Interscopic User Interface Concepts for Fish Tank Virtual Reality Systems

Frank Stenicke[*] Timo Ropinski[†] Gerd Bruder[‡] Klaus Hinrichs[§]

Visualization and Computer Graphics (VisCG) Research Group
Department of Computer Science
Westälische Wilhelms-Universität Münster

ABSTRACT

In this paper we introduce new user interface concepts for fish tank virtual reality (VR) systems based on autostereoscopic (AS) display technologies. Such AS displays allow to view stereoscopic content without requiring special glasses. Unfortunately, until now simultaneous monoscopic and stereoscopic display was not possible. Hence prior work on fish tank VR systems focussed either on 2D or 3D interactions.

In this paper we introduce so called *interscopic* interaction concepts providing an improved working experience, which enable great potentials in terms of the interaction between 2D elements, which may be displayed either in monoscopic or stereoscopic, e.g., GUI items, and the 3D virtual environment usually displayed stereoscopically. We present a framework which is based on a software layer between the operating system and its graphical user interface supporting the display of both mono- as well as stereoscopic content in arbitrary regions of an autostereoscopic display. The proposed concepts open up new vistas for the interaction in environments where essential parts of the GUI are displayed monoscopically and other parts are rendered stereoscopically. We address some essential issues of such fish tank VR systems and introduce intuitive interaction concepts which we have realized.

Keywords: fish tank VR, autostereoscopic displays, interscopic user interfaces

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented and virtual realities; H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical user interfaces (GUI), Interaction styles

1 INTRODUCTION

In recent years virtual environments (VEs) have become more and more popular and widespread due to the requirements of numerous application areas. Two-dimensional desktop systems are often limited in cases where natural interfaces are desired, for example, when navigating within complex 3D scenes. In such cases virtual reality (VR) systems using tracking technologies and stereoscopic projections of three-dimensional synthetic worlds support better exploration of complex datasets. Although costs as well as the effort to acquire and maintain VR systems have decreased to a moderate level, these setups are only used in highly specific application scenarios within some VR laboratories. In almost each human-computer interaction process – even when 3D tasks have to be accomplished, for instance, in CAD, CAE, or medical analysis – VR systems are rarely applied ([4]), even not by experts and least of all by ordinary users.

One reason for this is the inconvenient instrumentation required to allow immersive interactions in such VR systems, i.e., the user is forced to wear stereo glasses, tracked devices, gloves etc. Furthermore the most effective ways for humans to interact with synthetic 3D environments have not finally been resolved ([4, 8]). Devices with three or more degrees of freedom (DoFs) may provide a more direct interface to 3D manipulations than their 2D counterparts, but using multiple DoFs simultaneously still involves problems for both developers as well as users. And as a matter of fact 2D interactions are performed best with 2D devices usually supporting only two DoFs ([14, 23]). Hence 3D user interfaces are often the wrong choice in order to accomplish tasks requiring exclusively or mainly two-dimensional control ([4, 14]).

Because of their suitability for 2D and menu-based interaction tasks desktop systems with keyboard and mouse are the *de facto* user interface since long time. This user interface can even be used to perform 3D interaction, for instance, via 3D widgets ([4, 10]). However, as mentioned before there is an obvious need for interacting with 3D datasets in a more intuitive way than supported by standard desktop-based environments ([9]). Most software systems, in particular 3D modeling applications, but also scientific applications, include 2D user interface elements, such as menus, texts and images, in combination with 3D content. While 3D content usually benefit from stereoscopic visualization providing an improved depth perception, 2D GUI items often do not require stereo display. Therefore, interactions between monoscopic and stereoscopic elements, which we refer to as *interscopic interactions*, have not been investigated with special consideration of the interrelations between the elements.

For the combination of 2D and 3D content, *fish tank VR systems* ([31]) support both desktop-based as well as immersive interaction. These systems enhance desktop systems via a stereoscopic display on a conventional display. For immersive interaction head tracking is used to realize head coupled perspective stereoscopic projection. Furthermore, 3D input devices are available for intuitive interaction. Mulder et al. have stated the following benefits of fish tank VR systems ([18]). In comparison to large projection-based VR systems, e.g., CAVE ([11]), fish tank VR systems require less components, i.e., they are easier to set up, calibrate, and transport. Moreover, fish tank VR systems can provide many pixels per degree in the viewers field of view (FoV). Since fish tank VR systems require less specialized hardware, these systems are less expensive in both initial costs and maintenance. Another important factor is versatility, which means that smaller fish tank VR systems can be used as ordinary desktop computer systems, larger ones as display media for presentations. Thus fish tank VR systems have the potential to become more common in the future. Because of their desktop-based nature fish tank VR systems can be interfaced with standard desktop devices such as mouse and keyboard. Of course, also 3D input devices, e.g., tracked gloves and space mouse, are usable in fish tank VR systems. For example, pointing devices can

[*]e-mail: fsteini@math.uni-muenster.de
[†]e-mail: ropinski@math.uni-muenster.de
[‡]e-mail: g_brud01@math.uni-muenster.de
[§]e-mail: khh@math.uni-muenster.de

be used to control the mouse cursor on the screen surface of the display by projecting the device's positional and orientational data to 2D positions on the screen ([32]).

For a long time fish tank VR systems had the same problems as projection-based VR, i.e., users were forced to wear stereo glasses and head trackers. On current autostereoscopic (AS) displays users can see 3D data without wearing any instruments, because both stereo half images are spatially separated; the corresponding half images are projected to the correct eye using, for instance, lenticular rasters or LCD barriers ([12]). Thus the user is able to perceive a stereoscopic image in a fixed area called *sweet spot*. When the AS display features a head tracker in order to rearrange the raster respectively barrier, or when multiple sweet spots are supported, the user can even move in front of the display. Unfortunately, the separation of the stereo half images influences viewing of monoscopic content in such a way that the most essential elements of the GUI, such as menus, images or texts, are distorted. Although there are some displays allowing to switch off the LCD barriers in order to display monoscopic content on the display, until now it was not possible to display monoscopic and stereoscopic content simultaneously. Hence simultaneous viewing is possible only by using an additional regular 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 the stereoscopic visualization have to be viewed on a different display.

Although, current stereo-in-a-window ([6, 29]) systems show stereoscopic content in one window time-sequentially or using filtering techniques, these visualizations are restricted to only one rectangular window, while stereoscopic glasses are still required. The interaction with stereoscopic content using two-dimensional strategies involves further problems, for example, monoscopic representation of the mouse cursor disturbs stereoscopic perception, and precise interactions, for example with widgets or handles, are not possible. For these reasons, most prior work on fish tank VR systems either focussed on 2D or 3D interactions only.

In this paper we introduce new user interface strategies for fish tank VR systems using AS display technologies. The concepts are based on a framework implemented as a software layer between the operating system (OS) and its graphical user interface. Hence it is possible to display arbitrary shaped areas of the GUI either in a monoscopic or in a stereoscopic way. We address the essential issues of such fish tank VR systems and explain intuitive interaction paradigms which we have developed on top of the framework.

The remainder of this paper is structured as follows. Section 2 summarizes work related to our concepts. Before we introduce the main interscopic interaction concepts, we briefly describe implementation details and technical aspects of the framework in Section 3. In Section 4 we present interscopic interaction concepts which address problems of the interaction process in the described fish tank VR setup. Section 5 concludes this paper and gives an overview about future work.

2 Related Work

In the past much work has been done in order to provide hardware-based approaches for the interaction in fish tank VR system environments. In 2000, the *mUltimo3D* group at the Heinrich-Hertz-Institute in Germany built an AS display system consisting of a gaze tracker, a head tracker and a hand tracker. These were combined with an autostereoscopic display for viewing and manipulating objects in 3D ([16]). 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. Similar approaches support natural interactions with stereoscopic content by tracking the users hand and fingers with magnetic fields ([30]) or optical-based solutions ([3]). These systems rather address tracking technologies for fish tank VR systems than advanced interactions or visualizations in these environments.

In recent years, many software systems have been proposed which extend existing 2D desktop environments to so called *3D desktops*. These approaches provide a virtual 3D space in which 2D GUI elements such as windows are replaced by three-dimensional representations ([27, 20, 1, 21, 2]). Hence more space is available to display certain 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. Current development indicates that also graphical user interfaces for operating systems, which are essentially of a two-dimensional nature will evolve to 3D and include depth information.

At present concepts for hardware-based approaches have been proposed to display monoscopic and stereoscopic content simultaneously on one AS display ([24]). However, interaction concepts have not yet been developed for these displays. Because of the enormous costs these devices are not commercially available at present and do only exist as prototype solution.

Due to the lack of simultaneous display of monoscopic and stereoscopic content 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. There are some VR approaches which examine *hybrid interfaces* combining 2D and 3D interaction using different display or interaction technologies ([5, 28]). For example, Benko et al. propose in [5] techniques to grab monoscopically displayed objects from a projection screen in order to view them stereoscopically using a head mounted display (HMD). However, an instrumentation of the user is still required to allow this gestural-based interaction.

The aim of our research is not to debate the validity of desktop-based interaction concepts nor VR-based interaction technologies, but to explore in how far these concepts can be combined in order to provide new user interfaces and paradigms for fish tank VR system.

3 Interscopic User Interface Framework

To provide a technical basis for the visualization and interaction techniques introduced later on in this paper, we briefly discuss the implementation of our interscopic user interface framework ([22]). Though the concepts can be easily transferred to virtually any current graphics desktop system, our prototype implementation has been developed for the windows operating system. It deals with arbitrary 3D applications either driven by OpenGL or DirectX. To allow simultaneous viewing we need to modify the monoscopic content in order to make it perceivable on AS displays and generate a stereo pair out of the 3D content. Since these are diverse image processing operations first we have to separate the 2D from the 3D content. To achieve this separation, our technique acts as an integrated layer between the rendering application and the OS. By using this layer we ensure that the operating system takes care about rendering the 2D GUI elements in a native way (see Figure 1 (step 1)).

In the following subsections we first describe how to appropriately process 2D and 3D content in a graphics desktop environment, before explaining how to merge these for the display on various AS displays.

3.1 Processing 2D Content

To achieve a correct display of 2D content on AS displays it has been processed with respect to the corresponding structure, e.g., a lenticular raster, attached to the panel. The easiest case is preparing 2D content for vertical interlaced AS displays ([12]), which have a vertically oriented prism raster attached in front of the LCD panel.

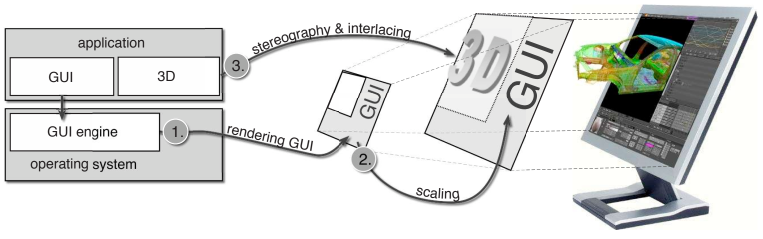


Figure 1: Illustration of the interscopic user interface framework showing 2D and 3D content simultaneously.

This prism 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. Usually images viewed on vertical-interlaced AS displays are viewed, such that the two half images (in the odd resp. even columns) show the same scene but with a binocular disparity resulting in the stereoscopic effect. However, when viewing 2D content, e.g., the desktop, the two separated images perceived by the eyes do not match, which leads to an awkward viewing experience. To make this content perceivable we have to ensure that the left and the right eye perceive the same information, resulting in a flat two-dimensional image embedded in the image plane sometimes referred to as focal plane ([7, 6]). To achieve this effect with vertical-interlaced AS displays the image has simply to be scaled in the horizontal direction by a factor of two (see Figure 1 (step 2)). This scaling ensures that on both the odd and the even columns the same information is displayed. To support other AS displays, e.g., multiple view systems ([13]) which need up to nine different views of the same scene, a corresponding image processing pattern has to be applied. However, since all AS displays have to display more than one view on the screen and we want the user to perceive the same image regardless of the user's position, the operating system has to render the 2D content with a resolution being lower than the original screen resolution of the AS display. In case n different views can be displayed by an AS display, we have to render only $\frac{1}{n}$th of the pixels needed for the current resolution. Thus we exploit a virtual desktop having $\frac{1}{n}$th the resolution of the native AS display resolution. In case of vertical-interlaced AS displays this is a virtual desktop having $\frac{1}{2}$ the resolution of the AS display (see Figure 1).

Resolution / Interpolation	Evaluation summary
800×600	4.0
1024×768 bilinear	3.24
1024×768 bicubic	3.26
1280×960 bilinear	1.92
1280×960 bicubic	1.71
1600×1200	1.0

Table 1: ASD setups for the reading task evaluation.

We have performed a usability study in order to evaluate to which degree we can scale the resolution of the virtual desktop. If the resolution is more than the half of the resolution of the target display, scaling the content yields 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 by the user.

We have tested several reading tasks on an AS display using different screen resolutions for the virtual display and different interpolation methods when scaling the content, i.e., bilinear vs. bicubic. The users had to reveal the quality, depth perception, eye-strain and further properties on a five-point Likert scale, where 1 corresponds to a negative evaluation, while 5 corresponds to the positive one. As expected users reveal the quality of the display best for the 800×600 resolution and worst for the native resolution of the display. However, because the 800×600 resolution restricts the provided display size, we decided to apply the 1024×768 resolution for the virtual display, which yields in the same resolution for the 2D information after the merging process. Since the bicubic only yields slightly better results, while requiring more calculation resources we apply the bilinear interpolation.

To exploit this virtual desktop we had to develop an appropriate display driver. This display driver ensures that the windows OS announce an additional monitor with the necessary resolution and mirrors the desktop content into this screen. For implementing this display driver we have used the driver development kit (DDK) provided by Microsoft.

The overall process for preparing 2D content is depicted in step 1 and step 2 shown in Figure 1. To allow proper viewing of 2D content on the AS display, we render these elements into the virtual desktop having at least a quarter of the resolution of the AS display (step 1). Later on the generated image is scaled uniformly by a corresponding factor before it is written to the main desktop (step 2). As mentioned above, we could also have used a virtual desktop having, for instance, half the resolution and scale the result non-uniformly by a factor of two along the horizontal axis.

In cases where a different AS display is used, the content cannot be simply scaled but it has to be displayed according to the lenses raster. This can be done easily using our approach since, the layer enables control about each pixel usually displayed by the OS. Since this is an operation being computationally slightly more complex the frame rate decreases, though allowing still real-time rendering and processing of the content.

In addition to adapt the content to be perceived on the AS display, further image processing techniques can be applied to the GUI elements in order to enhance perception and interaction behaviour as well as supporting an improved *working experience*. These techniques are described in detail in Section 4.

3.2 Generating Stereoscopic Images

Since only a few 3D applications natively support stereoscopic viewing on certain AS displays, in most cases we have to adapt also the 3D content in order to generate stereocopic images. 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. Despite of some issues with context changes this works properly for many applications; we have tested, e.g. 3D Studio Max, Maya, Cinema4D, AVS Express, Google Earth etc. However, applications making heavy usage of multipass ren-

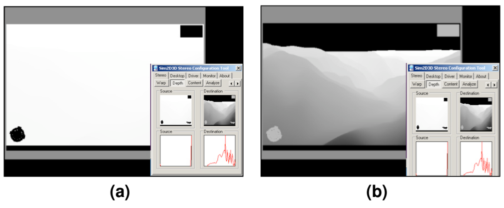


Figure 2: Google Earth example with (a) original depth values having a bad depth separation and (b) depth values after applying the range transformation.

dering were problematic in this context. For some of the concepts described later in Section 4 we have used this approach. In addition we decided to use image warping techniques ([17]) to generate stereoscopic images (see Figure 1 (step 3)). This technique performs a reprojection of the monoscopic image depending on the values stored in the depth buffer. Image warping has the drawback that not all of the scene content visible from both eyes is present in a single monoscopic image. Especially planes almost perpendicular to the view direction suffer from this effect. However, we use a simple pixel filling approach similar to the one described in ([15]). This is a scanline based image processing, which can be integrated directly into the image warping technique in order to enhance the performance. In each half image we set the color at a pixel position for which no sufficient information is available to the last proper color encountered on the scan line. In practice this leads to very convincing results, were the errors cannot be spotted anymore. Most images shown in this paper have been generated using this technique.

Since image warping takes into account the depth values of a scene, the image quality is directly dependent on the depth distribution. With existing 3D applications this becomes a problem especially when depth values are not equally distributed in the depth buffer. Figure 2 shows a screenshot of Google Earth with the accompanying depth image as well as two depth histograms generated by our application. In the left depth histogram it can be seen the head-up-display (HUD) elements at the lower-left and the upper-right are in terms of depth values far in front of the rest of the scene. Since this would lead to a stereoscopic image with a *deep* overall depth but with relatively *flat* scene objects, we apply a range transformation and obtain the depth histogram shown in Figure 2 (b). This transformation is similar to a grey value transformation as used in image processing. We analyze the depth value distribution, identify peaks and stretch them to spread the range form an offset value close to the near clipping plane till the far clipping plane. As it can be seen the range transformation can be modified using the GUI. This allows for example to eliminate the depth interval between objects, or make adjacent object even overlap in depth. Additionally, the stereo effect can be improved by interactively adapting the eye separation and the focal length, i.e., the distance from the user to the focal plane. The adapted image is illustrated in Figure 3.

Similar to the preparation of the 2D content the stereoscopy effect can also be applied when using virtually arbitrary AS displays. Only the pattern into which the half images are rendered has to be adapted. For vertical interlaced AS displays this is the typical stripe pattern, while for lenticular multiple view displays a more complex pattern has to be applied ([13]). With more sophisticated displays having an own image processor, e.g., the Philips 42-3D6C01 ([19]), we simply transmit a monoscopic image with accompanying depth information and let the display generate the appropriate half images.

3.3 Merging Mono- and Stereoscopic Content

To separate the 2D and the 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 a region and selects it to be displayed stereoscopically with the mouse cursor. In contrast to display the entire screen stereoscopically this has the benefit, that not all 3D windows are used to generate stereoscopic images. Manual selection is beneficial, for example, when having a 3D modeler providing multiple 3D views showing an object from different locations. In this case one may only want to have the perspective 3D view stereoscopic but not the orthographic side views. Furthermore, as described in Section 4.2.3 arbitrary 3D objects can be rendered stereoscopically.

The final merging step of 2D and 3D content is depicted by step 3 in Figure 1. An obvious problem arises, when 2D and 3D content areas overlap each other. This may happen when either a pull-down menu, a context menu or the mouse cursor overlaps a 3D canvas. In this case the separation cannot be performed on the previous 3D window selection process only. To properly render the mouse cursor, context menus and pop-up windows which may appear on top of the 3D canvas 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 written to the display, which are not occluded by 2D objects.

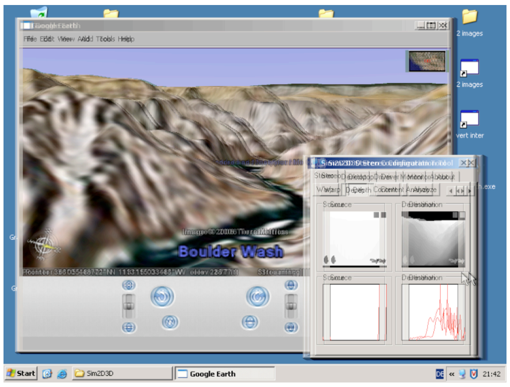


Figure 3: Example screenshot of Google Earth application and the interscopic user interface framework on a vertical interlaced AS display.

Figure 3 shows a resulting image in the described interlaced stereoscopic mode. The content of the large window is shown in stereoscopic. In addition we have assigned depth values to both windows, which makes them to appear at different distances to the user (see Section 4.1.1). The task bar and the desktop with its icons are rendered monoscopically, i.e., they have a zero parallax.

4 Interscopic Interaction Concepts

When considering factors that have led to the widespread acceptance of conventional display systems, the synergy between the display and the interaction devices and the corresponding techniques is essential. Desktop systems have shown their benefits for many problem domains including 2D as well as 3D interaction tasks. Hence there is no reason to throw out 30 years of 2D interface

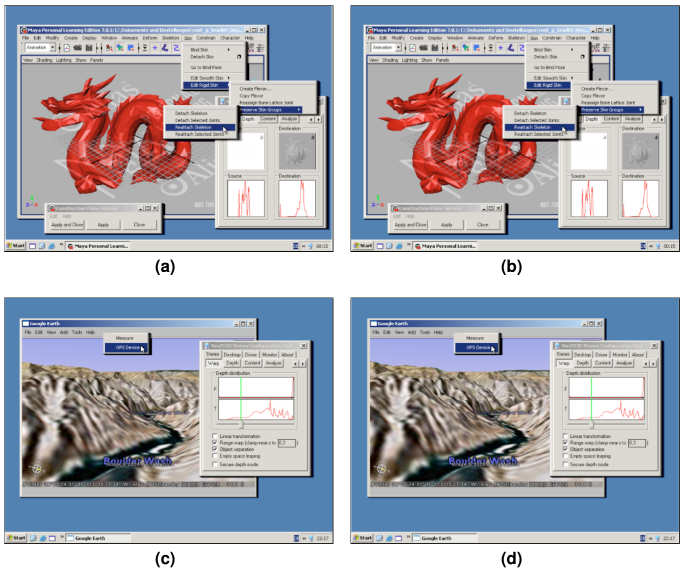


Figure 4: Autostereocopic images showing desktop with Maya modeling application (a) for the left and (b) for the right eye respectively a desktop with the Google Earth application (a) for the right eye and (b) for the left eye. To perceive a stereoscopic effect the images can be viewed from a distance of approximately 5 inch with eyes focussing at (top) infinity respectively (bottom) crossed eyes.

work. However, as mentioned before fish tank VR systems have proven the capability to increase the interaction performance and, moreover, they have the potential to be accepted by users as new user interface paradigm for specific tasks as well as for standard desktop-based interactions.

Nevertheless it is important to address the benefits of the presented fish tank VR setup and to further enhance the interaction with novel concepts in such a way that the system provides a demonstrable improvement in comparison to using an ordinary desktop system. Hence we propose concepts which provide a *focus of attention* and a *user-centered guidance model*, which supports the user during an interaction process. We enable the user to focus on the current task and provide affordances which guide the user during the interaction. For this purpose we use the functionalities of the software framework presented in Section 3 that allows us to present aribtrary elements of the user interface either in a monoscopic or stereoscopic way.

For the design of interscopic interaction techniques, we have formulated a set of rules to be observed in order to provide intuitive and helpful interaction paradigms. First of all we decide to maintain 2D tasks in 2D, we provide and use depth information only when necessary or when it supports the user. This includes to keep 2D windows aligned to the view plane because we believe that there is no need for changing the orientation of windows which would result in disturbed perception of data contained in the window, e.g., texts or images. Affordances already provided by the graphical user interface, for instance, by buttons or sliders, should be emphasized in order to guide the user. Moreover, the user's attention should be focused on the desired tasks as well as on important system information. Finally we want to use the possibilities of our concepts to allow a better immersion and fusion between the user interface and the real world using novel visual approaches.

4.1 Visual Enhancements

We introduce several visual enhancements in order to guide the user during the interaction. Hence it is easier for the user to focus on relevant aspects or certain regions of the screen and to handle application or system specific tasks.

4.1.1 Window Arrangement

When interacting either with an application consisting of multiple windows or when working with several applications simultaneously, the desktop includes numerous windows, which partially or completely overlap. 3D desktop replacements ([27, 20, 1, 21, 2]) try to address the complex representation of this situation by diminishing the limiting space of the desktop by means of providing a virtual 3D space to arrange windows. For example, the windows may be arranged on a spherical shape, oriented freely or they may be located behind each other. As mentioned above, we believe that it is important to maintain the orientation of the windows aligned to the desktop surface in order to have a sufficient viewing quality. Moreover this constrain helps to reduce the DoFs and supports the arrangement process ([26]).

However, the windows have a priority given by the system or the application in terms of the sequence in which they have to be processed; a system alert message box, for instance, has highest priority. Furthermore the user can implicitly assign the priority when accessing the windows, i.e., the longer a window has not been in the focus the lower gets its priority. We use this information and arrange these windows with respect to this priority. Hence a certain depth is assigned to each window so that it is arranged in a stereoscopic way such that the first level window is the one which appears closest to the user; the user can optionally define how depth the windows should be arranged. In addition we add shadows to all windows to further increase the depth perception.

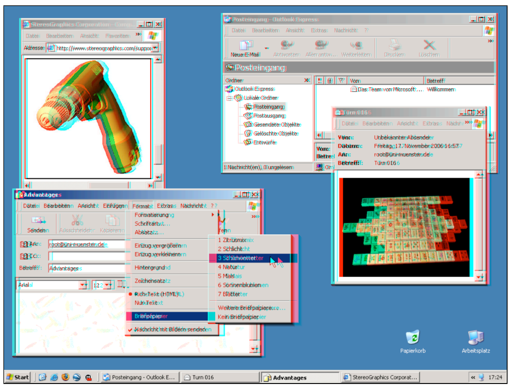


Figure 5: Interscopic user interface with several mono- as well as stereoscopic windows arranged with different depths. The image is rendered in red-cyan anaglyph mode.

The top figure of Figure 4 shows two example stereoscopic half images of the 3D modeling software Maya for the (a) left eye and (b) right eye with several windows arranged after each other. In the bottom of Figure 4 the Google Earth application is illustrated with (a) the image for the right eye and (b) the image for the left eye. The model inside the main window of the Maya application as well as the terrain model inside the Google Earth window are also rendered stereoscopically. In addition, the images show the user interface of the software in which we have integrated the concepts presented in this paper. Figure 5 shows an anaglyph version of the interscopic user interface with several mono- as well as stereoscopic windows arranged with different depths.

The opportunity to change the depth of windows allows the user also to change the depth of each window manually by using the mouse wheel. This can be exploited for example to examine the content of a window. Usually, in ordinary 2D desktop environments if the content of a window cannot be scaled the user would approach the head to the screen in order to investigate the data. As mentioned in Section 1 when using an AS display this is limited because the user has to stay in the sweet spot in order to maintain a correct stereoscopic image without any interference. Thus instead of moving the head to the display towards the window, the window can be moved from the display towards the direction of the head.

Alternatively, all windows belonging to the same application may be displayed stereoscopically with the same depth. This supports the user especially if several applications are active simultaneously. Switching to another application corresponds to interact with the windows at a different depth.

4.1.2 3D GUI Items

In a graphical user interface the usage of affordances showing the user how to interact with certain objects is essential when providing an intuitive interface ([25]). We use the presented technology to emphasize existing affordances and to introduce new ones. GUI items such as buttons etc. – even when 2D – already have a three-dimensional appearance provided by borders and shadows. In the context of an OS these items are interpreted as windows, which can be further emphasized by using stereoscopic representation for the corresponding GUI elements. Thus, buttons appear as 3D objects attached to the display's surface indicating the user to press them.

When the user moves the mouse cursor over the shape of the interaction handle, e.g., a button, the mouse cursor is also rendered stereoscopically with the corresponding depth of the underlying GUI element (compare to Section 4.2.1). Furthermore we use another effect to emphasize an affordance; we decrease the stereoscopic effect of the button which is located under the mouse cursor indicating an already initiated press action which further shows the user the affordance of a possible clicking at the button.

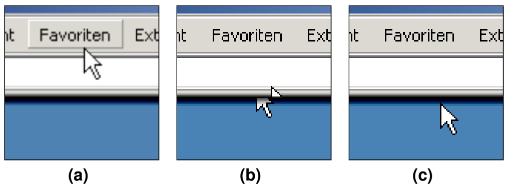


Figure 6: 3D depth cursor moved over two windows.

4.1.3 Window Extending Stereoscopic Content

Current stereo-in-a-window solutions based on time-sequential stereoscopy have the drawback, that stereoscopic content extending the window size gets cut-off, resulting in a distorted stereoscopic image. Thus, the stereoscopic effect gets lost and the immersion gets disturbed. With our framework we can hijack each stereo content based on display lists and render this information anywhere on the display either monosopically or in stereoscopically. This can be done since we can control each pixel of the user interface after the described capturing process. Thus, we are able to generate a stereoscopic window in which the 3D content can extend the given window edges, such that the objects appear to come out of the window and appear ahead of monoscopic content outside the stereoscopic window such as the 2D desktop background with its icons etc. To further increase this effect we are also able to render shadows onto the monoscopic areas behind the stereoscopic objects which extend the window edges. Moreover the stereoscopic objects can be dragged out of a stereoscopic window and they can be placed anywhere, in particular on the monoscopic desktop, while still rendered stereoscopically.

4.2 Interscopic Interactions

In the previous section we have pointed out some concepts how we emphasize affordances using visual enhancements in order to improve interscopic interaction and to advance the experience when working in fish tank VR systems. In the following subsections we consider interscopic interaction techniques which support the user when interacting with 2D and 3D content. It has been shown that using 3D interaction paradigms for inherently 2D interaction tasks is disadvantageous ([10, 14]). The more the user has to control, the more the user has to specify and think about the corresponding tasks. Therefore we constraint the interactions to those which are similar to the two-dimensional proceedings when interacting in desktop environments. The main interface for 2D interaction is the mouse cursor which may be interfaced with an ordinary mouse, a space mouse or tracked gloves by projecting the perceived DoFs data to the AS display's surface by means of ray-casting approaches. Application specific virtual input devices such as a virtual hand are controlled depending on wether the developer has implemented the interface within a corresponding 3D application.

4.2.1 3D Depth Cursor

An often named drawback, when interacting with stereoscopic representations using desktop-based interaction paradigms is the monoscopic appearance of the mouse cursor, which disturbs the perception of the stereoscopic scene. Therefore we provide two different strategies for displaying the mouse cursor. The first one exploits a 3D mouse cursor which hovers over both 3D objects within stereoscopic windows as well as over 3D GUI items as mentioned above. Thus the mouse cursor is always visible at top of the objects' surface. When moving the cursor over the surface of a three-dimensional object, the user gets an improved perception of the depth of such an object serving as an additional shape cue. The alternative is to display the cursor always at the image plane. In contrast to ordinary desktop environments the mouse cursor 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 6 shows a sequence of stereoscopic half images of a mouse cursor that is moved along the surface of two windows having different depths. In Figure 6 (b) a shadow of the window with lower depth values can be seen on the mouse cursor which is arranged behind the window.

4.2.2 Monoscopic Interaction lens

Many two-dimensional as well as three-dimensional applications provide interaction concepts which are best applicable in two dimensions using 2D interaction paradigms. One example, are 3D widgets ([10]) which reduce degrees of freedom manipulated simultaneously. Since these interaction paradigms 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 two-dimensional plane defined by the lens shape. The user can change the depth of the lenses content, which is usually displayed with zero parallax, i.e., at the image or focal plane. Thus the user can focus on the given tasks and tools to perform 2D or 3D interactions in the same as done on an ordinary monoscopic display. This can be used to read text on a stereoscopic object, or to interact with 3D widgets often used in numerous 3D applications.

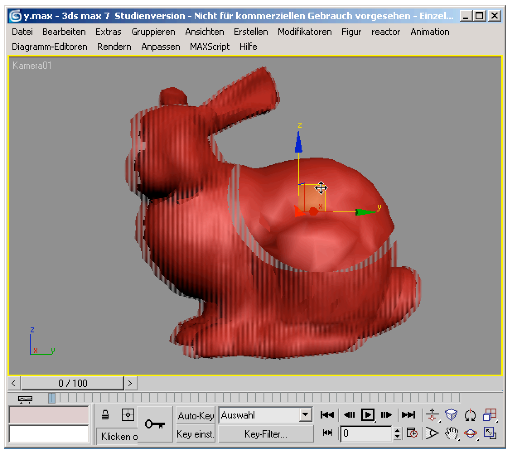

y.max - 3ds max 7 Studienversion - Nicht für kommerziellen Gebrauch vorgesehen - Einzel...
Datei Bearbeiten Extras Gruppieren Ansichten Erstellen Modifikatoren Figur reactor Animation
Diagramm-Editoren Rendern Anpassen MAXScript Hilfe
Kamera01
z
y
x
0 / 100
Klicken o
Auto-Key
Auswahl
Key einst.
Key-Filter...
0

Figure 7: Monoscopic interaction lens used to enable interaction with a 3D translation widget in 3D Studio Max

Figure 7 shows a monoscopic interaction lens used to control a 3D translation widget of 3D Studio Max. The content within the disc shaped lens centered around the mouse cursor is rendered monoscopically, while the rest of the 3D model is rendered stereoscopically.

4.2.3 Stereoscopic Lens

The opposite approach to the monoscopic interaction lens described in the previous section is to render objects within a specific lens in the stereoscopic mode. We use this feature in the following way. The user can activate a stereoscopic rendering for arbitrary objects by clicking at them when this functionally is activated. Afterwards the corresponding object is rendered stereoscopically while other objects within the same window stay in a monoscopic render mode. Furthermore the user can change the stereo parameters in order to improve the exploration of desired objects.

Figure 8 (a) shows a set of virtual 3D objects in a Maya scene, each rendered monoscopically. After the user has select the second object from the left, only this object is rendered stereoscopically and the user can adjust the stereoscopic settings using the GUI of our framework. Thus, this object of interest is emphasized and in the focus of the user's attention.

5 Conclusion and Future Work

In this paper we have introduced interscopic interaction concepts for fish tank VR systems based on AS display technologies and we have described technical aspects about the interscopic user interface framework. The proposed techniques have proven the capability to increase the interaction performance in such setups. Moreover, these strategies have the potential to be accepted by users as new user interface paradigm for specific tasks as well as for standard desktop-based interactions since they enhance the working experience, in particular, when simultaneous interaction with monoscopic and stereoscopic content is intended.

Although, we have not performed a usability study for the interscopic interaction concepts, which has to be conducted in the future work, we have surveyed subjects which have tested the interscopic user interface. The results of the survey indicate that the subjects are highly motivated to use the described fish tank VR system, since as they remarked an instrumentation is not required and the users like the experience of using the system. In particular, they evaluated the window arrangement as very helpful when they need to identify the active window. 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 perform a usability study in which we compare how effective the described concepts are in comparison to the usage of an ordinary system, for example, when performing several 3D modeling tasks. We will integrate further functionality and visual enhancements using more stereoscopic and optionally physics-based motion effects. Moreover, we will examine further interscopic interaction techniques, in particular, for domain-specific interaction tasks, and evaluate in how far they improve the interaction in fish tank VR systems.

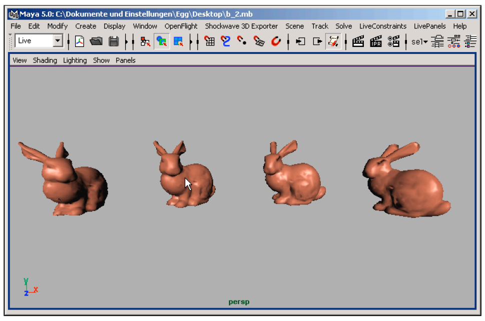
Acknowledgements

The authors wish to acknowledge the Stanford 3D scanning repository for providing the 3D bunny and dragon datasets. Furthermore, Autodesk deserves thanks for providing free evaluation versions of the applications to test the concepts proposed this paper.

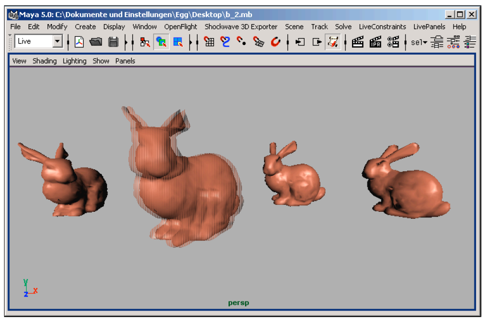
References

[1] 3DNA. http://www.3dna.net/.

[2] A. Agarawala and R. Balakrishnan. Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen. In *Proceedings*


Maya 5.0: C:\Dokumente und Einstellungen\Egg\Desktop\b_2.mb
File Edit Modify Create Display Window OpenFlight Shockwave 3D Exporter Scene Track Solve LiveConstraints LivePanels Help
Live
View Shading Lighting Show Panels
persp

(a)


Maya 5.0: C:\Dokumente und Einstellungen\Egg\Desktop\b_2.mb
File Edit Modify Create Display Window OpenFlight Shockwave 3D Exporter Scene Track Solve LiveConstraints LivePanels Help
Live
View Shading Lighting Show Panels
persp

(b)

Figure 8: 3D Maya scene with (a) four objects rendered monosopically and (b) after selecting one object this object is rendered stereoscopically.

of the SIGCHI conference on Human Factors in computing systems, pages 1283–1292, 2006.

[3] 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.

[4] R. Balakrishnan. A Grant of 3D. Keynote speach Symposium on 3D User Interfaces, 2006.

[5] 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.

[6] P. Bourke. Autostereoscopic Lenticular Images (http://local.wasp.uwa.edu.au/ pbourke/stereographics/lenticular/).

[7] P. Bourke. Calculating Stereo Pairs (http://astronomy.swin.edu.au/ pbourke/stereographics/stereorender/), 1999.

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

[9] W.-S. Chun, J. Napoli, O. S. Cossairt, R. K. Dorval, D. M. Hall, T. J. P. II, J. F. Schooler, Y. Banker, and G. E. Favalora. Spatial 3-D Infrastructure: Display-Independent Software Framework, High-Speed Rendering Electronics, and Several New Displays. In A. Woods, M. T. Bolas, J. O. Merritt, and I. E. McDowall, editors, *In Stereoscopic Displays and Virtual Reality Systems XII*, volume 5664, pages 302–312, 2005.

[10] 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.

[11] C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. Kenyon, and J. C. Hart. The CAVE, Audio Visual Experience Automatic Virtual Environment. *Communications of the ACM*, pages 64–72, June 1992.

[12] N. A. Dodgson. Autostereoscopic 3D Displays. In *Computer*, volume 38, pages 31–36, 2005.

[13] N. A. Dodgson, J. P. Moore, and S. P. Lang. Multi-View Autostereoscopic 3D Display. In International Broadcasting Convention, pages 497-502, 1999, 1999.

[14] 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.

[15] P. Kozankiewicz. Fast Algorithm for Creating Image-based Stereo Images. In *Proceedings of WSCG*, pages 59–66, 2002.

[16] 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.

[17] W. R. Mark. *Post-Rendering 3D Image Warping: Visibility, Reconstruction, and Performance for Depth-Image Warping*. PhD thesis, University of North Carolina at Chapel Hill, 1999.

[18] J. D. Mulder and R. van Liere. Enhancing Fish Tank VR. In *Proceedings of Virtual Reality*, pages 91–98. IEEE, 2000.

[19] Philips 42-3D6C01. (http://www.inition.co.uk).

[20] Project Looking Glass. (http://www.sun.com/software).

[21] 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.

[22] 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.

[23] 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.

[24] SeeReal Technologies. Next Generation Display Technology (http://www.seereal.com/en/products/nextgen/nextgen.pdf).

[25] B. Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 1997.

[26] 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.

[27] SphereXP. (http://www.spheresite.com).

[28] 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.

[29] H. Tramberend. A Display Device Abstraction for Virtual Reality Applications. In *Proceedings of Afrigraph*, pages 75–80, 2001.

[30] C. van Berkel. Touchless Display Interaction. In *SID 02 DIGEST*, 2002.

[31] C. Ware, K. Arthur, and K. Booth. Fish Tank Virtual Reality. In *Proceedings of CHI*, pages 37–42, 1993.

[32] C. Ware and K. Lowther. Selection using a One-Eyed Cursor in a Fish Tank VR Environment. *Transactions on Computer-Human Interaction*, 4(4):309–322, 1997.