usually tightly coupled with a particular rendering back-end which hinders the integration of up-to-date rendering techniques or the usage of alternative rendering packages.

In effect, although many systems are available for creating and developing of interactive graphics applications, due to compatibility and customization issues universities and research institutions tend to write their own extensions to existing frameworks. In the next sections we present an alternative approach for encapsulating interaction metaphors in an easily shareable and integrable interaction subsystem.

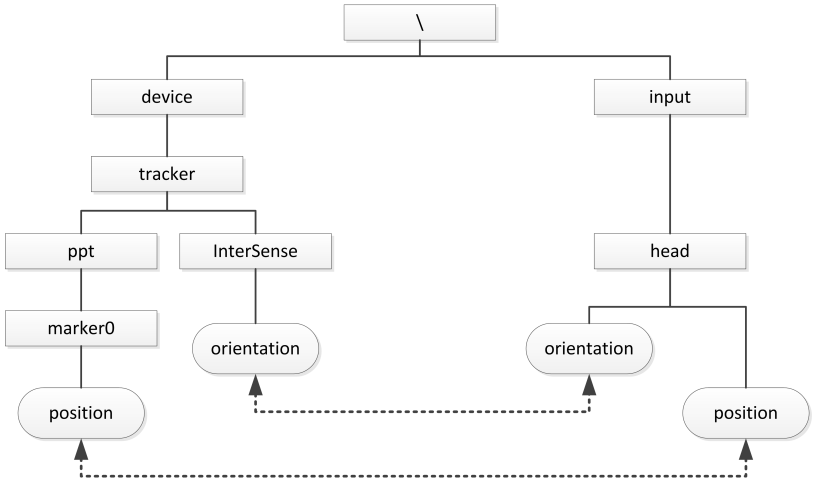

Figure 6.1: Example of a hierarchically structured virtual namespace. The path of the *variable* "position" reveals that it is the position of a tracked marker with id 0. An *alias* name of the same variable reveals additional semantics, here - the head position.

6.3 Shared Interaction Space

In this chapter we discuss a flexible framework concept capable of providing a suitable balance between common demands of interactive graphics systems. In particular, we are looking for a way to extend *already implemented* interaction techniques by only a few lines of code, so that they become portable from one hardware or software configuration to another without modification, while keeping the complexity of the system hidden from the user and the performance penalties and latency at a minimum. A concrete implementation of the concept, i. e., the VINS open source framework, is presented in Section 6.4.

6.3.1 Overall Concept

As main concept of our framework we propose to use a shared memory space with named variables, which are structured in a hierarchical manner suitable for interactive applications. Thus a hardware device could create a named variable or a set of variables and update them, sharing in this way its output data with other components of the system. An interaction technique would then read the input variable provided by the device and write to an output variable to communicate changes. Finally, the application could read a set of output variables and update its state accordingly. Existing flow-graph based frameworks, e. g., FlowVR [AR06], usually use a processing module as central building block. In such frameworks an application is considered as a set of connected modules, which exchange complex messages and attributes in some synchronized manner in order to communicate. In contrast, in our concept we concentrate on the organisation of the data itself and not on the way it is created or used. In particular, we propose the use of very simple and easily understandable data types, e. g., simple numeric types, strings and vector structs, which are hierarchically grouped in *namespaces*. The semantics of a variable could then be anticipated by tracking the

path from the variable to the root-namespace. Consider for instance the example structure shown in Figure 6.1. The meaning of the variables `position` and `orientation` could easily be anticipated by their *address path*, i. e., `\device\tracker\ppt\marker0\position` means that `position` (a `vec3f` type) is the position of marker 0, which is detected with PPT, which is a tracker device. In a similar manner `\device\tracker\InterSense\orientation` means, that `orientation` (a `quatf` type) is detected with a tracker device named InterSense. By adding *aliases*, i. e., alternative names or address patterns for the same data unit, one could extend the semantics of a variable, e. g., the aliases `\input\head\position` and `\input\head\orientation` reveal that the same variables are representing the position and orientation of the user's head and allow the interaction developer to abstract from the concrete hardware implementation of the tracking setup. In order to keep the framework as simple and flexible as possible, we consider the following requirements:

- **User interface**: The API must be kept as small as possible, still providing understandable and transparent functionality.
- **Transparent data transfer**: The framework should provide a shared data pool in which every element could be accessed in the same way, no matter if it is provided from the same thread, process or computer.
- **Temporal traceability**: The data must be provided on demand, be always available (if the provider exists), and its temporal properties (e. g., last update time, previous values) must be easily traceable.
- **Hierarchical structure**: The data pool has to provide suitable and easily extensible internal structure to support interactive applications.
- **Access to the application's internal states or structure**: In a modular system every module should be self-contained and independent from all other modules. Nevertheless, for interaction with components of the virtual environment, certain parts of this environment must be made available. For instance, certain interaction techniques require an interface for casting a ray through the virtual scene in order to select an object.

Since the proposed framework is essentially a shared memory space the API would ideally contain only two functions with simple semantic, i. e., the `get` function

```
1 T get(string variableName);
```

which retrieves the most recent value of the variable with the name `variableName` and the `set` function

```
1 void set(string variableName, const T& value);
```

which sets a new value of the variable. Nevertheless, usually some additional functions, e. g., `create, request` or `initialize`, are required. Section 6.4 gives a thorough presentation of the API and implementation details of the VINS system, which implements this concept.

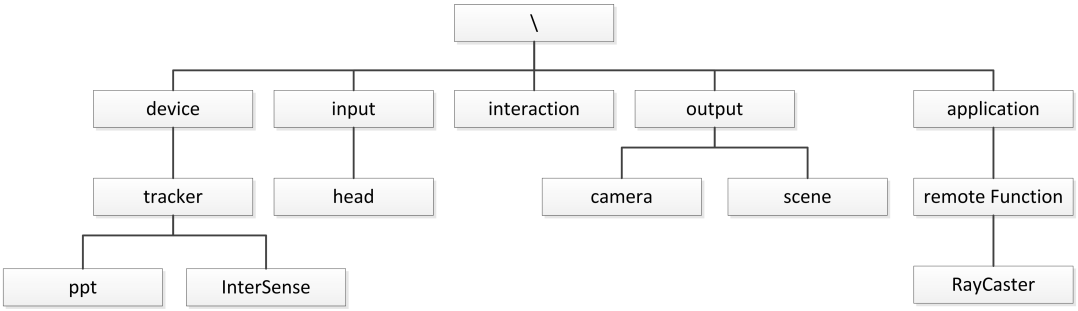

Figure 6.2: Conceptual structure of the virtual interaction namespace. The input device data is mapped under `\device`, which is then remapped under `\input`. The `\interaction` provides a space for the interactive techniques to share internal states, before the results are written to the `\output` namespace and remapped in the `\application` namespace.

Since we want to keep the latency as small as possible, it is clear that different data transfer techniques must be engaged if the variables are updated and read by the same application or by different processes on different computers. Nevertheless, in order to provide easily transferable and modular interaction techniques this complexity has to be hidden from the interaction developer. The *temporal traceability* condition allows in particular to check if the current value is up-to-date or not. Moreover, since previous values can be retrieved, interaction techniques with complex gesture recognition algorithms can be implemented.

The last two conditions, *hierarchical structure* and *access to application states*, are in the core of our concept and are discussed in more detail in the following section.

6.3.2 Hierarchical Structure

As already discussed, the core idea in our conceptual framework is to group the basic data units, i. e., the *variables*, in semantically meaningful groups which we call *namespaces*. For that we consider the basic structure shown in Figure 6.2. The structure consists of five main namespaces which are children of the root namespace "\".

Device namespace

All hardware devices should be mapped to the `\device` namespace. Thus all devices needed for interaction should create their own subspaces and variables and write their output data in those variables. The namespace might be further subdivided into device types or classes, or just by device name.

Input namespace

The input semantics for the interaction techniques must be mapped to the namespace `\input`. In particular, the structure of this namespace should reflect the meaning of the input data and not how it is acquired. The variables within the namespace are usually either simple aliases for input device data, as shown in Figure 6.1, or they are created and updated from simple

transformer functions, e. g., converting handedness and up-vector of a matrix or vector. Nevertheless, there is no constraint for the functionality or complexity of the transformers nor for their implementation. For instance, one could create an application, which reads the data from `\device\kinect\frame`, determines the user's head position and orientation with sophisticated computer vision algorithms and writes the results to the variables `\input\head\position` and `\input\head\orientation`. Since the virtual space is shared among all computers connected in a network, this application could then be easily moved to another machine to reduce the overall load on the target computer.

An instant benefit of the separation in device and input namespaces is that input sources could be redirected or manipulated on the fly, without affecting the functionality of the subsequent processing units. Indeed, similar to DIVERSE [KASK02], one could remap an input variable (e. g. `\input\head\position`) from a tracker device to, for instance, a slider or some other "fake" input generator by just changing the appropriate alias. This could allow a programmer to develop an interactive application in her office or laboratory and only test and fine-tune it on the target platform, and would significantly benefit testing and debugging of the application, since the input states and dynamics could be easily simulated.

Interaction namespace

The substructure of the `\interaction` namespace is deliberately left unspecified. In this namespace functional blocks, which implement interaction techniques, could share their internal states or, if an interaction technique is designed to be modular, their intermediate data.

Output namespace

In analogy to the `\input` namespace the `\output` namespace provides semantic abstraction of the interaction output and its mapping in the application. For instance, it might be appropriate for an interaction technique to create and manipulate the position and orientation of an "avatar" object, while the application is using a complex, animated "player" object for a visually appealing representation. Thus the variable `\output\avatar\position` should be mapped to `\application\player\gotoPosition`. Again the output mapping variables are usually created and maintained by simple functional blocks or are simply aliases of the application's variables. An appropriate initial structure of this namespace is given in [TJV+10, WL10].

Application namespace

The `\application` namespace is the domain in which application specific variables reside. In this namespace a set of variables should be provided which allow the application to synchronize its internal state in the most convenient way. More interesting is the subspace `remoteFunction`, which is aimed to allow access to the application's internal states. As an example a `RayCaster` function is shown in Figure 6.2. To use the function a functional block creates the variables `\application\remoteFunction\RayCaster\<block-name>\ray` and

`\application\remoteFunction\RayCaster\<block-name>\run`. If the functional block then sets the variable `ray` to the ray to be casted and the variable `run` to `true` the application should start updating all variables under the `\application\remoteFunction\RayCaster\<block-name>\result` namespace, i. e., position and orientation of the first intersecting object, intersection point, etc.

The type and number of input and output parameters as well as a sufficient set of remote functions is subject of specification and is currently under development.

6.3.3 Application Scope and Limitations

As mentioned previously the main objective of the concept is to provide an infrastructure to easily connect existing modules to a functional application. While this gives an application developer the freedom to design the application as she finds it best, it also means that the appropriate low-level components must be either already available or developed in the first place. In order to simplify initial development a package, which implements this concept, should provide a suitable and extensible set of pre-compiled modules. On the other hand, since the shared interaction space supports design modularity, the modules themselves are usually considerably simpler as if they were implemented as part of a single application, where commonly threading and synchronisation issues arise. In addition, already developed modules could easily be reused without modification.

When talking about distributed frameworks, it is sometimes tempting to generalize the use of a single tool or concept for each level of the distribution. However, the granularity of an interactive system might differ significantly between and within applications, ranging from distributed rendering or simulation running on a multi-processor grid through inter-process communication on the application level to collaborative applications connected throug the Internet, which have parts updating with tremendous latency. In our current focus, the presented concepts are perfectly suited for seamless communication between processes and threads running on a low latency network, as provided in most HCI laboratories. While distributed rendering is also a crucial part of many interactive visualizations, sharing named variables to communicate low level graphic primitives is usually not adequate, and there are some dedicated frameworks, e. g., FlowVR's Balzac [AR06] or EQUALIZER [zEQ], which provide more appropriate solutions in this context. Scaling a level up in the granularity, i. e., supporting distribution over low-latency networks to foster collaborative VR applications, is currently not supported by the VINS implementation, but will be addressed in the future.

6.4 Implementation of the VINS Framework

VINS is an open source, platform independent framework, which implements the concepts described in the previous section. An application using the VINS framework does not need to be distributed, but, if needed, it could easily be extended to run on multiple computers by just modifying functions or threads to run as their own process.

6.4.1 API

One of the main objectives for developing this framework was to provide a simple and transparent, but still powerful API, while hiding the complexity in the background. In the current implementation, the API is provided by 3 main classes, i. e., *Variable*, *Namespace* and *Timestamp*, and few static functions encapsulated in the class *Root*.

Before the framework can be used it has to be initialized by calling *Root*:`initialize(...)`. Once initialized the framework can be used to create variables, to request access to variables or entire namespaces and to read from or write to the variables. An example of using the API is shown in Listing 6.1. Usually an application or thread would execute some infinite loop, constantly reading the values of the variables and writing the results. To keep this example simple, we refrained from such a loop. The classes `vec{234}{ifd}`, `mat{34}{fd}` and `quat{fd}` are simple utility classes to encapsulate vectors, matrices and quaternions. Since most graphics systems already implement their own classes for this, we are providing alternative implementations, which write in a `type*` buffer provided by the user, e. g., *Root*`::get(string, float*)`.

```
#include <vins.h>
using namespace vins;

int main(int argc, char* argv[]) {
    Root:initialize("myWalkingMetaphor", ...);

    // request access to variables
    Root::request("\\input\\head\\position");
    Variable& slider = Root::request("\\input\\slider\\value");

    // create some variables
    Root::create<vec3f>("\\output\\camera\\main\\position");
    Variable& visible = Root::create<bool>("\\output\\mainMenu\\visible");

    // get the most current values of the variables
    vec3f headPos = vec3f(Root::get("\\input\\head\\position"));
    float sliderValue = float(slider);

    // ... do something

    // write the output
    Root::set("\\output\\camera\\main\\position", headPos);
    visible = bool(sliderValue >= 0.5f);
    return 0;
}
```

Listing 6.1: Example usage of the VINS API.

During the initialization the process creates its own empty representation of the global namespace. The calls of `request(...)` are needed in order to allow the process to create its own copies of the variables and connect them with their sources, which then take care of updating those copies. The call of the `create(...)` function puts a new variable with the specified type in the global namespace. If the variable is then needed by another unit, this unit gets connected to it and then receives asynchronously updates at each call of `set(...)`. As it can be seen from the example, variables can be accessed directly by calling the *Root*`::get(...)` and *Root*`::set(...)` functions. While this makes a variable accessible at each point of the program without the need for forward references between different functions and threads, it will result in traversing the namespace tree each time a variable is needed. Therefore, if more frequent access is needed, one can save the reference returned by *Root*`::request(...)` or *Root*`::get(...)`. Alternatively, one can get access to entire namespaces and address the variables within. An example for this is given in Listing 6.2.

```
1  // request access to a namespace
2  Namespace& input = Root::requestNamespace("\\input\\");
3  // ...and to subordinate namespace
4  Namespace& head = input.getNamespace("\\head\\");
5
6  // get/set values of a namespace variable
7  vec3f headPos = vec3f(head.get("position"));
8
9  // get/set values of a variable in subordinate namespace
10 float sliderValue = float(input.get("\\slider\\position"));
```

Listing 6.2: Example of using namespaces.

Using namespaces gives the programmer an option to reduce the number of namespace tree traversals to a minimum, while still maintaining a ”reasonable” amount of references.

The third and last class in the VINS API is the *Timestamp* class which is used for temporal synchronisation between the variables. By default, calling the `get(...)` method will return the most up-to-date value of a variable. Nevertheless, sometimes it is desirable to retrieve the values of a variable, which are most coherent with other related variables. Consider for example a hardware setup in which the user's head position is tracked with an optical tracking system, working (for some reason) with $25fps$, while her head orientation is tracked with an InterSence sensor with $120Hz$ refresh rate. Calling the function `get("\\input\\head\\position")` and then `get("\\input\\head\\orientation")` will result in getting the most recent value of the user's head position and orientation. Nevertheless, the value of the orientation might be offset up to $40ms$ from the value of the position. To make it even worse, there are no constraints for retrieving the variables' values at the same time; thus an application developer could, for example, retrieve the head position at the start of a function and its orientation, say $20ms$, later. To address such issues, VINS uses *timestamps*. Two examples of using timestamps for synchronisation are given in Listing 6.3.

```
Variable& headPos = Root::request("\\input\\head\\position");
Variable& headOri = Root::request("\\input\\head\\orientation");

// get the values at specific point in time
Timestamp& ts = headPos.getTimestamp();
ts.setConstraints(-5.0f, 5.0f, BEST_FIT);
ts.sample();    // lock the time

// get the value in locked time
vec3f pos = vec3f(headPos(ts));
quatf ori = quatf(headOri(ts));

// ...

// get values relative to each other
Timestamp& ts2 = headPos.getTimestamp();
ts2.setConstraints(-5.0f, 5.0f, MOST_RECENT);
headOri.setTimestamp(ts2);

pos = vec3f(headPos); // auto lock the time
// get the value in locked time
ori = quatf(headOri);

headOri.removeTimestamp();

//...
```

Listing 6.3: Example of using timestamps for synchronisation

In both cases the variable `headPos` is used to provide a timestamp for synchronisation. In the first case, the point in time is explicitly locked and used to retrieve the values of both variables. The sample time could then be locked again by just calling `sample()`. The first and second parameter of the `setConstraints(...)` function specify the maximal deviation of the sample's time from the locked time (here $\pm 5ms$), and the third parameter specifies the selection strategy if multiple samples satisfy the range, i. e., the sample closest to the locked time, the most recent sample within the range, etc. An exception is thrown, if no sample could satisfy the constraints.

In the second case the timestamp `ts2` is not locked explicitly, but registered to the `headOri` variable. Therefore, the timestamp "knows" to lock itself by the call of `headPos.get()` (here implemented as cast operator) and the variable `headOri` "knows" to return its timestamped value, and not the most recent one. The time constraint can then be released by calling `removeTimestamp()`.

In all examples until now the values of the variables were pooled by some thread. Although one can simply check if a variable has changed since the last call of `get()` by calling its

`changed()` method, in many cases an event mechanism is desired. One could implement this in VINS by just registering callback functions or methods to the framework. By default each variable or namespace is set to not fire callbacks, whether there are any registered or not. Thus this should be explicitly enabled for each variable or namespace. An example of using event callbacks is given in Listing 6.4.

```
class SomeClass {
   public:
      // class member callbacks
      void handleValueChanged(string path);
      void anotherValueChanged(Variable& v);
      // or static
      static void handleStructureChanged(string path);
      static void anotherStructureChanged(Namespace& ns);
}inst(...);

// register the callbacks
Root::addValueChangedCallback("someName", &inst,
                        SomeClass::handleValueChanged);
Root::addValueChangedCallback("anotherName", &inst,
                        SomeClass::anotherValueChanged);
Root::addStructureChangedCallback("name",
                        SomeClass::handleStructureChanged);
Root::addStructureChangedCallback("yaName",
                        SomeClass::anotherStructureChanged);

Variable& headPos = Root::request("\\input\\head\\position");
Variable& headOri = Root::request("\\input\\head\\orientation");
Namespace& input  = Root::requestNamespace("\\input\\");
// enable callbacks: call handleValueChanged on change
headPos.reportValueChanged("someName");

// all input variables should use anotherValueChanged;
// this overrides previous settings on headPos
input.reportValueChanged("anotherName");

// headOri reports nothing
headOri.reportValueChanged("");

// ... and for structure changes
input.reportStructureChanged("name");
input.getNamespace("\\head\\").reportStructureChanged("yaName");
```

Listing 6.4: Example of using callbacks

The callbacks could either be global functions or *functors*, or members of a class. The appropriate "valueChanged" callback is called each time the value of a variable gets updated. Similarly the appropriate "structureChanged" callback is called to communicate changes in the structure of the namespace, i. e., adding or removing a variable or namespace. If an empty string is set as name of the callback, the reports are efficiently suppressed for this variable or namespace, which is the default behaviour.

6.4.2 Network

From a networking point of view, the implementation could be considered as a peer-to-peer network with global coordination between one or more processes running on the same or different nodes of a cluster. The coordination server rules the connection between the processing units as well as the creation of and access to all variables and namespaces. Setting and updating of the variables' values is achieved via direct connections between the peers. Common distributed frameworks avoid depending on a global coordination server, since disconnecting the node on which the server is running would lead to disrupting all already running programs. Nevertheless, in HCI laboratories, where most of the nodes are connected to some specific hardware device, which is usually crucial for the system, e. g., a tracking server or a rendering node, simultaneous and hassle-free functioning of all nodes at all times is usually required. On the other hand, single coordination could provide some benefits against a "true" distributed architecture. For instance, in our system the coordination server is the only point at which the entire namespace with all variables (but not their values) and all connected processing units is visible. Therefore it is a suitable place for implementation of a user interface to control or debug the network. Furthermore, such an architecture allows to reduce the number of broadcasts to a minimum.

Each processing unit must first register itself to the coordination server and inform it about its *peer coordination port* and name. The name of a unit is a composition of a user defined string and the unit's host IP, process ID and thread ID, thus it allows unambiguous identification of this unit. The peer coordination port is a TCP port, which other peers should use to coordinate the data transfer with this peer. Since we want to minimize the communication latency, we are using different communication channels for data transfer between the processing units, depending on the expected data volume and refresh rates and whether they are in the same process, in different processes on the same node or on different nodes. If the processing units reside within the same process, they simply access the same internal data structure, representing the part of the namespace needed by the process. For communication between processes on the same node *named memory files* are used. Finally, for communication between processes running on different nodes a UDP connection between the nodes is used. The processing units, which need to transfer data, coordinate the connection type and endpoints via TCP connection on the peer coordination ports. In particular, if a unit wants to create a variable, it sends a request to the coordination server with information about the variable address path and its type. The coordination server then checks if a variable with the

(a) (b)

Figure 6.3: Examples of student projects using VINS. (a) A Demo of the NeoAxis game engine updated to work with HMD and optical head tracking. (b) City visualization rendered with the IrrLicht engine in our 3-Wall CAVE. Nintendo's Wii controller is used for navigation and the user's head position is tracked with Kinect (not in the picture).

same name already exists, and if not registers it to the namespace, creating all namespaces in the address path if not already existing. If then another unit requests access to the variable, the coordination server checks if it exists, and if this is the case sends the connection information of the creator to the requesting unit. The two units can then connect through TCP on their peer coordination ports and exchange information about available and needed communication channels and, in the case of success, the variable owner can start sending updates to the requesting peer. Although somehow complex, this multi-handshaking process ensures the best communication between the components of the system and allows flexibility to extend the number of available communication channels. Adding support for new communication protocols or networks, for example, would not make already compiled modules unusable, since they could simply reject the request for a communication channel and search for another one supported on both sides.

6.5 Initial Feedback

We have used VINS in the context of some of our research projects, but also in some student projects (cf. Figure 6.3). In the scope of those student projects, several students have used VINS for developing their applications using a variety of hardware systems (e. g., CAVEs, multi-touch walls, hand-held devices, Microsoft Kinects). To receive feedback from different users, we encouraged the students to inform us about any problems and to give comments about the VINS framework. In addition, we performed informal interviews and questionnaires with students, who have worked at least 3 months with VINS. The major observations concerning the requirements in terms of performance, flexibility and ease-of-use, as described in

[BJ98], as well as problems with the use of the VINS framework are discussed in the following.

Students were able to adopt existing interaction metaphors and to program custom interaction metaphors without significant effort. For instance, object selection and manipulation (i. e., translation, rotation and scaling) metaphors as well as camera manipulation metaphors could be implemented within few hours. Furthermore, students stated that VINS provides an easy to use interface to the hardware, i. e., instead of directly processing the low-level data in different formats or writing their own networking code the shared memory space allowed them to use various devices without knowing the underlying low-level data format or network connections.

As expected by using abstract data space, students were able to easily exchange their interaction metaphors independently of the used hardware. For instance, an interaction metaphor for mapping a skeletal posture onto a virtual avatar was transferred from tracking using active LED markers to tracking using the Microsoft Kinect. Moreover, students were able to easily integrate new hardware libraries, such as the Microsoft's Kinect SDK or Wiigle.

The majority of problems or inconveniences that occurred were caused by difficulties with API syntax or conceptual ideas. For example, some students, mostly used to the Java programming language, had difficulties with the overloaded cast, assignment and function call operators, commonly confusing them with constructor or reference copy semantics. Some students had also difficulties with the network connection itself, e. g., firewall setting issues or port ranges, but less with the implementation of VINS itself. However, most of these issues were easily fixed by providing the students with instructions to avoid such pitfalls.

6.6 Conclusion

In this chapter we introduced VINS, which provides a seamless distributed data pool, in which user defined named variables reside in hierarchically grouped namespaces. Although an initial structure of this shared space was discussed, there is still a long way to go until this structure becomes clean and systematized enough to provide sufficient support for a large number of applications. Furthermore, a suitable set of "standard" remote functions has to be defined, which are sufficient for multiple cases. And while the current implementation is performing sufficiently, there is still enough space for improvements. These issues as well as provision of sufficient community support will be addressed in the future.

7 Chapter 7

Conclusions and Future Work

There it is ... every writer writes with the knowledge that nothing he writes is as good as it could be.

(Harry Crews)

In this thesis we have addressed the challenge to allow touch and multi-touch interaction with virtual objects, which are stereoscopically displayed in front of or behind the surface. We have evaluated how the touch gesture changes when the objects are floating in the vicinity in front of and behind the screen and investigated the applicability of perceptual illusions for 3D stereoscopic touch interaction. The benefits and the limitations of touch interaction for exploration of virtual environments were then discussed in a case study with a multi-modal metaphor, which simulates a human transporter vehicle. In the last chapter of this work we have presented the VINS framework, which implements a seamless distributed memory space to support reusable design of interactive techniques, with special focus on 3D interactive environments.

While many of the results already provide valuable insights and many practical design implications, which an interface designer may readily use in real-world applications, there are a number of limitations and possibilities, which have not been investigated in sufficient depth. For instance, the effect of unusual viewing angles on the detection of object manipulations and the possibility to apply rotational and curvilinear object shifts have not been addressed yet. Furthermore, in our investigations of imperceptible object manipulations we have mainly concentrated on moving the objects toward the display surface, such that haptic feedback is provided at the moment in which the user touches the surface. Nevertheless, the results might be generalized for arbitrary interaction paradigms. For instance, with an in-air interaction technique a particular object might be moved toward the user's hand or away from it, depending on the "appropriateness" or on the availability of the object for the currently performed action. Thus, an object which should not be moved might be made "unreachable" for the user.

Another possible direction for future research will be to combine stereoscopic visualiza-

tion with curvilinear and shape-changing displays. As with the object shifts, recent work in the domain has revealed that haptic illusions might be provoked by visual cues. While these results have already been applied in virtual reality setups to provide some passive haptic feedback without additional instrumentation, little work has been done to prove the applicability of such interfaces in projection or desktop based scenarios. Nevertheless, this might enable many interface enhancements, especially with shape changing displays, which are currently impossible due to mechanical or electrical constraints.

In general, while we have made some of the first steps toward augmenting stereoscopic visualizations with touch interaction, there are still many opportunities for further enhancement of the overall user experience, which will be addressed in the future.

Bibliography

[AA13] F. Argelaguet and C. Andujar. A survey of 3d object selection techniques for virtual environments. *Computers & Graphics*, 37(3):121 – 136, 2013.

[AKO95] Y. Adachi, T. Kumano, and K. Ogino. Intermediate Representation for Stiff Virtual Objects. In *Proceedings of Virtual Reality Annual International Symposium*, pages 195–203, 1995.

[AR06] J. Allard and B. Raffin. Distributed physical based simulations for large vr applications. In *Proceedings of Virtual Reality (VR)*, pages 89–96, 2006.

[BCW+06] D. Bowman, J. Chen, C. A. Wingrave, J. Lucas, A. Ray, N. F. Polys, and Q. Li. New directions in 3d user interfaces. *The International Journal of Virtual Reality*, 2(5):3–14, 2006.

[Ber00] A. Berthoz. *The Brain's Sense of Movement*. Harvard University Press, 2000.

[BF07] H. Benko and S. Feiner. Balloon selection: A multi-finger technique for accurate low-fatigue 3d selection. In *Proceedings of the Symposium on 3D User Interfaces (3DUI)*, pages 79–86, 2007.

[BJ98] A. Bierbaum and C. Just. Software Tools for Virtual Reality Application Development. In *Course Notes for SIGGRAPH 98 Course 14, Applied Virtual Reality*, 1998.

[BJH+01] A. Bierbaum, C. Just, P. Hartling, K. Meinert, A. Baker, and C. Cruz-Neira. VR Juggler: A Virtual Platform for Virtual Reality Application Development. In *Proceedings of Virtual Reality (VR)*, pages 89–96, 2001.

[BKH97] D. Bowman, D. Koller, and L. Hodges. Travel in Immersive Virtual Environments: An Evaluation of Viewpoint Motion Control Techniques. In *Proceedings of Virtual Reality Annual International Symposium*, pages 45–52, 1997.

[BKLP04] D. Bowman, E. Kruijff, J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison-Wesley, 2004.

[BLRS98] R. Blach, J. Landauer, A. Roesch, and A. Simon. A Flexible Prototyping Tool for 3D Real-Time User Interaction. *Proceedings of Eurographics Workshop of Virtual Environments*, pages 195–203, 1998.

[BP08] A. Berthoz and J.-L. Petit. *The physiology and phenomenology of action*. Oxford University Press, 2008.

[BPMB05] E. Burns, A. T. Panter, M. R. McCallus, and F. P. Brooks, Jr. The hand is slower than the eye: A quantitative exploration of visual dominance over proprioception. In *Proceedings of Virtual Reality (VR)*, pages 3–10, 2005.

[BSS13] G. Bruder, F. Steinicke, and W. Stuerzlinger. Touching the void revisited: Analyses of touch behavior on and above tabletop surfaces. In *Proceedings of the IFIP TC13 Conference in Human-Computer Interaction (INTERACT)*, 2013. (in press).

[BSVH10a] G. Bruder, F. Steinicke, D. Valkov, and K. H. Hinrichs. Augmented virtual studio for architectural exploration. In *Proceedings of the Virtual Reality International Conference (VRIC)*, pages 1–8, 2010.

[BSVH10b] G. Bruder, F. Steinicke, D. Valkov, and K. H. Hinrichs. Immersive virtual studio for architectural exploration. In *Proceedings of the Symposium on 3D User Interfaces (3DUI) (Poster Presentation)*, pages 125–126, 2010.

[BSWL12] G. Bruder, F. Steinicke, P. Wieland, and M. Lappe. Tuning self-motion perception in virtual reality with visual illusions. *Transactions on Visualization and Computer Graphics (TVCG)*, 18(4), 2012.

[BW10] H. Benko and D. Wigdor. Imprecision, inaccuracy, and frustration: The tale of touch input. In C. Müller-Tomfelde, editor, *Tabletops - Horizontal Interactive Displays*, Human-Computer Interaction Series, pages 249–275. Springer London, 2010.

[BWB06] H. Benko, A. D. Wilson, and P. Baudisch. Precise selection techniques for multi-touch screens. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 1263–1272, 2006.

[Car77] R. H. S. Carpenter. *Movements of the eyes*. Pion Limited, London, 1977.

[CDH11] A. Cohé, F. Dècle, and M. Hachet. tbox: A 3d transformation widget designed for touch-screens. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 3005–3008, 2011.

[CK10] M. Csisinko and H. Kaufmann. Vital - the virtual environment interaction technique abstraction layer. In *Proceedings of the Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)*, pages 77–86. Shaker Verlag, 2010.

[CKC+10] L.-W. Chan, H.-S. Kao, M. Y. Chen, M.-S. Lee, J. Hsu, and Y.-P. Hung. Touching the void: direct-touch interaction for intangible displays. In *Proceedings of*

the Conference on Human Factors in Computing Systems (CHI), pages 2625–2634, 2010.

[CN95] C. Cruz-Neira. *Virtual Reality based on Multiple Projection Screens: The CAVE and Its Applications to Computational Science and Engineering.* PhD thesis, University of Illinois at Chicago, 1995.

[CSL+13] T. Carter, S. A. Seah, B. Long, B. Drinkwater, and S. Subramanian. Ultrahaptics: Multi-point mid-air haptic feedback for touch surfaces. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 505–514, 2013.

[DFK12] F. Daiber, E. Falk, and A. Krüger. Balloon selection revisited: Multi-touch selection techniques for stereoscopic data. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 441–444, New York, NY, USA, 2012. ACM.

[dHGP08] G. de Haan, E. J. Griffith, and F. H. Post. Using the wii balance board™as a low-cost vr interaction device. In *Proceedings of the Virtual Reality Software and Technology (VRST)*, pages 289–290, 2008.

[DKK07] A. Y. Dvorkin, R. V. Kenyon, and E. A. Keshner. Reaching within a dynamic virtual environment. *Journal of NeuroEngineering and Rehabilitation*, 4(23), 2007.

[DL01] P. Dietz and D. Leigh. DiamondTouch: a multi-user touch technology. *ACM Symposium on User Interface Software and Technology (UIST)*, pages 219–226, 2001.

[DP02] R. P. Darken and B. Peterson. *Spatial Orientation, Wayfinding, and Representation.* Lawrence Erlbaum Associates, 2002.

[DRE+11] P. Didyk, T. Ritschel, E. Eisemann, K. Myszkowski, and H.-P. Seidel. A perceptual model for disparity. *Transactions on Graphics*, 30(4), 2011.

[DVS+12] F. Daiber, D. Valkov, F. Steinicke, K. H. Hinrichs, and A. Krüger. Towards object prediction based on hand postures for reach to grasp interaction. In *Proceedings of the Workshop on The 3rd Dimension of CHI: Touching and Designing 3D User Interfaces (3DCHI)*, 2012.

[FBB+08] P. Figueroa, W. F. Bischof, P. Boulanger, H. J. Hoover, and R. Taylor. Intml: A dataflow oriented development system for virtual reality applications. *Presence: Teleoperators and Virtual Environments*, 17:492–511, 2008.

[Fer08] J. Ferwerda. Psychophysics 101: How to run perception experiments in computer graphics. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH) courses*, 2008.

[FPW+00] B. Fröhlich, J. Plate, J. Wind, G. Wesche, and M. Göbel. Cubic-Mouse-Based Interaction in Virtual Environments. *IEEE Computer Graphics*, 20(4):12–15, 2000.

[Gro05] O. I. A. Group. *Open Inventor C++ Reference Manual*. Addison-Wesley, 2005.

[GW07] T. Grossman and D. Wigdor. Going deeper: a taxonomy of 3d on the tabletop. In *Proceedings of the Workshop on Horizontal Interactive Human Computer Systems (TABLETOP)*, pages 137 – 144, 2007.

[Han05] J. Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 115–118, 2005.

[HB09] C. Holz and P. Baudisch. The generalized perceived input point model and how to double touch accuracy by extracting fingerprints. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 1165–1174, 2009.

[HB11] C. Holz and P. Baudisch. Understanding touch. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, CHI '11, pages 2501–2510, New York, NY, USA, 2011. ACM.

[HBCdlR11] M. Hachet, B. Bossavit, A. Cohé, and J.-B. de la Rivière. Toucheo: Multitouch and stereo combined in a seamless workspace. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 587–592, 2011.

[HBTF09] A. Hilsendeger, S. Brandauer, J. Tolksdorf, and C. Fröhlich. Navigation in virtual reality with the wii balanceboard™. In *GI-Workshop Virtuelle und Erweiterte Realität*. ACM, 2009.

[HCC07] M. Hancock, S. Carpendale, and A. Cockburn. Shallow-depth 3d interaction: design and evaluation of one-, two- and three-touch techniques. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 1147–1156, 2007.

[HIW+09] O. Hilliges, S. Izadi, A. D. Wilson, S. Hodges, A. Garcia-Mendoza, and A. Butz. Interactions in the Air: Adding Further Depth to Interactive Tabletops. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 139–148, 2009.

[IHT06] H. Iwata, Y. Hiroaki, and H. Tomioka. Powered Shoes. *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH) - Emerging Technologies*, 2006.

[Ins01] B. Insko. *Passive Haptics Significantly Enhances Virtual Environments*. PhD

thesis, Department of Computer Science, University of North Carolina at Chapel Hill, 2001.

[IRA07] V. Interrante, B. Riesand, and L. Anderson. Seven League Boots: A New Metaphor for Augmented Locomotion through Moderately Large Scale Immersive Virtual Environments. In *Proceedings of Symposium on 3D User Interfaces*, pages 167–170, 2007.

[JWBF01] R. S. Johansson, G. Westling, A. Bäckström, and J. R. Flanagan. Eye-Hand Coordination in Object Manipulation. *Journal of Neuroscience*, 21(17):6917–6932, 2001.

[KASK02] J. Kelso, L. Arsenault, S. Satterfield, and R. Kriz. DIVERSE: A Framework for Building Extensible and Reconfigurable Device Independent Virtual Environments. In *Proceedings of Virtual Reality (VR)*, pages 183–190, 2002.

[KBMF05] L. Kohli, E. Burns, D. Miller, and H. Fuchs. Combining Passive Haptics with Redirected Walking. In *ACM Augmented Tele-Existence*, volume 157, pages 253 – 254, 2005.

[KIP13] S.-C. Kim, A. Israr, and I. Poupyrev. Tactile rendering of 3d features on touch surfaces. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 531–538, 2013.

[Kit94] R. M. Kitchin. Cognitive maps: What are they and why study them? *Journal of Environmental Psychology*, 14(1):1 – 19, 1994.

[Koh10] L. Kohli. Redirected touching: Warping space to remap passive haptics. *Proceedings of the Symposium on 3D User Interfaces (3DUI)*, pages 129–130, 2010.

[KT04] F. L. Kooi and A. Toet. Visual comfort of binocular and 3d displays. *Displays*, 25(2–3):99–108, 2004.

[LCE08] G. Liu, R. Chua, and J. T. Enns. Attention for perception and action: task interference for action planning, but not for online control. *Experimental Brain Research*, 185:709–717, 2008.

[MCG10] A. Martinet, G. Casiez, and G. Grisoni. The design and evaluation of 3d positioning techniques for multi-touch displays. In *Proceedings of the Symposium on 3D User Interfaces (3DUI)*, 2010.

[Min95] M. Mine. Virtual Environments Interaction Technqiues. Technical Report TR95-018, UNC Chapel Hill Computer Science, 1995.

[ML04] S. Malik and J. Laszlo. Visual touchpad: a two-handed gestural input device. In *Proceedings of the Conference on Multimodal Interfaces (ICMI)*, pages 289–296, 2004.

[MOB03] A. P. Mapp, H. Ono, and R. Barbeito. What does the dominant eye dominate? a brief and somewhat contentious review. *Perception & Psychophysics*, 65(2):310–317, 2003.

[MTH+08] S. Mizobuchi, S. Terasaki, J. Häkkinen, E. Heinonen, J. Bergquist, and M. Chignell. The effect of stereoscopic viewing in a word-search task with a layered background. *Journal of the Society for Information Display*, 16(11):1105–1113, 2008.

[MTSH+10] C. Müller-Tomfelde, J. Schöning, J. Hook, T. Bartindale, D. Schmidt, P. Oliver, F. Echtler, N. Motamedi, P. Brandl, and U. Zadow. Building interactive multi-touch surfaces. In C. Müller-Tomfelde, editor, *Tabletops - Horizontal Interactive Displays*, Human-Computer Interaction Series, pages 27–49. Springer London, 2010.

[Nor98] D. Norman. *The Design of Every-Day Things*. PhD thesis, MIT, 1998.

[NS03] K. Nickel and R. Stiefelhagen. Pointing gesture recognition based on 3d-tracking of face, hands and head orientation. In *Proceedings of the Conference on Multimodal Interfaces (ICMI)*, pages 140–146, 2003.

[OS05] J.-Y. Oh and W. Stuerzlinger. Moving objects with 2d input devices in cad systems and desktop virtual environments. In *Graphics Interface 2005*, pages 195–202, 2005.

[PBC+95] R. Pausch, T. Burnette, A. C. Capehar, M. Conway, D. Cosgrove, R. DeLine, J. Durbin, R. Gossweiler, S. Koga, and J. White. A Brief Architectural Overview of Alice, a Rapid Prototyping System for Virtual Reality. In *Proceedings of Computer Graphics and Applications*, pages 195–203, 1995.

[PWF08] T. Peck, M. Whitton, and H. Fuchs. Evaluation of reorientation techniques for walking in large virtual environments. In *Proceedings of Virtual Reality (VR)*, pages 121–128, 2008.

[PWS88] R. L. Potter, L. J. Weldon, and B. Shneiderman. Improving the accuracy of touch screens: an experimental evaluation of three strategies. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 27–32, 1988.

[Raz05] S. Razzaque. *Redirected Walking*. PhD thesis, University of North Carolina, Chapel Hill, 2005.

[RDH09] J. L. Reisman, P. L. Davidson, and J. Y. Han. A screen-space formulation for 2d and 3d direct manipulation. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 69–78, 2009.

[Rek13] J. Rekimoto. Traxion: A tactile interaction device with virtual force sensation. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 427–432, 2013.

[RKW01] S. Razzaque, Z. Kohn, and M. Whitton. Redirected Walking. In *Proceedings of Eurographics*, pages 289–294, 2001.

[RSH05] T. Ropinski, F. Steinicke, and K. Hinrichs. A Constrained Road-based VR Navigation Technique for Travelling in 3D City Models. In *Proceedings of the Conference on Artificial Reality and Telexistence (ICAT)*, pages 228–235, 2005.

[San00] T. D. Sanger. Human arm movements described by a low-dimensional superposition of principal components. *Journal of Neuroscience*, 20(3):1066–1072, 2000.

[SB13] F. Steinicke and G. Bruder. Using perceptual illusions for redirected walking. *Computer Graphics and Applications - Spatial Interfaces*, 2013.

[SBJ+10] F. Steinicke, G. Bruder, J. Jerald, H. Frenz, and M. Lappe. Estimation of Detection Thresholds for Redirected Walking Techniques. *Transactions on Visualization and Computer Graphics (TVCG)*, 16(1):17–27, 2010.

[SBRH08] F. Steinicke, G. Bruder, T. Ropinski, and K. Hinrichs. Moving towards generally applicable redirected walking. In *Proceedings of the Virtual Reality International Conference (VRIC)*, pages 15–24, 2008.

[SCP95] R. Stoakley, M. Conway, and Y. Pausch. Virtual reality on a wim: interactive worlds in miniature. In *Proceedings of Conference on Human Factors in Computing Systems (CHI)*, pages 265–272, 1995.

[SD12] S. Stellmach and R. Dachselt. Look & touch: gaze-supported target acquisition. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 2981–2990, 2012.

[SES99] D. Schmalstieg, L. M. Encarnaç ao, and Z. Szalavári. Using transparent props for interaction with the virtual table. In *Proceedings of the Symposium on Interactive 3D Graphics and Games (I3D)*, pages 147–153, 1999.

[SFH+02] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavari, M. Encarnaçăo, M. Gervautz, and W. Purgathofer. The Studierstube Augmented Reality Project. In *Presence: Teleoperators and Virtual Environments*, pages 32–45, 2002.

[SFS02] M. Santello, M. Flanders, and J. F. Soechting. Patterns of hand motion during grasping and the influence of sensory guidance. *J. Neurosci.*, 22(4):1426–1435, 2002.

[SGH+12] P. Song, W. B. Goh, W. Hutama, C.-W. Fu, and X. Liu. A handle bar metaphor for virtual object manipulation with mid-air interaction. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 1297–1306, 2012.

[SGI04] SGI. *A Technical Overview of OpenGL Performer 3.1.* SGI White Paper, 2004.

[SGLS93] C. Shaw, M. Green, J. Liang, and Y. Sun. Decoupled Simulation in Virtual Reality with the MR Toolkit. *Transactions on Information Systems*, 11(3):287–317, 1993.

[SHR+08] J. Schöning, B. Hecht, M. Raubal, A. Krüger, M. Marsh, and M. Rohs. Improving interaction with virtual globes through spatial thinking: Helping users ask "why?". In *Proceedings of the Intelligent User Interfaces (IUI)*, pages 129–138. ACM Press, 2008.

[SKK+09] T. Shibata, S. Kurihara, T. Kawai, T. Takahashi, T. Shimizu, R. Kawada, A. Ito, J. Häkkinen, J. Takatalo, and G. Nyman. Evaluation of stereoscopic image quality for mobile devices using interpretation based quality methodology. In *SPIE 7237, 72371E*, 2009.

[SLE10] F. Steinicke, A. Lecuyer, and M. Ernst. *IEEE VR Tutorial: Walking Through Virtual Worlds.* Courses Notes. 2010.

[SRHM06] F. Steinicke, T. Ropinski, K. Hinrichs, and J. Mensmann. Urban City Planning in Semi-immersive Virtual Reality. In *Proceedings of the Conference on Computer Graphics Theory and Applications (GRAPP)*, pages 192–199, 2006.

[SSV+09] J. Schöning, F. Steinicke, D. Valkov, A. Krüger, and K. H. Hinrichs. Bimanual interaction with interscopic multi-touch surfaces. In *Proceedings of the IFIP TC13 Conference in Human-Computer Interaction (INTERACT)*, pages 40–53, 2009.

[Ste06] F. Steinicke. *Multimodal Metaphors for Generic Interaction Tasks in Virtual Environment.* PhD thesis, Institut für Informatik, WWU Münster, 2006.

[Ste08] A. Steed. Some useful abstractions for re-usable virtual environment platform. In *Proceedings of the Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)*, 2008.

[Ste11] F. Steinicke. Natural locomotion interfaces - with a little bit of magic! *SBC Journal on 3D Interactive Systems*, Special Issue - VR and HCILabs:86–89, 2011.

[STSD10] M. Spindler, C. Tominski, H. Schumann, and R. Dachselt. Tangible views for information visualization. In *Proceedings of the Interactive Tabletops and Surfaces (ITS)*, pages 157–166, 2010.

[STU07] M. Schwaiger, T. Thümmel, and H. Ulbrich. Cyberwalk: Implementation of a Ball Bearing Platform for Humans. In *Proceedings of HCI*, pages 926–935, 2007.

[SVH11] S. Strothoff, D. Valkov, and K. H. Hinrichs. Triangle cursor: Interactions with objects above the tabletop. In *Proceedings of the Interactive Tabletops and Surfaces (ITS)*, pages 111–119, 2011.

[THS+01] R. M. Taylor, II, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. Vrpn: a device-independent, network-transparent vr peripheral system. In *Proceedings of the Virtual Reality Software and Technology (VRST)*, pages 55–61, 2001.

[TJV+10] R. M. Taylor, J. Jerald, C. VanderKnyff, J. Wendt, D. Borland, D. Marshburn, W. R. Sherman, and M. C. Whitton. Lessons about virtual environment software systems from 20 years of ve building. *Presence: Teleoperators and Virtual Environments*, 19(2):162–178, 2010.

[Tra99a] H. Tramberend. AVANGO: A distributed virtual reality framework. In *Proceedings of the Virtual Reality (VR)*, 1999.

[Tra99b] H. Tramberend. Avocado: a distributed virtual reality framework. In *Proceedings of Virtual Reality (VR)*, pages 14–21, 1999.

[TS07] R. J. Teather and W. Stuerzlinger. Guidelines for 3d positioning techniques. In *ACM Future Play*, pages 61–68, 2007.

[UAW+99] M. Usoh, K. Arthur, M. Whitton, R. Bastos, A. Steed, M. Slater, and F. Brooks. Walking > Walking-in-Place > Flying, in Virtual Environments. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 359–364, 1999.

[VB04] D. Vogel and R. Balakrishnan. Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users. In *ACM Symposium on User Interface Software and Technology (UIST)*, pages 137–146, 2004.

[VBBS11] D. Valkov, G. Bruder, B. Bolte, and F. Steinicke. Viargo: A generic vr-based interaction library. In *Proceedings of the Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)*, 2011.

[VGH12a] D. Valkov, A. Giesler, and K. H. Hinrichs. Evaluation of depth perception for touch interaction with stereoscopic rendered objects. In *Proceedings of the Interactive Tabletops and Surfaces (ITS)*, pages 21–30, 2012.

[VGH12b] D. Valkov, A. Giesler, and K. H. Hinrichs. Vins - shared memory space for definition of interactive techniques. In *Proceedings of the Virtual Reality Software and Technology (VRST)*, pages 145–153, 2012.

[VGH14] D. Valkov, A. Giesler, and K. H. Hinrichs. Imperceptible depth shifts for touch interaction with stereoscopic objects. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*. ACM, 2014. (in press).

[VMH13] D. Valkov, A. Mantler, and K. Hinrichs. Haptic props: Semi-actuated tangible props for haptic interaction on the surface. In *Proceedings of the Adjunct Publication of the User Interface Software and Technology (UIST Adjunct)*, pages 113–114, 2013.

[VSB+10] D. Valkov, F. Steinicke, G. Bruder, K. Hinrichs, J. Schöning, F. Daiber, and A. Krüger. Touching floating objects in projection-based virtual reality environments. In *Proceedings of Joint Virtual Reality Conference (JVRC)*, pages 17–24, 2010.

[VSBH10a] D. Valkov, F. Steinicke, G. Bruder, and K. H. Hinrichs. A multi-touch enabled human-transporter metaphor for virtual 3d traveling. In *Proceedings of the Symposium on 3D User Interfaces (3DUI)*, pages 79–82, 2010.

[VSBH10b] D. Valkov, F. Steinicke, G. Bruder, and K. H. Hinrichs. Traveling in 3d virtual environments with foot gestures and a multi-touch enabled wim. In *Proceedings of Virtual Reality International Conference (VRIC 2010)*, pages 171–180, 2010.

[VSBH11] D. Valkov, F. Steinicke, G. Bruder, and K. Hinrichs. 2d touching of 3d stereoscopic objects. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 1353–1362, 2011.

[WB08] C. Wingrave and D. Bowman. Tiered developer-centric representations for 3d interfaces: Concept-oriented design in chasm. In *Proceedings of Virtual Reality (VR)*, pages 193–200, 2008.

[WB10] A. D. Wilson and H. Benko. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 273–282, 2010.

[WCF+05] M. Whitton, J. Cohn, P. Feasel, S. Zimmons, S. Razzaque, B. Poulton, and B. M. und F. Brooks. Comparing VE Locomotion Interfaces. In *Proceedings of Virtual Reality (VR)*, pages 123–130, 2005.

[WHB06] C. A. Wingrave, Y. Haciahmetoglu, and D. A. Bowman. Overcoming world in miniature limitations by a scaled and scrolling wim. In *Proceedings of the Symposium on 3D User Interfaces (3DUI)*, pages 11–16, 2006.

[WIH+08] A. Wilson, S. Izadi, O. Hilliges, A. Garcia-Mendoza, and D. Kirk. Bringing physics to the surface. In *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pages 67–76, 2008.

[WL10] C. A. Wingrave and J. J. LaViola. Reflecting on the design and implementation issues of virtual environments. *Presence: Teleoperators and Virtual Environments*, 19(2):179–195, 2010.

[WL12] J. Wang and R. Lindeman. Comparing isometric and elastic surfboard interfaces for leaning-based travel in 3d virtual environments. In *Proceedings of the Symposium on 3D User Interfaces (3DUI)*, pages 31–38, 2012.

[zAV] Avigle – avionic digital service platform . http://www.avigle.de.

[zEQ] The equalizer parallel rendering framework . http://www.equalizergraphics.com/index.html.

[zIMa] The Immersion Stereoscopic Tabletop. Immersion SAS . http://www.immersion.fr/.

[ziMb] imuts – interscopic multi-touch surfaces . http://imuts.uni-muenster.de.

[zIn] InSTInCT – Touch-based interfaces for Interaction with 3D Content . http://anr-instinct.cap-sciences.net/.

[zLI] Linuxpmi project . http://linuxpmi.org.

[zMO] Mosix – cluster operating system . http://www.mosix.cs.huji.ac.il.

[zVIa] 3dvia virtools vr library . http://www.3dviavirtools.com.

[zVIb] Vizard vr toolkit – rapid prototyping for novices . http://www.worldviz.com/products/vizard.

[zVT] Kitware's visualization toolkit (vtk) . http://www.kitware.com.