Interactive tools for image-based stereoscopic artwork

Efstathios Stavrakis and Margrit Gelautz
Vienna University of Technology
Interactive Media Systems Group
Favoritenstrasse 9-11/188/2, A-1040 Vienna, Austria

ABSTRACT
In this paper we describe a set of interactive tools that we have built as an extension to our image-based stereoscopic non-photorealistic rendering system. The base system is capable of automatically turning stereoscopic input images to stereoscopic pictures that resemble artwork, including concept drawings, cartoons and paintings. The tools described here aim to complement the traditional stereoscopic viewing experience of the end-user by enabling him to interact with the perceived stereoscopic space. The observers of the generated artwork can easily enhance the perceived depth by manipulating the two artistic-looking projections while stereo viewing. The users can examine the stereoscopic artwork via the use of stereoscopic cursors, as well as explore the structure of multi-layered artwork by peeling away layers at different depths to reveal other layers, initially occluded.

Keywords: stereoscopic interactivity, stereoscopic cursors, stereoscopic non-photorealistic rendering

1. INTRODUCTION
The availability of digital stereoscopic technologies has greatly simplified the generation and visualization of stereoscopic content in both arts and sciences. On the one hand, stereoscopy has enabled scientists to better comprehend the spatial extents and interactions in some micro- and macro-scopic worlds, real or virtual, which are otherwise difficult to understand via conventional 2D images.

On the other hand, the stereoscopic medium provides an extra degree of freedom for artists who have been seeking to communicate spatiality in their works. Compared to stereoscopic photography, stereoscopic painting, drawing, illustration and even cartooning, are less known manifestations of stereoscopy in arts. In pre-digital times, the generation of handcrafted stereoscopic artworks required artists to have in-depth technical understanding of stereoscopy, good knowledge of binocular depth perception, as well as be rather accurate and meticulous craftsmen, regardless of their artistic and aesthetic capacity. Many of the technical requirements have become less crucial with the advent of digital stereoscopy, since stereoscopic systems are usually capable of tackling them while simultaneously concealing them from the end user.

Consequently, interactive stereoscopic techniques have become popular in 3D virtual and augmented reality applications. Typically, such stereoscopic systems use tracking to identify the position and orientation of the user’s head and then they utilize this information to render accurately stereoscopic image pairs of the available virtual geometric objects of the 3D scene. Users may be provided with a wide range of tools to interact with the virtual worlds and can freely move within them, however the complexity of these worlds is limited to the availability of 3D objects. As these systems are primarily based on 3D geometry, while very flexible, they may not be ideal for 2D artists (e.g. painters). In addition, obtaining and maintaining virtual and augmented reality systems is still quite costly and thus less appealing for the end-user (artist or art enthusiast).

Equivalently, advances in image-based modeling and rendering may eventually enable digital artists to use photographic images instead of 3D models as their raw material. Nevertheless, the rapidly expanding areas of automatic stereoscopic image synthesis and analysis cannot alone lead artists to adopt stereoscopy as an artistic medium, neither their audiences may find the automatically generated content satisfactory. Enhancing

Further author information:
E.S.: E-mail: stathis@ims.tuwien.ac.at
M.G.: E-mail: gelautz@ims.tuwien.ac.at
interactivity in the process of image-based artistic stereoscopic content generation, as well as visualization, may be a key to the wider use of stereoscopy in arts.

Along this line of investigation, we describe a number of interactive methods and tools that can be used within an image-based rendering system to inspect and interact with digital computer-generated stereoscopic artworks. We first provide an overview of our stereoscopic Non-Photorealistic Rendering (NPR) system; we then outline the use of horizontal image translation as a way of manipulating stereoscopic space, while preventing frame cancellation. We also describe an image-based method for rendering stereoscopic cursors and show how the method can be applied to provide a simple stereoscopic pointer, as well as a stereoscopic magnifying glass. Finally, we provide a description of the spatial arrangement of artwork generated in our system, which enables us to provide a special interactive tool that can be used to peel away layers of the artwork which are at different distances.

2. STEREOSCOPIC NON-PHOTOREALISTIC RENDERING SYSTEM

The stereoscopic Non-Photorealistic Rendering system we have built turns stereoscopic input images to stereoscopic pictures that resemble artwork. The architecture of the system can be broken down into three distinct parts: the Input, the Core and the Output Layers. These layers handle the input functionality, the algorithmic processing and the output functionality of the system, respectively. Inputs to the system are provided in the form of images, stored on disk or directly captured from video cameras. The Core Layer of the system is composed by various modules that can be used to process the inputs and provide adequate stereoscopic output. The Output Layer is responsible for providing visual feedback to the user. In addition, end-users can affect the output of the system by interacting with the graphical user interface of the various modules of the system. An overview of how the different parts of the system relate can be seen in Figure 1. The system provides general purpose functionality via graphical modules that can be used by the end-user to intuitively control and calibrate decoupled imaging devices (i.e. video cameras), rectify and undistort images and extract stereo disparity maps. Apart from these base modules, state-of-the-art algorithms for automatic generation of stereoscopic paintings, drawings and cartoons are realized separately as different modules. An overview of the stereoscopic non-photorealistic rendering pipeline used by most algorithms of our system can be seen in Figure 2.

The separation of the output functionality from the other parts of the system enables the implementation of stereoscopic interactive tools and techniques as part of the Output Layer. This design decision enables any module of the system that provides stereoscopic output to take advantage of the interactive framework, independent of its underlying algorithms.
3. MANIPULATING STEREOSCOPIC SPACE

Basic interaction with a real three-dimensional scene, using only a single stereoscopic image pair, does not require expensive and complicated to setup virtual reality equipment. Instead, standard stereoscopic graphics hardware can be used (i.e. a graphics card capable of stereoscopic rendering and a pair of active stereo shutter glasses). The level of interactivity possible is, of course, limited when compared to complete 3D systems. This can be partially compensated by inferring geometric attributes of the depicted scene objects, without however attempting to fully reconstruct and segment the scene’s objects. In our system, we use stereo matching algorithms to extract disparity maps from the input stereoscopic image pairs. These disparity maps are utilized as auxiliary data both by the NPR stereo algorithms, as well as by the interactive methods described in this section.

Two types of interactivity can be distinctly identified within our system: a) manipulation of perceived scene depth and b) inspection of the stereoscopic artwork. First, we encapsulate a set of algorithms that enable end-users to manipulate the overall perceived scene depth by changing the separation between the two images of the stereoscopic image pair, while minimizing eye-strain and preventing a common depth cue conflict arising between the binocular depth provided by the stereoscopic images and the frame of the display. Then, we describe a set of stereoscopic cursors that can be used in the image-based system to inspect various aspects of the stereoscopic artwork.

One way of manipulating stereoscopic space is to alter the overall disparity of the stereoscopic images. This can be achieved by adjusting the horizontal separation of the two image planes, which subsequently induces the eyes of the observer into a uniform vergence movement as if the viewed scene was closer or further from the user. The result of manipulating this type of disparity can be used to perceptually move the whole stereoscopic space occupied by the scene in the z-dimension relative to the observer.

3.1 Frame Cancellation

A common problem of stereo image pairs arises when binocular disparity is incompatible with an overlap cue to depth. The most common example of this is when the frame of the presentation window, within the digital display, perceptually occludes an object that has negative disparity. This means that the object’s stereo depth provides the viewer’s visual system the sensation that it is located not on or behind the display, but in front of it. As, however, the frame is always at the plane of zero parallax, which coincides with the physical surface of the display, the occlusion of the object by the frame signals to the viewer that the frame is in front of the displayed object, which is apparently true. Then the two contradicting depth cues of binocular disparity and overlap are handed down to the visual system which resolves the conflict usually by dropping the information provided by
disparity and favoring that of overlap alone, a typical case of depth cue dominance.\textsuperscript{3} This conflict is sometimes also referred to as frame cancellation and is tightly related to the breakdown of stereopsis. Various researchers have proposed algorithms that dynamically prevent frame cancellation, mainly by selecting viewpoints that do not let objects intersect with the frame around the display area.

Treating this very problem of conflicting overlap and disparity cues in stereo images and sequences, McVeigh et al.\textsuperscript{4} proposed an algorithm that applies horizontal image translation according to a statistical analysis of the disparity information present in the image pairs. Their approach, however, is to always ensure that all perceived points lie on or behind the screen, without considering the tremendous depth sensation provided by scene objects appearing between the user and the physical display. For instance, consider the case of an object’s left and right stereo projections that do not intersect the presentation frame but have negative disparities. Perceptually, bringing this object behind the display may not always be as engaging as allowing it to float in space in front of the display. In fact, features protruding the plane of zero parallax give a sense of presence in the perceived 3D environment, while scenes perceptually behind the display provide a feeling of looking through a window.

Ware et al.\textsuperscript{5} also provide insight on how to automatically adjust the parameters of a stereo display for optimal stereo viewing. Their algorithm is targeted toward virtual scenes where the scene is generated by the computer and is continuously changing. In contrast, our content is mostly static and our goal is to allow the user to intuitively adjust the parameters of the stereo system in order to bring features of his liking behind, on, or in front of the display. While very similar in concept, Ware et al. perform false-eye separation and not image shifting. Many of the ideas and results of the user studies can be taken into consideration in our case too, however, the practical implementation cannot be adapted. The main reason is that the viewpoints in our case are precalculated and fixed, while most false-eye separation algorithms strive to select the optimal separation between the viewpoints for a given scene and use it to render the two stereo projections from.

In his doctoral dissertation, Wartell\textsuperscript{6} discusses various problems that arise when using stereoscopic head-tracked displays and describes fusion control algorithms that can be used to counter them. In our work, the stereoscopic display system is based on static stimuli and not all techniques are meaningful in this context. However, of particular interest is the image shifting technique presented by Wartell, in which the two image projections are translated either closer or further away from each other in order to bring the perceived frustum at a certain distance from the observer.

Bringing these ideas together, we let the user of the stereoscopic system, described here, to interactively change the separation of the two projections by horizontally translating the two components. At the most basic form, we allow the user to provide input via the keyboard to increase or decrease the separation of the stereo images. We define horizontal image translation as a pair of translation vectors ($\vec{t}_L$ and $\vec{t}_R$), always parallel to the $x$-axis, whose magnitudes are equal, but their directions opposite ($\vec{t}_R = -\vec{t}_L$).

The two image planes are of the same size ($width \times height$) and, when viewed in our system, they are both centered at ($\frac{width}{2}$, $\frac{height}{2}$), with initial values of $\vec{t}_L = \vec{t}_R = 0$. A positive value of horizontal image translation denotes that two image planes are moving toward each other, otherwise further away.

### 3.2 Horizontal Image Translation

We found empirically that in most cases users of the stereoscopic viewing system tend to set a horizontal image translation that perceptually brings scene objects in front of the screen rather than behind it, but only if frame cancellation would not occur. Driven by this observation, instead of always positioning the two stereo images so that $|\vec{t}| = 0$, we adjust the viewing parameters of the system so that an optimal horizontal image translation is calculated by processing the available disparity information of the current image pair. Optimality is considered in regards to the initial goal of preventing frame cancellation.

The algorithm does not set all scene points behind the frame, as other techniques do. Instead it scans the border of the disparity map associated with the reference view of the stereo pair for the maximum disparity value. The width of the border can be set as a user parameter. This maximum disparity value is converted to a displacement $D_{max}$ in image space and the magnitude of the horizontal image translation vectors, which are then applied to the centers of the images, is set to $|\vec{t}| = \frac{D_{max}}{2}$. This way the pixels around the border with the highest disparity will be brought into the plane of zero parallax and therefore they will be on the surface of the
display. This allows a scene to be partially in front and behind the display at the same time, without triggering a disparity-overlap cue conflict. This is because the border scan approach ensures that no features in front of the display are also at the border, therefore objects that are inside the border may freely be either in front or behind the display, depending on their original disparity values and scene structure.

The idea behind horizontal image translation for practical stereo viewing is simple and can also be thought of in terms of a stationary viewer to whom we move the scene closer or farther. Consider a point $p$ in space and its left and right projections $p_L$ and $p_R$, respectively, as shown in Figure 3. If the initial perceived depth of $p$ is in front of the display surface, at which case $\vec{t}_L = \vec{t}_R = 0$ as shown in the leftmost component of Figure 3, then to shift this point so that it appears behind the display we can use a positive $|\vec{t}|$ and the resulting perceived depth of $p$ will be as is shown in the middle diagram of Figure 3. Finally, if instead of applying a positive horizontal image translation we applied a negative one, then the point would come even closer to the observer, as shown in the rightmost component of Figure 3.

Based on the image shifting algorithm described above, we further allow the users of our system to interactively select any pixel in the geometry of the reference view, which should be brought onto the plane of zero parallax. If the selected image pixel is $i_L$ and its disparity is $D_{i_L}$, then we apply horizontal image translation vectors with a magnitude of $|\vec{t}| = D_{i_L}$. Interactivity provided via a pointing device, in contrast to the keyboard and automatic initial adjustment of image separation, has two further shortcomings that should be carefully considered. First, as the user interacts with the resulting artwork of the system, without seeing the disparity map, the image plane of the reference view will cease to correspond with the image plane of the disparity map. Care must be taken so that the coordinates of the user’s pointer are mapped into the coordinate system of the reference image plane, so that they can be successfully used to retrieve the disparity value of the selected pixel in the disparity map$^\star$.

The second problem is related to the speed of applying the horizontal image translation vectors. Ware et al.$^5$ performed user studies which are particularly interesting in this context. They reveal that dynamic changes in eye separation should not be rapid, so that the visual system of the viewer can adapt to the new depth information without an abrupt readjustment of the human sensory facilities that could cause them to perceptually exit the virtual world. They report that a low rate of change is an acceptable speed for adjusting the separation of the viewpoints gradually. In our case we set this rate to a predefined default value, and let users adjust it. The

$^\star$This is important when rendering with OpenGL’s stereo buffers in a standard widget’s window. Most window managers will only be able to supply the user’s pointer coordinates in the coordinate system of the widget and not in the image planes’ displayed within it. Normally one should remap these coordinates to identify the pixel location the user has selected.
difference in our approach is that we are not changing the virtual eye separation, but the image separation and we
do not have any information about the visual system of the observer and the display hardware, therefore the rate
of change that is acceptable and comfortable will depend on the size of the images and the viewing configuration.
Instead of directly applying the translation vector described earlier, we perform this change over a period of time
using linear interpolation, which provides a smooth unnoticeable adjustment of the overall position of the scene
in depth.

4. STEREOSCOPIC CURSORS

Commonly used cursors in most computer applications do not provide a perceptually sound mechanism for
interaction with stereoscopic image pairs. When stereo pairs are overlaid with a standard monoscopic cursor,
which is rendered at the same position in both views, then the cursor always appears to be at the plane of
zero parallax. This is particularly inappropriate when the cursor is moved over opaque scene objects that
have negative parallax (i.e. are perceptually between the viewer and the display), because the visual system is
signaled rivalrous stimuli at the location of the cursor. In 3D systems where the stereoscopic projective geometry
is known, a stereoscopic cursor can be trivially rendered in both views with appropriate disparities and even
research attempting to identify the shape of effective stereoscopic cursors has been presented by Barham and
McAllister.7

Nevertheless, in image-based software, e.g. stereoscopic image viewers, an appropriate stereoscopic cursor
cannot be modelled directly. In addition, the goal is not to provide active manipulation of the stereoscopic
content, but only the visual appearance of the scene. One way to provide stereoscopic cursors in image-based
systems, such as ours, is to utilize dense disparity maps. In the following, we describe a simple stereoscopic
cursor equivalent to the monoscopic cursor of standard graphical desktop applications and then we show how
more advanced interactive tools, such as an image-based magnifying glass, can be designed.

4.1 Simple Stereoscopic Cursor

The simplest form of a stereoscopic cursor in our system is a target-like indicator of position, similar to the ones
commonly used in photogrammetry. We want to model the perceptual position of the cursor to always be on the
surface that is pointed at, therefore the pointer must be rendered at different locations in image space so that it
can be perceived to float in space. We generate a cursor that is a cross made out of two orthogonal lines. In the
reference view, the cursor is directly rendered with its center at the coordinates of the monoscopic cursor. The
cursor in the coordinate space of the second view is generated by rendering the cursor at the same coordinates as
in the reference view and applying a horizontal offset that is equal to the disparity value in the reference view
converted to an image space displacement. Figure 4 shows a painting with two cursors rendered with appropriate
disparities for two different points in space.

4.2 Stereoscopic Magnifying Glass

Magnification is a useful transform that allows a user to zoom into an image area and observe the finer details
present. In monoscopic images this can be done in a number of ways. A simple approach is to select the region
to be magnified and scale it by applying a linear or bilinear image interpolation. In an image-based stereoscopic
context, however, this can only be done by scaling respective image regions from both images of the stereo pair.
Scaling of respective regions alone is not sufficient either for perceptually correct magnification. The magnified
regions must be rendered with an additional disparity applied so that frame cancellation effects between the
magnifier and the surrounding scene are eliminated.

To model an appropriate stereoscopic magnifying glass that can be used directly in a stereoscopic environment,
we first define a shape (e.g. a rectangle, circle, etc.), which will act as the metaphor for the magnifying glass.
The shape is centered around the image coordinates of the user’s pointer in the left view. Just like the simple
stereoscopic cursor described in the previous section, we retrieve the disparity value of the point and generate
the coordinates of the magnifying glass in the second view. The shape is then texture mapped for each view by
a magnified texture region of the respective original left and a right textures.

For each of the vertices making up the shape of the magnifying glass, we generate texture coordinates
that are the original vertex coordinates scaled by a zoom factor $z_f \in (0,1]$. A zoom factor $z_f = 1$ produces
no magnification, while a factor close to zero produces the maximum possible magnification. The area to be
magnified in the original view is exactly defined by the shape of the magnifying glass, therefore a second user-
defined parameter $z_s$ can be used to change the size of the region to be magnified.

The disparity at the center of the magnified region in the reference view cannot be directly used to model the
magnifier in the second view, because that would inadequately allow the magnifying glass to perceptually slice
objects that are not planar. This would be the case if we used the same approach as with the simple stereoscopic
cursor, since the whole planar shape of the magnifying glass would be sufficiently displaced in the second view
to ensure that the magnifier is perceptually in front of the center, but would not account for points around
the center that have higher disparities than the center. There are various ways to remedy this shortcoming
by modifying the disparity of the magnifier’s planar surfaces. For example, it would be sufficient to find the
maximum disparity of the region to be magnified and dynamically adjust the disparity of the magnification
planes. This would have the effect of the magnifying glass to shift perceptually at different distances without
producing any conflicting cues between the magnification planes and the scene. However, we found that when
moving the magnifier rapid changes in disparity caused it to abruptly shift between depth planes in order to
remain always in front of them. A perceptually more stable alternative was to calculate the maximum disparity
of the whole scene and apply it to the magnification planes at all times, thus bringing the magnifier at a distance
from the viewer closer than any other feature in the scene. This is illustrated in Figure 5. If the disparities
of the image planes viewed in the stereoscopic system are manipulated via the algorithms described earlier for
horizontal image translation, the whole stereoscopic scene will perceptually shift from its original position. This
may introduce an undesired conflict between the perceived depth at which the magnifier appears and the scene,
therefore the respective horizontal image translation must be applied to the magnification planes too.

In Figure 6, an example stereoscopic cartoon is magnified with a circular lens. The magnification techniques
described here have very small computational payload and can be added to any real-time, image-based or not,
stereoscopic system where a dense disparity map is available.

4.3 Observing the Anatomy of Stereo Artworks
The compelling feature of stereo artworks to encapsulate depth information motivates new interactive methods.
It is not uncommon for observers of pictorial artworks to attempt to reverse engineer and “see” behind the
finished work, even by using their imagination. Observers tend to do this in order to either understand the
understructure of the work or visualize the development process. Having the ability to see a piece of art at any
stage of its creation is particularly beneficial for art education and instruction.

In monoscopic works the process of the artwork’s evolution can be captured and played back. The important
aspect is the temporal order in which the marks making up the final work are placed by the artist. Computer
Figure 5. A rectangular stereoscopic magnifying glass is shown with its center at the 3D point $c$. (a) If zero disparity is applied, the magnifying glass is rendered at the plane of zero parallax. (b) If we apply the maximum scene disparity, in this case at point $p$, to the center of the magnifier, $c$ is perceptually shifted in 3D space and all scene points will be perceived to be behind the magnifying glass.

Figure 6. (a) Left and (b) right components of a cartoon rendering with a circular magnification region.
generated content is fairly easy to record and playback, selected commercial applications for photo editing and painting (e.g. Corel Painter) provide this functionality already. Each and every step the artist takes when using a computer system, from concept sketching to finish, can easily be stored in digital form and reused. Common ways to present the evolution of monoscopic computer-generated artwork is to playback an animation of the creation process over time.

However, in stereoscopic works the temporal aspect of the artwork is not the only interesting hidden characteristic. Spatial structure beneath the surface of the artwork is significant in understanding how marks can be placed in the two stereoscopic components in order to produce an effective stereo work. Artists tend to work in the two stereo components in sequence; that is, to work on a particular area of the piece in the one component and then place similar marks on the second. The artist will then view the artwork in stereo to ensure that it provides an appropriate depth sensation and will continue this three-step iterative process to completion. It would be very tedious and mostly impractical to record the process of a stereo artist and play it back like in the case of monoscopic artwork. There would be need to shift through the monoscopic and stereoscopic modes artists themselves use in the creation process, while they are working on the two components, since the artist will work on a single component at a time. Therefore the work will exhibit binocular rivalry throughout this process, except the individual snapshots that can be taken when the artwork is being checked for consistency (i.e. there are in both the left and right components equivalent marks). Note also that marks placed by the artist between the left and right stereo components rarely have a one-to-one mapping, down to the level of e.g. individual paint strokes. This is notoriously difficult to achieve by hand, and artists will ensure the final appearance will be stereoscopically consistent and not the intermediate states. This one-to-one consistency can be guaranteed by special stereoscopic devices built for the sole purpose of stereoscopic drawing and painting, such as Tamari’s 3DD.

Our interest here is not to show the observer how the two components are made asynchronously, but to enable him to see through the artist’s eyes at certain intervals over the spatial evolution of the work. We assume that the workflow of the artist involves a stereo pair of photographs that is used as a reference to create the artwork. The artist will first create a stereoscopic concept sketch and then progressively will use brushes, from large to small, in order to apply color.

To provide an intuitive user interface for viewing these stages and exploring the evolution of the painting over the z-dimension, we provide an orthographic projection of the whole stereo painting’s structure to the viewer. At the furthest point from the viewer, the stereo photographic components, which were the input to our system, are placed so that their planes are perpendicular to the viewing direction. In front of these images a concept stereo sketch generated by our system is positioned. Since an orthographic projection is used to construct the viewing volume within which the observer is looking at the artwork, features will appear to have a constant size and will be perceived at the same depth (monocularly) regardless of how close to the viewpoint they are rendered. Incrementally, from back-to-front and coarse-to-fine, each paint layer that was calculated by the stroke-based painting algorithm is placed within the volume. The volume will then be sliced using a clipping plane in order to allow the viewer to see behind the finer strokes that occlude rougher strokes which are positioned further at the back.

The viewing volume used to render the stereo artwork within is of $W \times H \times D$ dimensions and the eyes are at the plane where $z = 0$. The $W \times H$ dimensions of the frustum correspond to the size of the stereoscopic images used throughout the algorithmic processing. Recall that each stroke of an individual layer has an associated depth value $d_S$ that is guaranteed to be within a range of $[0, 255]$, with 255 being closer to the viewpoint. However, strokes usually fall within a smaller range. We scale this subrange linearly so that depth values of strokes occupy the whole $[0, 255]$ range. This helps to increase the distance between strokes, because many scenes have a limited amount of depth and therefore strokes at different depths are usually densely distributed inside a smaller range of depth values. The motivation for this scaling is to increase the z-resolution of the paintings and allow the clipping plane to travel more distance between the strokes of an individual layer, providing easier control of the clipping plane to the end-user.

The overall depth $D$ of the volume is calculated according to the amount of layers that are to be displayed within it and an additional offset value for spacing the layers can also be applied. If there are $N$ layers then $D = N(256 + \text{offset})$. This ensures that the viewing frustum is sufficiently sized to accommodate all layers. If
the roughest layer is considered to have an index $n = 0$ and for any refining layers $n$ is incremented, then the overall z-position of each stroke $S_Z$ of each layer can be trivially calculated as $S_Z = (256 + ofset)(N - n) - dS$. If we have not spaced apart the refining layers, then their differently sized strokes would be at the same distance from the viewer and therefore it would not be possible to use 3D clipping to toggle the visibility of individual strokes, hence slicing would not be intuitive.

For instance consider the stereo image pair shown in Figures 7(b) and Figures 7(c). When painted using three painting layers ($N = 3$), the stereo components are arranged as shown in Figure 7(a). The clipping plane provided to the user can be interactively shifted along the z-axis. Strokes that are between the viewpoint and the clipping plane are hidden, allowing the rougher layers to be seen. A slice through the painting can be seen in Figure 7(d) and 7(e). Clipping is equivalently performed in both the left and right stereo components and, since at every depth plane the visibility of the stereo primitives is consistently toggled, a stereo impression of the understructure can be perceived.

5. CONCLUSION

We have described a set of interactive methods over which we have built image-based tools as part of our stereoscopic Non-Photorealistic Rendering system. The interaction methods allow a user to manipulate the stereoscopic space within which the artwork is viewed, by interacting only with the two projections of the stereoscopic artwork generated by the rendering algorithms of the system. The interactive graphical user interface is further enriched with a simple stereoscopic cursor that can be used to inspect relative depth of the stereoscopic renditions, as well as a stereoscopic magnifying glass which enables users to enlarge and inspect specific regions of interest. Finally, a specialized tool is provided to slice through the various depth layers present in multi-layer stereo artwork in order to reveal occluded detail.

The collection of these tools allows the user to explore the stereoscopic space occupied by the stereoscopic artwork within the stereoscopic space itself. By combining interaction techniques of both modes, the artwork can be decomposed or explored by the spectator, while simultaneously maintaining an eye-strain-free stereo display, which further enhances the viewing experience.

ACKNOWLEDGMENTS

This work was supported by the Austrian Science Fund under projects P15663 and P19797.

REFERENCES

Figure 7. (a) A third-person view of the left component of a stereo painting with three layers. The user looks from the eye position. A consistent right view is calculated for the other eye. (b) and (c) show the input left and right stereo images, while (d) and (e) show the left and right components of the sliced painting at a user defined section, as this is visible from the user’s point of view.