Towards Applicable 3D User Interfaces for Everyday Working Environments

Frank Steinicke, Timo Ropinski, Gerd Bruder, and Klaus Hinrichs

Visualization and Computer Graphics (VisCG) Research Group
Institute of Computer Science, Westfälische Wilhelms-Universität Münster,
Einsteinstraße 62, 48149 Münster, Germany
{fsteini,ropinski,g_brud01,khh}@math.uni-muenster.de
http://viscg.uni-muenster.de

Abstract. Desktop environments have proven to be a powerful user interface and are used as the *de facto* standard human-computer interaction paradigm for over 40 years. However, there is a rising demand on 3D applications dealing with complex datasets, which exceeds the possibilities provided by traditional devices or two-dimensional display. For these domains more immersive and intuitive interfaces are required. But in order to get the users' acceptance, technology-driven solutions that require inconvenient instrumentation, e.g., stereo glasses or tracked gloves, should be avoided. Autostereoscopic display environments equipped with tracking systems enable users to experience 3D virtual environments more natural without annoying devices, for instance via gestures. However, currently these approaches are only applied for specially designed or adapted applications without universal usability.
In this paper we introduce new 3D user interface concepts for such setups where minimal instrumentation of the user is required such that the strategies can be easily integrated in everyday working environments. Therefore, we propose an interaction system and framework which allows to display and interact with both mono- as well as stereoscopic content simultaneously. The challenges for combined mouse-, keyboard- and gesture-based input paradigms in such an environment are pointed out and novel interaction strategies are introduced.

Key words: HCI, autostereoscopic display environments, 3D user interfaces

1 Introduction

In recent years 3D user interfaces (UIs) have become more and more popular and widespread due to the requirements of several application areas, where two-dimensional desktop systems lack immersive and intuitive interaction. In addition the user's ability to perform complex interaction tasks increased, since bi-manual interactions or six degrees of freedom (DoFs) manipulations do not require much effort and are easy to learn even for non-experts. Current 3D UIs are technology-driven solutions providing more immersive exploration of and interaction with complex datasets, in particular by using stereoscopic projection and tracked six DoFs input devices. Although the costs for such a setup have reached a moderate level, experts just like ordinary users rarely uses these systems – even when 3D tasks have to be accomplished [3]. One reason for this is the inconvenient instrumentation required to allow immersive interactions, i.e., the user is forced to wear stereo glasses, tracked devices, gloves etc. [12]. Furthermore the most effective ways for humans to interact with synthetic 3D environments have not finally been resolved [3, 6]. Devices that enable control over multiple DoFs simultaneously still involve problems, which are often avoided by the usage of their 2D counterparts – as a matter of fact 2D interactions are performed best with 2D devices [3, 18, 9]. However, while in real life humans are able to move and turn objects freely in a single motion, this natural interaction is absent in two-dimensional interfaces; the user is forced to decompose 3D tasks into several 2D tasks. In addition, shortage of spatial input in typical 3D applications leads to the need to switch modes. This procedure results in ineffectiveness, in particular when switching between manipulation and navigation techniques is required in a repetitive manner.

Most desktop-based 3D applications include three-dimensional content in combination with two-dimensional elements for graphical user interface (GUI) interaction. While 3D content usually benefits from stereoscopic display, 2D GUI items often do not require immersive visualization. For such a system current autostereoscopic (AS) displays can be used to view 3D data stereoscopically without wearing any devices [8]. Thus the user is able to perceive a stereoscopic image in a fixed area called *sweet spot*. When the AS display features an optical head tracker, the user can even move in front of the display, while the tracking system can be further exploited to allow gesture-based interaction [11]. However, the separation of the stereo half images performed by an AS display (see Section 3.1) influences viewing of monoscopic content in such a way that essential elements of the GUI are distorted. Although some displays allow to display monoscopic content on the display, simultaneously display of mono- as well as stereoscopic content is not supported. Thus, simultaneous viewing requires an additional conventional display to show the monoscopic content. But only few applications support rendering of a stereoscopic window on a different display. Nevertheless, problems arise from decoupling interaction and visualization; interactions with 2D GUI elements have to be performed on the 2D screen, whereas 3D content is displayed stereoscopically on an AS display.

In this paper we introduce new 3D user interface concepts as a solution to the lack of spatial input and intuitive interaction techniques for direct manipulation of mono- as well as stereoscopic content in desktop environments. We propose an AS display environment and present a framework which enables to display arbitrary shaped areas of the GUI either in a mono- or in a stereoscopic way. Furthermore, the framework allows interaction between both “worlds” and thus opens up new vistas for human-computer interaction (HCI). Hence, the user can interact with any 2D or 3D application via familiar mouse/keyboard devices in combination with natural gestures.

The remainder of this paper is organized as follows. Section 2 summarizes related work. In Section 3 we describe the proposed setup, while Section 4 introduces interaction strategies for such everyday working environments. Section 5 presents implementation details. The results of an experimental evaluation are discussed in Section 6. Section 7 concludes the paper and gives an overview about future work.

2 Related Work

AS Display Environments In 2000, the Heinrich-Hertz-Institute built an AS display system consisting of a gaze tracker, a head and a hand tracker [11]. The head tracker gives the user a look-around capability, while the gaze tracking activates different applications on the desktop. The hand tracker enables the user to navigate and manipulate objects in 3D space via simple gestures, where computer vision is the major technological factor influencing the type of gesture that are supported. Similar approaches support gesture-based interactions by tracking the users hand and fingers with magnetic fields [23] or optical-based solutions [2]. These approaches rather address tracking technologies than advanced 3D user interfaces. Although, these systems potentially support novel forms of interaction they are restricted to specific applications designed for these setups [2]; simultaneous display of and interaction between mono- and traditional devices with stereoscopic content is not considered.

Simultaneous Monoscopic and Stereoscopic Display Although, current stereo-in-a-window systems [5, 22] show stereoscopic content either in one window time-sequentially or using filtering techniques, these technologies are restricted to only one rectangular window and glasses are still required. Hardware-based approaches have been proposed to display monoscopic and stereoscopic content simultaneously on one AS display [13]. However, interaction concepts have not yet been developed for these displays and these systems only exist as prototype solutions. Due to the lack of simultaneous display most interaction approaches only propose improvements for interactions either in 2D using monoscopic display or in 3D using stereoscopic display, but they do not combine both worlds. The interaction with stereoscopic content using two-dimensional strategies involves further problems, for instance, monoscopic representation of the mouse cursor disturbs stereoscopic perception, whereby precise interactions are impeded.

3D User Interfaces In recent years, many frameworks have been proposed which extend 2D GUIs for operating systems (OSs) to so called *3D desktops*, but also existing OSs evolve to 3D and include depth information [1, 16, **?**]. These approaches provide a virtual 3D space in which 2D GUI elements are replaced by three-dimensional counterparts. Hence, more space is available to display further information. Although these environments provide a fancy visualization, it has not been investigated in how far they improve the interaction process, since they force the user to perform 3D interactions where 2D interactions are intended. Due to the mentioned shortcomings of virtual reality (VR) interfaces, hybrid approaches have been proposed which combine 2D and 3D interaction using different display or interaction technologies [4, 21]. For example, Benko et al. have discussed techniques to grab monoscopically displayed objects from a projection screen in order to view them stereoscopically using a head mounted display [4]. However, an instrumentation of the user is still required.

Bi-manual Interactions When interacting with the hands numerous factors have to be considered. With respect to the tasks, the hands need to be moved symmetrically or asymmetrically, some tasks can be performed better with the dominant, others with the non-dominant hand. Also the used input devices have a major impact on the way how bi-manual interactions are performed. For instance, the used devices can be equal (e.g., keyboard and keyboard), different (e.g., mouse and keyboard), and they can support different DoFs or involve constraints.

These approaches are applied in everyday tasks as well as in most user interfaces. Writing on a paper, when one hand holds the pencil while the other clamps the paper, involves asymmetrical interactions. In many computer games navigation tasks are performed by the dominant hand using the mouse, whereas status changes are accomplished with the non-dominant hand via keyboard shortcuts. Interactions techniques for large-screen displays or VR environments often involve symmetrical bi-manual manipulation in order to scale, or rotate virtual objects.

However, the combination of traditional devices and gestures in AS display environments that run ordinary 3D applications has not been considered until now. The aim of this paper is not to debate the validity of desktop-based interaction concepts – there is no need to throw away 40 years of 2D UI research – neither the benefits of technology-driven VR approaches. The objective is to explore in how far these concepts can mutually adapt to each other in order to provide efficient interfaces that will be accepted by users as setups for their daily working environments.

3 Proposed System Setup

In this section we present the setup which we believe has the potential to be accepted by the users since natural as well as immersive interactions are supported, whereas instrumentation of the user is avoided.

3.1 Autostereoscopic Display Environment

On current AS displays users can see 3D data without wearing any instruments, for example by using lenticular rasters [8]. The lenticular screen is a plastic sheet molded to have the form of dozens of tiny lenses per inch. This raster operates as a beam splitter and ensures that the pixels displayed in each odd column are seen by the user's left eye, while the pixels displayed in each even column are perceived with the right eye. If the viewer positions her head in certain viewing

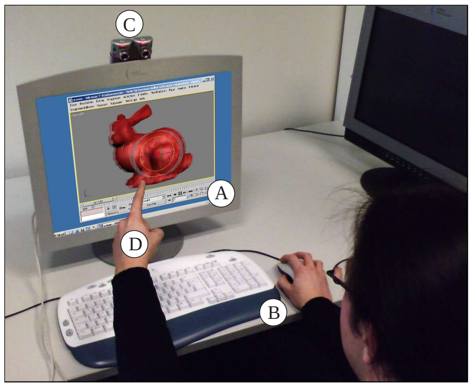

Fig. 1. 3D user interface setup includes (A) an AS display, (B) traditional mouse and keyboard, and (C) stereo-based camera setup. (D) The user applies gestures in order to perform 3D manipulations of a 3D scene. Due to the lenticular sheet the user perceives a stereoscopic image of the virtual scene displayed in 3D Studio Max.

positions, she perceives a different image with each eye giving a stereo image. When leaving a sweet spot to a neighboring position, the stereo half images have to be swapped in order to maintain the stereoscopic effect.

The separation of the stereo half images influences viewing of monoscopic content in such a way that the most essential elements of the GUI are distorted. Therefore, we have implemented a software framework (see Section 5), which provides full control over the GUI of the OS. Thus, any region or object can be displayed either mono- or stereoscopically. Furthermore, we are able to catch the entire content of any 3D graphics application based on OpenGL or DirectX. Our framework allows to change the corresponding function calls such that visualization can be changed arbitrarily. The interaction performed in our setup is primarily based on mouse and keyboard (see Figure 1). However, we have extended these devices with more natural interfaces.

3.2 Stereo-based Trackingsystem

AS displays can be equipped with eyes or head tracking systems to automatically adjust the two displayed images and the corresponding raster. Thus, the user perceives a stereo image in a larger region. Vision-based trackers enable non-intrusive, markerless computer vision based modules for HCI. When using computer vision techniques several features can be tracked, e.g., the eyes for head tracking, but it is also possible to track fingers in order to interpret simple as well as intuitive gestures in 3D. Pointing with the fingertip, for example, is an easy and natural way to select virtual objects.

As depicted in Figure 1 we use a stereo-based camera setup consisting of two USB cameras each having a resolution of 640×480 pixels. They are attached on the top of the AS display in order to track the position and orientation of certain objects. Due to the known arrangement of the cameras, the pose of geometric objects, e.g., user's hands, can be reconstructed by 3D reprojection. Besides pointing actions, some simple gestures signaling *stop*, *start*, *left* and *right* can even be recognized. These gesture input events can be used to perform 3D manipulations, e.g., to rotate or translate virtual objects (see Figure 1).

4 3D User Interface Concepts

Due to the availability of the described setup, traditional input devices can be combined with gesture-based paradigms. There are some approaches which use similar setups in artificial environments consisting of applications exclusively designed or even adapted therefore. Hence, these concepts are not applicable in daily working environments with ordinary applications. With the described framework we have full control over the GUI of the OS, in particular any arbitrarily shaped region can be displayed either mono- or stereoscopically, and each 3D application can be modified appropriately. The implementation concepts are explained in Section 5. In the following subsections we discuss implications and introduce several *universal* interaction techniques that are usable for any 3D application.

4.1 Universal Exploration

As mentioned in Section 3.1 our framework enables us to control any content of an application based on OpenGL or DirectX. Virtual scenes in such applications are often defined by so-called *display lists*. Using our framework enables us to hijack and modify these lists. Among other possibilities this issue allows us to change the viewpoint in a virtual scene. Hence, several navigation concepts can be realized that are usable for any 3D application.

Head Tracking Since binocular vision is essential for depth perception, stereoscopic projections are mainly exploited to give a better insight into complex three-dimensional datasets. Although stereoscopic display improve depth perception, viewing static images is limited, because other important depth cues, e.g, motion parallax phenomena, cannot be observed. *Motion parallax* denotes the fact that when objects or the viewer move, objects which are farther away from the viewer seem to move more slowly than objects closer to the viewer. To reproduce this effect, head tracking and view-dependent rendering is required. This can be achieved by exploiting the described tracking system (see Section 3.2). When the position and orientation of the user's head is tracked, this pose is mapped to the virtual camera defined in the 3D scene; furthermore the position of the lenticular sheet is adapted. Thus, the user is able to explore 3D datasets (to a certain degree) only by moving the tracked head. Such view-dependent rendering can also be integrated for any 3D application based on OpenGL.

Universal 3D Navigation and Manipulation Techniques However, exploration only by head tracking is limited; object rotation is restricted to the available degrees of the tracking system, e.g. 60°. Almost any interactive 3D application provides navigation techniques to explore virtual data from arbitrary

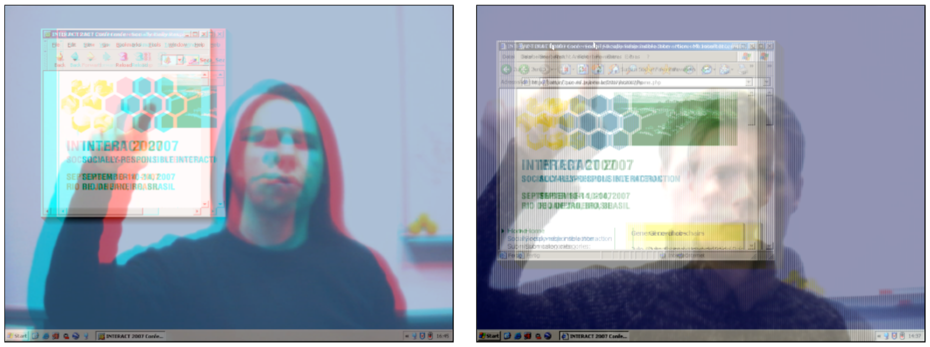

Fig. 2. Screenshot of an AS desktop overlaid with a transparent image of the user in (left) anaglyph mode and (right) vertical interlaced mode.

viewpoints. Although, many of these concepts are similar, e.g., mouse-based techniques to pan, zoom, rotate etc., 3D navigation as well as manipulation across different applications can become confusing due to various approaches.

The main idea to solve this shortcoming is to provide universal paradigms to interact with a virtual scene, i.e., using the same techniques for each 3D application. Therefore, we use gestures to translate, scale, rotate objects, respectively to move, fly, or walk through a virtual environment. These techniques are universal since they are applicable across different 3D applications. Moreover, individual strategies supported by each application can be used further on, e.g., by mouse- or keyboard-based interaction.

We have implemented these navigational concepts by using gestures based on virtual hand techniques [6]. Therefore, a one-to-one mapping in terms of translational and rotational mappings between the movements of the user's hand and the virtual scene is applied. Thus the user can start an arbitrary 3D application, activate gesture recognition and afterwards, the user can manipulate the scene by the combination of mouse, keyboard and gestures. Other concepts, such as virtual flying, walking etc. can be implemented, for instance, by virtual pointer approaches [6].

4.2 Stereoscopic Facetop Interaction

Besides depth information regarding the user's head and hand pose, we also exploit the images captured by the stereo-cameras mounted on top of the AS display (see Figure 1). Since the cameras are arranged in parallel, while their distance approximates the interpupillary distance of $\approx 65mm$, both images compose a stereoscopic image of the user. Due to the full control over the GUI, we are able to display both half images transparently into the corresponding columns of the AS display – one image into the even columns, one into the odd ones. Hence, the user sees her image superimposed on the GUI as a transparent overlay; all desktop content can still be seen, but users appear to themselves as a semi-transparent image, as if looking through a window in which they can see their own reflection. This visualization can also be used in order to enable stereo-based face-to-face collaboration which is not topic of this paper.

The technique of superimposing the user's image on top of the display has been recently used in the *Facetop* system [20]. More recently, Sony has released the Eyetoy that enables gesture interaction. In both approaches the user is able to perform 3D gestures in order to fulfill 2D interactions on the screen, where a visual feedback is given through captured images of the user. However, besides gesturing multiple DoFs for two-dimensional control, e.g., moving the mouse cursor by pointing, a stereo-based camera setup allows to use multiple DoF to enable 3D interaction. Furthermore, we use the stereoscopic projection of the user. This provides not only visual feedback about the position of the cursor on the screen surface, but also about its depth in order to simplify 3D interaction. A 3D representation of the mouse cursor is displayed at the tracked 3D position. A mouse click might be emulated if the position of the real finger and the visual representation of the finger stereoscopically displayed overlap in space. Alternatively, other gestures might be predefined, e.g., grab gestures. The depth information is also used when interacting with 2D GUIs. When using our framework, a corresponding depth is assigned to each window and it is displayed stereoscopically. In addition shadows are added to all windows to further increase depth perception. When finger tracking is activated, the user can arrange windows on the desktop in depth by pushing or pulling them with a tracked finger. Figure 2 shows screenshots of two stereoscopic facetop interaction scenarios. Both users arrange windows on the desktop by pushing them with the finger.

4.3 Combined Interaction Strategies

By using the described concepts we are able to combine desktop devices with gestures. This setup is beneficial in scenarios where the user holds a virtual object in her non-dominant hand using universal exploration gestures (see Section 4.1), while the other hand can perform precise interactions via the mouse (see Figure 1). In contrast to use only ordinary desktop devices, no context switches are required, e.g., to initiate status switches between navigation and manipulation modi. The roles of the hands may also change, i.e., the dominant hand can be used for gestures, whereas the non-dominant interacts via the keyboard.

Stereoscopic Mouse Cursor When using the described setup we experienced some drawbacks. One shortcoming, when interacting with stereoscopic representations using desktop-based interaction paradigms is the monoscopic appearance of the mouse cursor, which disturbs the stereoscopic perception. Therefore we provide two different strategies to display the mouse cursor. The first one exploits a *stereoscopic mouse cursor* which hovers over 3D objects. Thus the mouse cursor is always visible on top of the objects surface, and when moving the cursor over the surface of a three-dimensional object, the user gets an additional shape cue about the object. The alternative is to display the cursor always at the image plane. In contrast to ordinary desktop environments the mouse cursor

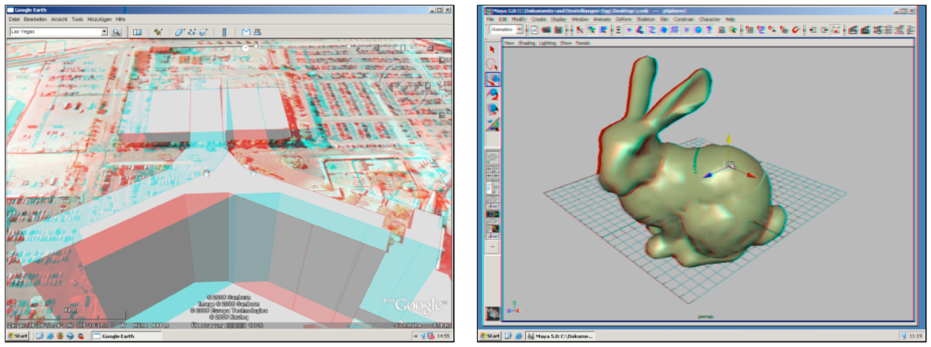

Fig. 3. Screenshots of the 3D user interface showing mono- and stereoscopic content simultaneously with appliance of (left) a stereoscopic mouse cursor and (right) monoscopic interaction lens. To perceive a stereoscopic effect both images can be viewed with anaglyph glasses.

gets invisible when it is obscured by another object extending out of the screen. Thus the stereoscopic impression is not disturbed by the mouse cursor, indeed the cursor is hidden during that time. Figure 3 (left) shows a stereoscopic scene in Google Earth where the mouse cursor is rendered stereoscopically on top of the building.

Monoscopic Interaction Lens Many 2D as well as 3D applications provide interaction concepts which are best applicable in two dimensions using 2D interaction paradigms. One example are 3D widgets [7] which reduce simultaneously manipulated DoFs. Since these interaction concepts are optimized for 2D interaction devices and monoscopic viewing we propose a *monoscopic interaction lens* through which two-dimensional interactions can be performed without loosing the entire stereoscopic effect. Therefore we attach a lens at the position of the mouse cursor. The content within such an arbitrary lens shape surrounding the mouse cursor is projected at the image plane. Thus the user can focus on the given tasks and tools to perform 2D or 3D interactions in the same way as done on an ordinary monoscopic display. This can be used to read text on a stereoscopic object, or to interact with 3D widgets.

5 Implementation

To provide a technical basis for the concepts described above, we explain some implementation details of our 3D user interface framework [17]. To allow simultaneous viewing monoscopic content need to be modified in order to make it perceivable on AS displays, while a stereo pair need to be generated out of the 3D content. Since these are diverse image processing operations first 2D is separated from 3D content. To achieve this separation, our technique acts as an integrated layer between 3D application and OS. By using this layer we ensure

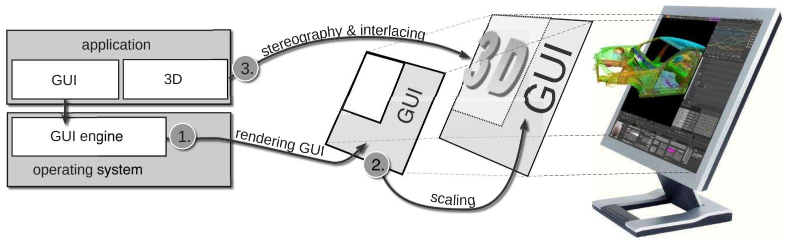

Fig. 4. Illustration of the interscopic user interface framework showing 2D and 3D content simultaneously.

that the operating system takes care about rendering 2D GUI elements in a native way (see Figure 4 (step 1)).

Processing 2D Content When viewing unadapted 2D content on AS displays two separated images are perceived by the eyes that do not match. This leads to an awkward viewing experience. To make this content perceivable we have to ensure that left and right eye perceive almost the same information, resulting in a flat two-dimensional image embedded in the image plane. To achieve this effect with (vertical-interlaced) AS displays the 2D content has to be scaled (see Figure 4 (step 2)) in order to ensure that in the odd and even columns almost same information is displayed. With respect to the corresponding factor, scaling content can yield slightly different information for both half images. However, since differences in both images are marginal, the human vision system can merge the information to a final image which can be viewed comfortably. Since we achieve proper results for a resolution of 1024×768 pixels we choose this setting for a virtual desktop from which the content is scaled to the AS displays native resolution, i.e., 1600×1200 pixels. Therefore, we had to develop an appropriate display driver which ensures that the OS announce an additional monitor with the necessary resolution and mirrors the desktop content into this screen.

Generating Stereoscopic Images Since only a few 3D applications natively support stereoscopic viewing on AS displays, in most cases we have to adapt also the 3D content in order to generate stereocopic images (see Figure 4 (step 3)). There are two techniques for making an existing 3D application stereoscopic. The first one is to trace and cache all 3D function calls and execute them twice, once for each eye. The alternative exploits image warping techniques. This technique performs a reprojection of the monoscopic image with respect to the values stored in the depth buffer. Image warping has the shortcoming that not all the scene content potentially visible from both eyes is presented in a single monoscopic image, and thus pixel filling approaches have to be applied [10]. Hence, we use the first approach, catch all 3D function calls in a display list, apply off-axis stereographic rendering, and render the content in the even and odd columns

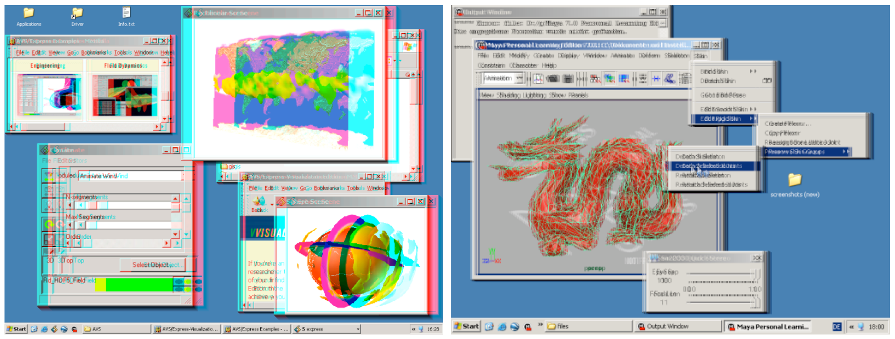

Fig. 5. Example screenshot of desktop with mono- as well as stereoscopic content shown (left) in anaglyph and (right) vertical interlaced mode.

for the left respectively right eye. We apply[**?**] with respect to the head position as described in Section 4.

Embedding Mono- and Stereoscopic Display To separate 2D and 3D content, we have to know which window areas are used for stereoscopic display. This can be either determined manually or automatically. When using the manual selection mechanism, the user is requested to add a 3D window or region and selects it to be displayed stereoscopically with the mouse cursor. When using automatic detection, our framework seeks for 3D windows based on OpenGL and applies stereoscopic rendering.

The final embedding step of 2D and 3D content is depicted by step 3 in Figure 4. An obvious problem arises, when 2D and 3D content areas overlap each other. This may happen when either a pull-down menu or a context menu overlaps a 3D canvas. In this case the separation cannot be performed on the previous 3D window selection process only. To properly render overlaying elements we apply a masking technique. This is for example important, when dealing with 3D graphics applications, whereas context menus provide convenient access to important features. When merging 2D and 3D content the mask ensures that only those areas of the 3D window are used for stereoscopic display, which are not occluded by 2D objects. Figure 5 shows two resulting screenshots in anaglyph respectively interlaced stereoscopic mode, where 3D content is shown in stereo. The windows appear at different distances to the user (see Section 4.2). The task bar and the desktop with its icons are rendered monoscopically.

6 Preliminary Experiments

In several informal user tests, all users have evaluated the usage of stereoscopic display for 3D applications as very helpful. In particular, two 3D modeling experts revealed stereoscopic visualization for 3D content in their 3D modeling environments, i.e., Maya and Cinema4D, as extremely beneficial. However, in

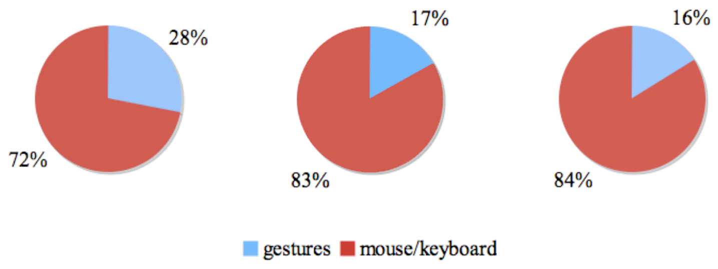

Fig. 6. Usage of gestures in comparison to traditional input devices constrained to (left) three DoFs, (middle) two DoFs and (right) one DoFs.

order to evaluate the 3D user interface we have performed a preliminary usability study. We have used the described experimental environment (see Section 3). Furthermore, we have used a 3D mouse to enable precise 3D interaction.

6.1 Experimental Tasks

We restricted the tasks to simple interactions in which four users had to delete several doors and windows from a virtual building. The building consisted of 290 triangles, where windows and doors (including 20 triangles) were uniformly separated. We have conducted three series. In the first series the user could use all provided input paradigms, i.e., mouse, keyboard, and gestures via a 3D mouse, in combination with stereoscopic visualization. In this series we have also performed subseries, where gestures were constrained to three, two and one DoFs. In the second series, only the mouse and keyboard could be used, again with stereoscopic display. In the last series, interaction was restricted to traditional devices with monoscopic visualization.

6.2 Results

We have measured the required time for the entire task and we have measured how long each input modality has been used.

Figure 6 shows that the less DoFs are available the less gestures have been used. When three Dofs were supported (left), one-third of the entire interaction time was spent on 3D manipulation by gestures with the objective to arrange the virtual building. With decreasing DoFs the required time for 3D manipulation also decreases. This is due to the fact that constraint-based interaction supports the user when arranging virtual objects. As pointed out in Figure 7 using gestures in combination with mouse and keyboard enhances performance, in particular when 3D manipulation is constrained approriatly. Participants accomplished the task fastest, when all devices could be used and only one DoFs was supported. Monoscopic display was advantageous in comparison to stereoscopic display. This is not unexpected since exploration of 3D objects was required only marginal; the focus was on simple manipulation where stereoscopic display was not essential.

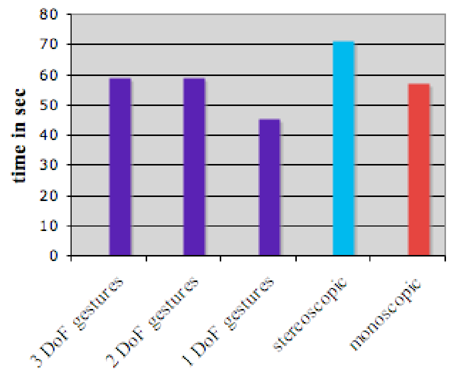

Fig. 7. Required time for the interaction task with stereoscopic display and gestures supporting three, two and one DoFs, and stereoscopic as well as monoscopic display only supporting mouse and keyboard without gesture.

7 Discussion and Future Works

In this paper we have introduced 3D user interface concepts which embed in everyday working environments providing an improved working experience. These strategies have the potential to be accepted by users as new user interface paradigm for specific tasks as well as for standard desktop interactions. The results of the preliminary evaluation indicate that the subjects are highly motivated to use the described framework, since as they remarked an instrumentation is not required. Moreover, users like the experience of using the 3D interface, especially the stereoscopic facetop approach. They evaluated the stereoscopic mouse cursor as clear improvement. The usage of the monoscopic interaction lens has been revealed as very useful because the subjects prefer to interact in a way which is familiar for them from working with an ordinary desktop system.

In the future we will integrate further functionality and visual enhancements using more stereoscopic and physics-based motion effects. Moreover, we plan to examine further interaction techniques, in particular, for domain-specific interaction tasks.

References

1. A. Agarawala and R. Balakrishnan. Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1283–1292, 2006.
2. Z. Y. Alpaslan and A. A. Sawchuk. Three-Dimensional Interaction with Autostereoscopic Displays. In A. J. Woods, J. O. Merritt, S. A. Benton, and M. R. Bolas, editors, *Proceedings of SPIE, Stereoscopic Displays and Virtual Reality Systems*, volume 5291, pages 227–236, 2004.

3. R. Balakrishnan. A Grant of 3D. Keynote speach Symposium on 3D User Interfaces, 2006.
4. H. Benko, E. W. Ishak, and S. Feiner. Cross-Dimensional Gestural Interaction Techniques for Hybrid Immersive Environments. In *Proceedings of the Virtual Reality*, pages 209–216. IEEE, 2005.
5. P. Bourke. Autostereoscopic Lenticular Images (`http://local.wasp.uwa.edu.au/~pbourke`).
6. D. Bowman, E. Kruijff, J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison-Wesley, 2004.
7. D. B. Conner, S. C. Snibbe, K. P. Herndon, D. C. Robbins, R. C. Zeleznik, and A. van Dam. Three-Dimensional Widgets. In *Symposium on Interactive 3D Graphics*, 1992.
8. N. A. Dodgson. Autostereoscopic 3D Displays. In *Computer*, volume 38, number 8, pages 31–36, 2005.
9. A. J. Hanson and E. Wernert. Constrained 3D Navigation with 2D Controllers. In *Proceedings of Visualization '97*, pages 175–182. IEEE Computer Society Press, 1997.
10. P. Kozankiewicz. Fast Algorithm for Creating Image-based Stereo Images. In *Proceedings of WSCG*, pages 59–66, 2002.
11. J. Liu, S. Pastoor, K. Seifert, and J. Hurtienne. Three Dimensional PC toward novel Forms of Human-Computer Interaction. In *Three-Dimensional Video and Display Devices and Systems SPIE*, 2000.
12. J. D. Mulder and R. van Liere. Enhancing Fish Tank VR. In *Proceedings of Virtual Reality*, pages 91–98. IEEE, 2000.
13. T. Peterka, R. L. Kooima, J. I. Girado, J. Ge, D. J. Sandin, A. Johnson, J. Leigh, J. Schulze, T. A. DeFanti. Dynallax: Solid State Dynamic Parallax Barrier Autostereoscopic VR Display. In *Proceedings of the IEEE Virtual Reality Conference 2007*, pages 91–98. IEEE, 2007.
14. Philips 42-3D6C01. (`http://www.inition.co.uk`).
15. Project Looking Glass. (`http://www.sun.com/software`).
16. G. Robertson, M. van Dantzich, D. Robbins, M. Czerwinski, K. Hinckley, K. Risden, D. Thiel, and V. Gorokhovsky. The Task Gallery: A 3D Window Manager. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 494–501, 2000.
17. T. Ropinski, F. Steinicke, G. Bruder, and K. Hinrichs. Simultaneously Viewing Monoscopic and Stereoscopic Content on Vertical-Interlaced Autostereoscopic Displays. In *Poster-Proceedings of the 33rd International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH06)*, 2006.
18. T. Salzman, S. Stachniak, and W. Stürzlinger. Unconstrained vs. Constrained 3D Scene Manipulation. In *8th IFIP International Conference on Engineering for Human-Computer Interaction*, volume 2254, 2001.
19. G. Smith, T. Salzman, and W. Stürzlinger. 3D Scene Manipulation with Constraints. In B. Fisher, K. Dawson-Howe, and C. O'Sullivan, editors, *Virtual and Augmented Architecture*, pages 35–46, 2001.
20. D. Stotts, J. C. Smith, and K. Gyllstrom FaceSpace: Endo- and Exo-Spatial Hypermedia in the Transparent Video Facetop Proceedings of the 15th ACM Conference on Hypertext and Hypermedia, pages 48 - 57, 2004.
21. Z. Szalavári and M. Gervautz. Using the Personal Interaction Panel for 3D Interaction. In *Proceedings of the Conference on Latest Results in Information Technology*, page 36, 1997.

22. H. Tramberend. A Display Device Abstraction for Virtual Reality Applications. In *Proceedings of Afrigraph*, pages 75–80, 2001.
23. C. van Berkel. Touchless Display Interaction. In *SID 02 DIGEST*, 2002.