Collaborative Interaction with Volumetric Displays

Tovi Grossman, Ravin Balakrishnan
Department of Computer Science
University of Toronto
www.dgp.toronto.edu
{tovi, ravin}@dgp.toronto.edu

ABSTRACT
Volumetric displays possess a number of unique properties which potentially make them particularly suitable for collaborative 3D applications. Because such displays have only recently become available, interaction techniques for collaborative usage have yet to be explored. In this paper, we initiate this exploration. We present a prototype collaborative 3D model viewing application, which served as a platform for our explorations. We outline three design goals, discuss the key interaction issues which were encountered, and describe a suite of new techniques in detail. In initial user observation sessions, we found that our techniques allowed users to successfully complete a variety of 3D tasks. Furthermore, interviews with experts in potential usage domains indicated that the techniques we developed can serve as a baseline for future collaborative applications for volumetric displays.

Author Keywords
Volumetric displays, collaborative interaction.

ACM Classification Keywords
H.5.2 [User Interfaces]: Interaction styles.

INTRODUCTION

Volumetric displays are a relatively new platform for displaying three-dimensional (3D) imagery [7]. These displays possess several interesting and unique properties in comparison to other 3D viewing technologies, which potentially make them particularly suitable for collaborative 3D applications [3]. An important advantage is that users do not have to wear any supplementary hardware which might hamper collaboration efforts. Further, the displays typically have a 360° field of view, allowing multiple users to work with the imagery from anywhere around the display. Despite these unique affordances, we are unaware of a thorough exploration looking at the interaction issues involved in making the display a collaboration platform.

We envision scenarios where a volumetric display could be utilized in a collaborative setting, which motivate such an exploration. As an example, a group of students could examine a virtual model of an anatomy specimen which they were studying in a laboratory. The students could work together to identify critical areas of the structure, label areas which may be anomalous, and perform various browsing operations to reveal hidden features. Carrying out such tasks on a volumetric display, while leveraging its unique features, could potentially improve the users' understanding of the 3D data [12, 19].

Before developing applications for such scenarios, it would be useful to first obtain a base understanding of the associated interaction design considerations. Previous work in volumetric display user interfaces for single users will provide guidance [13], however, the design of collaborative applications will likely raise new issues, which merits its specific investigation. In this paper, we initiate this exploration by developing a prototype collaborative 3D model viewing application that served as a platform for our interaction designs (Figure 1). The prototype allows multiple users to inspect, markup, and manipulate 3D scenes. The interaction techniques which we implemented were designed such that they would be applicable across various application tasks and usage domains. Results from initial usage observation sessions and interviews with application domain experts indicate that these interaction techniques can serve as a baseline for the design of future collaborative applications for volumetric displays.

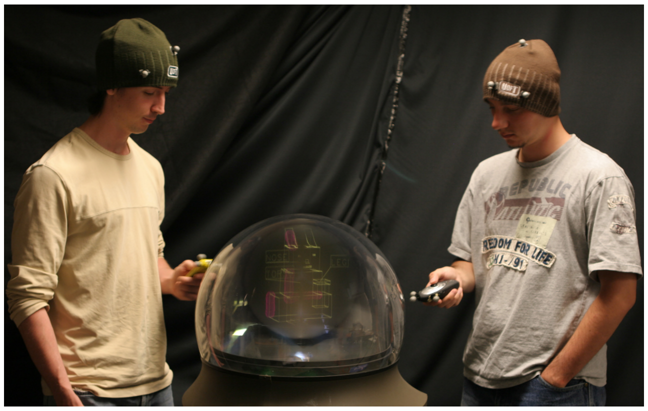

Figure 1. The prototype for our explorations allowed users to collaboratively interact with the volumetric display. Users' viewing positions and input devices were tracked in 3D.

BACKGROUND

In this section, we discuss the related literature which will guide our own work. We begin by reviewing previous work done with volumetric displays, followed by a discussion on the research done in single display groupware.

Interaction with Volumetric Displays

Volumetric displays are unique in that they provide imagery in true 3D space. Studies have shown that as a result, they can improve depth perception [12] and shape recognition [19]. However, since they have only recently become commercially available, little work has been done to explore interaction issues unique to them. Using physical mockups and wizard of oz prototypes, Balakrishnan et al. presented potential interaction scenarios [3]. While they did discuss the potential for collaborative use, the majority of the usage scenarios and interactions which they discuss are focused on a single user.

More recently, an interactive geometric model building application was developed for volumetric displays [13]. A user could provide input by performing hand gestures directly on and above the display surface. The application explored a myriad of interface operations, but for a single user only, so it is unclear how well those techniques would work in a collaborative setting.

Anticipating the interest of volumetric displays as a collaborative platform, Grossman et al. performed a study analyzing the effects of text orientation in volumetric displays [14]. Based on their empirical findings they introduced and validated an algorithm to optimize text orientation when the text is being viewed by multiple users. However, their algorithm was never implemented outside of the experimental setting. In the current work we implement their proposed algorithm within a prototype application.

Single Display Groupware

A branch of computer supported collaborative work which has received recent attention is single display groupware (SDG) - infrastructure which supports collocated groups interacting with a shared display [22]. A defining element of SDG is that users can interact simultaneously with the display, using their own input devices.

Early research in SDG began with the MMM system [4], which allowed multiple users to simultaneously interact with several common interface elements, such as menus and text editors. Stewart et al. [22] discuss three central properties of SDG which introduce new difficulties:

Shared User Interface

Interface elements must be accessible and able to handle simultaneous input from all users. This can be especially problematic in direct touch systems, as there may not be a central location that users can reach to access interface elements. For volumetric displays, this means that placing direct touch widgets on the display surface [13] may not be appropriate when multiple users are present.

One solution is to use popup menus and widgets, which can always be accessed regardless of user locations [21, 27]. An alternative solution is to use a non-direct input device. However, it can be challenging to define a control display mapping for non-direct input when users are interacting from various viewpoints [26]. Absolute mappings, which have been previously used to control volumetric display cursors [11], will not be appropriate, as the user position may diverge from the absolute control space. As such, our work will examine other possibilities.

Shared Feedback

SDG Systems must have the ability to communicate information to multiple users simultaneously, and also to individual users. If the users are working from various positions around the display, then simultaneously presenting information can be problematic as it can suffer from orientation effects [25]. With volumetric displays, this will be particularly problematic, as what appears forwards to one user could appear backwards to another user [14]. Presenting information to individual users is also problematic, as it can cause interference to other users for whom the information is not directed [28].

Coupled Navigation

When a single user navigates to a different area of the application data, other users will either also be forced to navigate simultaneously, which may be unexpected or unwanted, or have their views obscured by the one user who is navigating. This problem generalizes to any interaction which can result in conflicts when carried out simultaneously, or when unwanted by certain users. Greenberg et al. categorize such issues as concurrency control problems [9]. Possible approaches to this problem are to use locking mechanisms [9], coordination policies [16], or to rely on social protocols [9] to prevent conflicting actions. We anticipate that these approaches can be used for volumetric displays.

While most SDG research has been conducted with more widely available 2D technologies, some has been conducted in the 3D realm. Maybe most relevant to our work is the Two-User Responsive Workbench [2], which allows two users to stand around a physical table and interact with a 3D image. The users wear stereo shutter glasses, which interleave different images for each user, allowing each user to have an individualized 3D view of the scene. This work focused on implementation details and specialized views. Our work will explore new interaction techniques.

In summary, there exists a solid groundwork of research in single display groupware, but fewer results which are relevant to the design interactive applications for volumetric displays. Furthermore, due to the unique properties of volumetric displays, developing a collaborative application is not as trivial as generalizing existing SDG research. New interaction techniques specific to volumetric displays will need to be investigated.

DESIGN GOALS

Our review of the SDG literature, in addition with our own observational evidence, indicate that new interaction techniques are required to address the issues associated with collaborative use of volumetric displays. The following are three design goals which we have identified as having particular importance in developing such techniques.

Location Sensitive Interaction

Since users can stand anywhere around a volumetric display, the user interface should be accessible from any location, or be "omnidirectional". A similar design goal has been followed for tabletop displays [21], which also have a 360° viewing angle. However, users of volumetric displays may be standing and walking around the display, so discrete seating locations cannot be assumed, as they commonly are in tabletop applications [20, 21, 27]. Furthermore, since interaction cannot be truly direct, as users typically cannot reach into the display, a user at the front of the display may want to interact with imagery at the back of the display. Systems, therefore, should not make territorial assumptions, such as correlating display areas with viewing positions [20]. Thus, orientating data and widgets to the closest possible viewing location [21], may not be appropriate. Due to these additional challenges, we will explore "omnidirectional" interactions techniques, which can be used from anywhere around the display, but also leverage knowledge of the user's viewing locations.

Parallel Access

Providing parallel access is a recognized design goal in SDG applications [22]. Since we wish multiple users to be able to work with the display simultaneously, the user interface should be accessible to all users at all times, and interaction techniques should be able to be carried out in parallel. This design goal has particular relvance to navigation. As discussed previously, *coupled navigation* is a central difficulty in SDG applications [22]. In volumetric displays, the problem is increased since navigation is one of the core interactions in 3D applications [6].

Inter-user Understanding

The last design goal is that users should have an awareness and understanding of what other users are doing. This property has been identified previously and is generally addressed with simple techniques such as cursor coloring [4]. However, the unique properties of volumetric displays make this design goal particularly interesting. Volumetric displays provide data in true 3D space, so it can be difficult to understand what another user is seeing, since when viewing 3D data, the viewpoint can impact how the data is perceived. Furthermore, because users cannot directly reach in and touch the data, it will be difficult to point to an area or indicate an area of interest without some form of virtual aid. We commonly observe such difficulties when multiple users view and try to discuss static imagery on the volumetric display. As such, we will explore technological enhancements to facilitate awareness between users.

EXPLORATION PLATFORM

We developed an interactive 3D model viewing prototype to serve as a platform for our explorations. Users can view, inspect, label, markup, and modify 3D models in parallel. We chose this example application as it is general enough such that our interaction techniques could be applicable in a range of usage domains. Similar to previous prototype SDG applications, the application was developed to support two users [2, 27]. However, the majority of the interactions directly generalize to an arbitrary number of users.

IMPLEMENTATION DETAILS

Display Device

We used a 3D volumetric display developed by Actuality Systems [8], which generates a 10" spherical 3D volumetric image by sweeping a semi-transparent 2D image plane around the Z (vertical) axis (Figure 2). A total of 198 2D images (slices) of 768x768 pixels each are displayed around the Z-axis. The display's refresh rate is 24Hz.

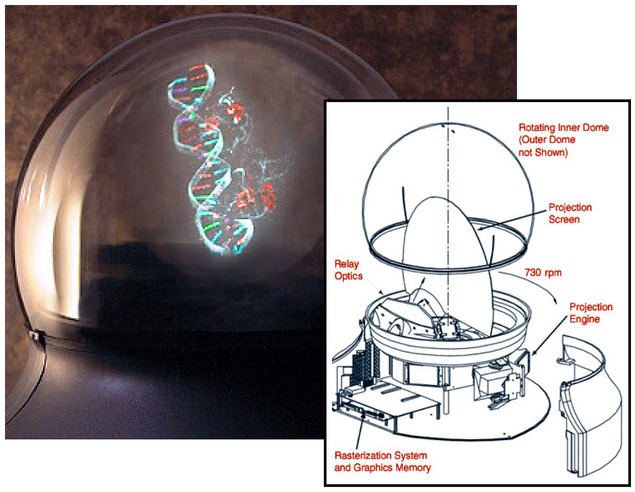
Rotating Inner Dome (Outer Dome not Shown)
Projection Screen
Relay Optics
730 rpm
Projection Engine
Rasterization System and Graphics Memory

Figure 2. Actuality System's volumetric display.

User Tracking and Input

Users stood and were free to walk around the display. We used a Vicon motion tracking system (www.vicon.com) to track the positions of the viewers' heads. Three passive reflective markers were placed on a hat which users wore. Each user held their own 3D input device, consisting of a wireless presentation mouse, augmented with 3 Vicon markers (Figure 1). The devices had a left and right button which could be used for our interaction techniques. Six Vicon cameras tracked the 3D location of the markers, and the data was streamed into our application at 120Hz. This hardware setup should be viewed simply as an enabling technology for our exploration, rather than one that would be used in any future real implementation. Two keyboards were placed on opposite sides of the display which allowed for concurrent input from each user.

Software

The software was programmed in C++ using a modified OpenGL library for the volumetric display. The application ran on a Pentium 4 PC running at 2 GHz.

INTERACTION TECHNIQUES

The interaction techniques which we have implemented can be categorized as follows: interface controls, navigation, markup and manipulation, and advanced interactions. While our focus is on novel techniques specifically developed for collaborative use, their implementations are integrated with previously developed 3D interaction techniques. This allows us to build a seamless and working user interface.

Interface Controls

Users can change tools, execute commands, and change system options through a 3D radial menu and an options dialog box. In this section we will also described the two types of cursors which are used for interface control: the depth ray and the 3D point cursor.

3D Radial Menu

In a previous volumetric display application, menus existed on the surface of the display, and were used through direct touch [13]. This could cause an accessibility problem [22] when there are multiple users, since there may not be a global location that all users could reach. Instead, similar to previous SDG applications, we use a popup menu [21, 27, 28]. To use the menu, the user presses the right button of their input device. Users each have their own menu, so it supports *parallel access*. The menu is displayed in a visible location of the display volume, and is oriented towards the user who activated it, so it is also *location sensitive*.

Since the display volume is 3D we felt it would be worthwhile to explore a *3D radial menu*. So, unlike traditional radial menus, the menu items, which consist of tools and commands, are distributed spherically in 3D space. A similar idea has been explored in 3D virtual environments [10]. Extending the menu to 3D increases the angle between menu items, which, in theory, reduces the motor constraints imposed on the user to select an item [1], and could potentially increase the efficiency of the menu.

Once the menu is activated, a 3D crosshair is displayed at its center, and the user controls it via a direct one-to-one mapping from the input device. The user moves towards the desired menu item and releases the button, or can release the button while still in the center area to cancel the menu.

Options Dialog Box

The *options dialog box* is used to control global parameters. It is activated through the 3D radial menu. The dialog box is a simplified version of a traditional GUI dialog box. Like the 3D radial menu, it is oriented towards the user, but it is displayed on a 2D plane. A standard cursor, controlled by the user's input device with a direct mapping, is projected onto the plane of the dialog box. The options dialog box is similar to the "pen & tablet" interaction metaphor used in immersive virtual reality environments [6].

Users can click on various options to toggle values, and click the exit option to exit the dialog box. As with the 3D radial menu, users can access their dialog boxes in parallel.

Depth Ray

The *depth ray* is used as the selection tool for the system. The depth ray is based on the ray casting metaphor [15], and has been previously found to be an efficient selection mechanism for volumetric displays [11]. It consists of a virtual ray emitted from the input device, and a depth marker, which can be moved forwards and backwards along the length of the ray. The depth marker is used to disambiguate when multiple selectable items are intersected by the ray. The intersected item closest to the depth marker is highlighted, and can be selected with a left click.

Each user can use their depth ray at the same time, so it supports *parallel access*. However an object highlighted by one depth ray is ignored by the other depth ray, which prevents both users from selecting the same object. Users can also use the depth ray to highlight objects without selecting them, to indicate a feature to the other user. Thus, the depth ray also supports *inter-user understanding*.

3D Cursor

The depth ray is an efficient mechanism for selection. However, for tools that require 3D positioning, we use a *3D cursor* [15], which is more appropriate for such tasks [5]. The cursor is rendered as either a sphere or a 3D crosshair, depending on the current tool being used.

For multiple users whose positions are not fixed, a strict absolute mapping, which has been previously used in single user scenarios [11], will not work. To ensure that the 3D cursor, and tools which rely on it, would be omnidirectional, we looked at alternative mappings. A relative mapping would be difficult to implement, since there is no obvious clutching mechanism for a device which is held in midair. Instead, we define a dynamic absolute mapping which is relative to the user's location. As such, the 3D cursor supports *location sensitive interaction*. Similar mappings are used in 3D virtual environments for hand-extension techniques, such as the go-go technique [18]. However, they are based on polar coordinates. Our technique uses Cartesian coordinates.

A default vector V is used to define the offset between the user's location, L, and location where the input device would map the cursor to the origin of the display volume. The mapping uses V', which is V rotated by the users viewing angle, so that the offset is appropriate regardless of the users position around the display. The cursor is controlled with a control-display gain of 1. Thus, the cursor position, C, is defined as:

$$C = D - (L + V'),$$

where D is the position of the input device. To prevent slight movements of the user's head from changing the cursor location, we only update L when the cursor leaves the display volume. Thus, the absolute mapping is only updated when necessary, and the update is invisible to the user, since it happens when the cursor cannot be seen. Since each user may prefer to hold the input device in a different position, we allow users to manually set the vector *V*, using a simple calibration procedure. This procedure would only need to be carried out once at the beginning of use.

Navigation

One of the benefits of viewing a virtual 3D model over a physical one is that users can easily inspect different areas of the model through navigation techniques that may not be possible in the physical world. Indeed, navigation has been identified as a "universal interaction task" for 3D environments [6]. Furthermore, navigation is particularly important for us to explore within a collaborative setting, because of the *coupled navigation* issue which could arise. To mitigate the effects of coupled navigation, as soon as one user begins any type of navigation operation, all other navigation operations are locked out, until the initial operation is completed. This does not eliminate potential problems, as one user may still navigate while the other user is in the middle of viewing something. By implementing the following navigation tools, we were able to explore such effects.

Location Aware Rotation

Since users can stand anywhere around the volumetric display, each user may have very different perspectives of the displayed imagery. Our rotation tool allows one user to see what another user is seeing, without having to physically walk around the display.

When users place their input device close the bottom portion of the display surface, a rotation widget fades in which indicates that a rotation can begin (Figure 3a). Clicking and holding the right button, while scrubbing either left or right, rotates the scene either left or right around the Z-axis. The rotation is spring loaded - if the user releases the button, the scene animates back to the original rotation. This allows users to take a quick glance of the model from another viewpoint, much like the "Glances" navigation tool [17]. Alternatively, users can pin the new rotation by sliding the input device upwards.

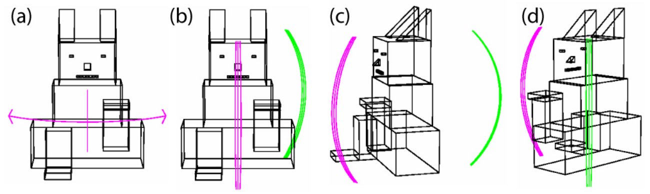

Figure 3. Location aware rotation. a) A widget is displayed indicating that a rotation can begin. b) When a rotation begins, viewpoint widgets indicate user viewpoints. c) Viewpoint widgets rotate with the scene. d) One user can align and snap to another user's viewpoint.

While rotating, viewpoint widgets are displayed for each user, indicating where their viewpoints were in relation to the scene before the rotation began. When a rotation begins these widgets are positioned directly between the user and the center of the display (Figure 3b), but the widgets rotate with the scene (Figure 3c). Users can see what another user's viewpoint was by continuing to rotate until that other user's viewpoint widget is aligned with their own viewing location (Figure 3d). The rotation will snap to this aligned location. Similarly, users can pass their viewpoint to the other user by rotating until their viewpoint widget is aligned and snaps to the other user's viewing location. This "viewpoint passing" interaction is *location sensitive* and supports our *inter-user understanding* design goal.

Panning

To pan the scene the user selects the panning tool. Once selected, a translation widget is rendered in the center of the display. Clicking the left button begins a pan, which is directly controlled by the position of the input device.

Zooming In

The *zoom-in tool* can be used to obtain a more detailed view of a certain area. The zoom-in tool utilizes the 3D cursor, which is displayed as a sphere (Figure 4a). A user can control the size of the sphere by twisting the input device left and right to decrease and increase the radius. The sphere acts as a preview to the new viewing volume; the smaller the sphere the higher the zoom level. Clicking the left button animates the scene to the new zoom level (Figure 4b). This is similar to the traditional marquee zooming, however, the contents of the sphere indicate to the user what the new view will be before committing to the zoom. The other user can also see this preview, with the opportunity to comment or provide instructions. Thus the zoom-in tool supports *inter-user understanding.*

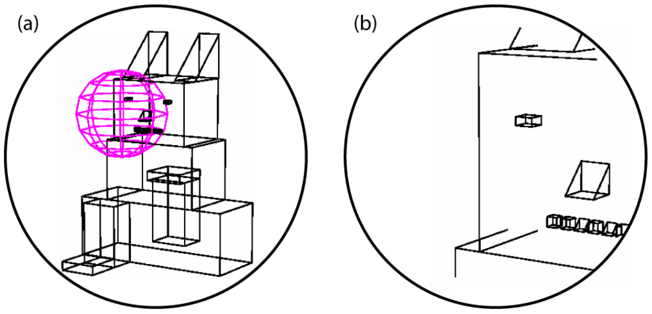

Figure 4. Zoom-In Tool. a) The user controls the position and radius of a spherical 3D cursor. b) Clicking the left button zooms in to the area represented by the sphere. Circles represent the display volume.

Zooming Out

To zoom out the user selects the *zoom-out tool* from the radial menu. When selected, a zooming widget is displayed in the center of the viewing volume. The user can hold the left button down and pull away from the display surface to decreases the zoom level. The zoom is always centered at the origin of the coordinate system, such that once the default minimum zoom level is reached, the model will be centered inside the viewing volume.

Markup and Manipulation

Regardless of the usage domain, users will likely want some way to highlight, and possibly even modify, areas of the displayed data. The ability to markup data is important for *inter-user understanding*, since it allows users to indicate areas of interest to one another. We explored the following markup and manipulation tools.

Highlighting Tool

Since users cannot reach into the virtual image, it can be particularly difficult for one user to indicate an area of interest to another user. The *highlighting tool* allows users to highlight 3D areas, so that they can explicitly define areas of interest to other users. Thus, the tool supports *inter-user understanding.* The tool is an extension to the 3D annotation tool developed by Tsang et al. [23], which only allowed 2D annotations projected onto 3D geometry. The highlighting tool is controlled by the 3D cursor, so it is *location sensitive.* It is rendered as a spherical cloud of points (Figure 5a). The user can press and hold the left button to sweep out a freeform 3D area, or "highlight cloud" (Figure 5b). Only exterior points of the cloud are displayed, so that the highlighted region consists of an outer surface of points. Like the zoom-in tool, the radius of the cursor can be modified by twisting the input device, allowing users to sweep out coarse or detailed regions.

This tool can support *inter-user understanding* without adding a highlight cloud. A user could use the cursor to indicate an area of interest to another user. To aid in this usage scenario, we provide volumetric magic lens [24] functionality to the highlighting tool cursor, which is activated through the options dialog box. When active, all elements of the scene within the bounds of the spherical cursor appear as a different color (Figure 5c). This magic lens functionality can also be used collaboratively - if both users have their cursor in the same area, the elements of the model within the intersection of the two cursors appear much brighter (Figure 5d). While the composition of two volumetric lenses has been previously explored [24], our implementation is in a true 3D volume, and allows separate users to each control a lens in parallel. The magic lens can also be set to cull out anything within its bounds

The highlighting tool supports *parallel access* as both users can create highlight clouds in parallel without any constraints. The color of any created highlight cloud matches the cursor color of the user who created it.

Text Flags

In addition to marking up a model with the highlighting tool, users may also want to label certain areas of interest. While 3D annotations have been previously explored [23], new issues arise in a collaborative volumetric display scenario. Specifically, the label can be positioned in true 3D space, and the label must be readable by all users.

To add a label, the user selects the *text flag tool* from the 3D radial menu. The text flag tool uses the 3D cursor, which is rendered as a crosshair. The cursor is used to create text flags, which consist of a line with a rectangle containing text at one end (Figure 6). The other end of the line is considered the origin, which is the 3D location being labeled. If the user wishes to label a general area, rather than a specific location, then the text flag tool could be used in combination with the highlighting tool.

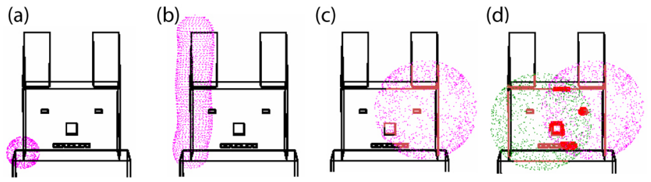

Figure 5. Highlighting tool. a) The highlighting tool is displayed as a spherical cloud of points. b) Holding down the left button adds a new highlight cloud. c) The highlighting tool can be used as a magic lens. d) The magic lens effect is amplified inside the intersection of both users' tools.

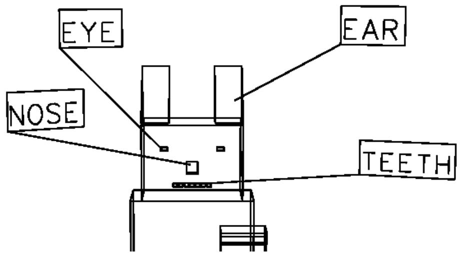

Figure 6. Text flags can be used to label areas of a model.

To create the flag the user positions the cursor at the 3D location of an area which they wish to label. Clicking the left button sets the origin of the flag. While the left button is still down, the user can position the other end of the flag. The user can then type on their keyboard to add the text.

A challenge with presenting text on the volumetric display, when multiple users are present, is orienting it so that it is readable [14]. We provide three possible modes of display for the text flags which address this orientation issue. This mode is set in the options dialog box.

The "optimized mode" optimizes the orientation of the flag to minimize the reading time for *both* users, so it supports *parallel access*. We use Grossman et al.'s optimization algorithm [14], which is based on the real-time location of the users, and so it is *location sensitive*. Users can override this algorithm by placing their cursor close to the text flag, causing it to temporarily orient towards that user.

The "rotate" mode causes all text flags to slowly rotate about the Z-axis. This gives each user a chance to see the text from the optimal orientation.

The last mode is a "privacy" mode. With this mode, the rotation is set such that the text is hidden from the user that did not create it, by keeping the text parallel to the other user's line of sight. Such private viewing has been previously suggested for SDG applications [2, 27]. This mode works best if the two users are standing at 90 degrees from each other, since the text will be facing the user who created it, and parallel to the other user. The mode does not work well if the users are at exact opposite sides, since the text will be parallel and thus hidden from both users.

We implemented a simple layout algorithm to update the positions of the text flags during use of the navigation tools. The algorithm guarantees that if the origin of a flag is in the viewing volume, then the text portion of the flag will also be visible. If the origin of the text flag is not in the viewing volume, then the text flag is not displayed.

As with the highlighting tool, the text flag tool supports *parallel access* as it can be used in parallel by both users. Text flags, can be selected and deleted by the depth ray, via a contextual popup menu. Once selected, the text within the flag can be edited, and the location of its endpoints can be modified. When a user selects a text flag, the text orients towards that user, regardless of the viewing mode.

Object Manipulation

While our prototype application was built around the scenario of *viewing* a 3D scene, there may be cases where users wish to *manipulate* the data. For example, when a team of car designers are viewing a car, one designer may want to demonstrate an idea to slightly change the curvature of the front hood. The availability of a manipulation tool could thus support *inter-user understanding*.

Manipulations are carried out with the depth ray, which can be used to select faces of the model (Figure 7a). Once selected, the user can drag the face along its normal vector by moving the input device (Figure 7b). More elegant modification tools can be imagined, but are beyond the scope of our work. Along with supporting *inter-user understanding,* these manipulations supports *parallel access,* as users can both modify the scene in parallel.

Advanced Features

Scene Splitting

We have described a number of operations, such as highlighting, labeling, and manipulating, which support *parallel access.* However, if users wish to work on areas of a scene which are too far apart, or require different zoom levels, then they may not be able to carry out the tasks in parallel. To support *parallel access* under such scenarios, we implemented *scene splitting.*

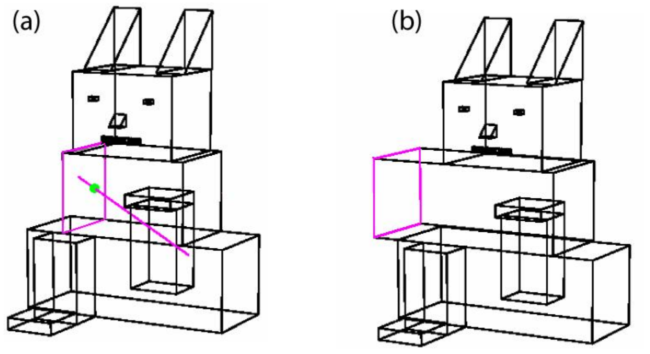

Figure 7. The depth ray (a) can select and drag faces (b).

Scene splitting divides the viewing volume into multiple viewports. This idea has been previously suggested but never implemented [3]. Users split the scene with a slicing gesture across the surface of the display with their input device. A 2D dividing plane rendered as a grid is drawn across the display volume, which divides the volume into the two viewports, one for each user (Figure 8).

Once the scene is split, users can carry out all of the previously described operations on their side of the display, with no effect on the other user's viewport. However, if one user is zoomed in, the location which they are working is indicated to the other user by displaying a sphere on the other's scene with the appropriate location and size. This feature supports *inter-user understanding*. To minimize distraction, text flags and highlight clouds which are added by one user do not appear in the other user's scene. However, any manipulations of faces are reflected immediately in the other user's scene to eliminate the need of conflict management techniques [9].

To return to a single viewport, the scenes can be merged by dragging the input device along the surface of the display from one side of the dividing plane to the other. When the scene is merged all text flags and highlight clouds that were added by either user while the scene was split are shown.

Hidden Surface Removal

One general limitation of almost all volumetric displays is that they are incapable of exhibiting occlusion of one part of the image volume by another (Figure 9a). This is because the light which illuminates a voxel is omnidirectional [7]. However, if an application is *location sensitive*, then surfaces which are behind other objects based on the user's viewpoint can be manually hidden. Unfortunately, this can only be done for a single user - since a surface which is hidden from one user may be visible to another. To compensate for this, we support three modes of hidden surface removal, which are set from the options dialog box.

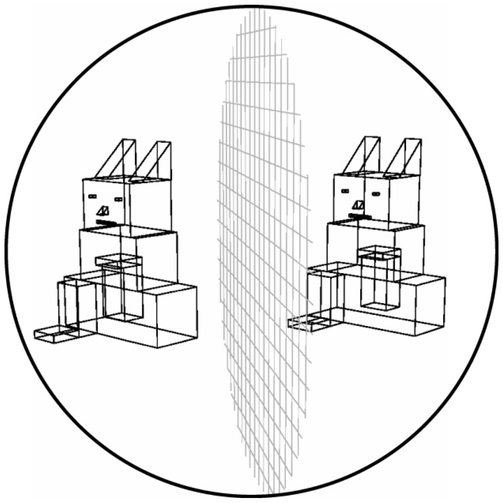

Figure 8. When the scene is split a grid divides the display space into two viewports, one for each user. The circle represents the display volume.

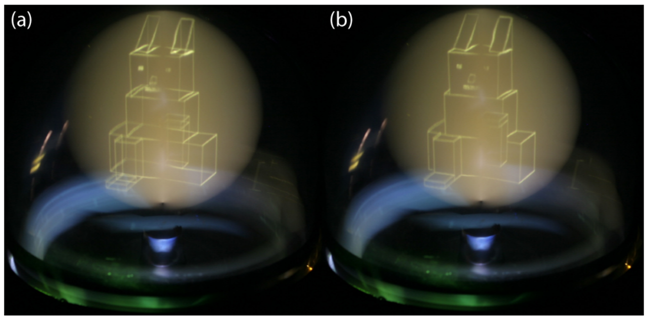

Figure 9. a) Typically volumetric displays are incapable of hidden surface removal. b) The same scene displayed using our hidden surface removal algorithm, which is updated in real-time based on the user's viewpoint. For illustration in this figure, the algorithm is based on the location of the camera.

The first two modes correspond to activating hidden surface removal relative to either the first or second user's viewpoint. Our algorithm clips all lines which are not visible based on the location of the midpoint between the user's eyes (Figure 9b). The location of the eyes is estimated using a default vector from the location of the user's hat. If more precision is desired, the exact offset between the eyes and the user's hat can be determined using a short calibration program. While these two modes will hide surfaces which the other user should be seeing, it could still be useful if the users want to take turns to see a more realistic view of the scene.

The third option is a merged hidden surface removal rendering of the scene. This option will render the union of what both users can see from their viewpoints. The effectiveness of this mode depends on the relative location of the users. Alternatively if both users wanted to see the scene with accurate hidden surface removal, they could split the scene to get their own views.

INITIAL USER OBSERVATION SESSIONS

We conducted three observation sessions, each with two volunteer users. Participants were members of our lab, and were familiar with the volumetric display, but had no previous experience interacting with it. A think-aloud protocol was used, and sessions lasted approximately one hour. Sessions began with a 15 minute instruction period, where participants tried out all of the features of the system. After the instruction period, an abstract model (Figure 9) was displayed and the two users were asked to carry out 4 tasks, of approximately 5-10 minutes each. In the first task users worked together to label 10 features of the model. In the second task each user was given a different list of features to label, requiring *parallel access.* In the third task one user indicated areas of interest for the other user to label, requiring *inter-user understanding.* The fourth task was meant to evaluate the various interaction techniques in combination: users were given a diagram which had 10 differences from the displayed 3D model. Users were asked to identify these 10 differences, and to then correct the differences on the virtual model.

Observations

Users were able to use most features of the system without difficulty after short instruction, and users were able to complete all tasks. The final task, in particular, went well, as participants successfully used a combination of the rotation, navigation, highlighting, and text flag tools to complete the task. Users made comments about specifically liking the optimized text flag rotation, the ability to manipulate models in parallel, and the hidden surface removal. More detailed observations in terms of our design goals, and encountered difficulties, are now discussed.

Location Sensitive Interaction

An interesting overall observation was that the groups used a combination of both the virtual rotation functionality, and "physical" rotation (walking right around the display). In several instances, users walked right around one another. The fact that users were able to use the techniques and complete the tasks while physically walking around the display indicates that our techniques successful supported *location sensitive interaction*. The users seemed to find the dynamic absolute 3D cursor mapping easy to work with, and liked how the orientation of the menus and text flags were updated based on their locations.

Parallel Access

Overall, users worked well in parallel. This was especially true in the second task, where users worked completely independent of one another. The interface elements for each user did not interfere with one another, except for a few instances when both users had their menus up at the same time. Users did comment that they were sometimes reluctant to perform virtual navigations as they did not want to change the viewpoint when the other user was doing something. Users often verbally discussed navigations before executing them. For example "let's zoom in again", or "we'll do the teeth last, because we have to zoom in".

Inter-User Understanding

The users took advantage of the tools which were provided to support *inter-user understanding*. In some cases users explicitly switched to the highlighting tool to point out areas to each other, but at other times they just used the location of their current tool cursors, to avoid an explicit tool change. One participant used the depth ray to indicate faces to the other user, because the depth ray could highlight an entire face. During the third task in one of the sessions, one user would highlight an area, pass his view to the other user, and then the other user would label it. The fact that the tasks were completed successfully indicates that our techniques did support communication and awareness between users.

Interface Limitations and Difficulties

A number of users found it hard to use the zoom-in tool. This was in part due to difficulties in twisting to change the zoom level, but was also because users were used to a marquee zooming tool. Furthermore, users did not like having separate tools for zooming in and out. In response to this, we iteratively redesigned the zoom out tool such that the user could position the 3D cursor to set the center of the zoom, and then click the left button and move the device forwards or backwards to zoom in our out.

Another limitation of our system was that the keyboards were not always accessible when users wished to add a text label. In one case, users had switched positions and a user tried using the keyboard which was in front of him, which didn't work, since the keyboards could only send input to their "owner's" text flags. To address this, multiple keyboards, with the ability to distinguish input between users, could be placed around the display.

DOMAIN EXPERT INTERVIEWS

In addition to the observation sessions, we conducted interviews with experts in potential usage domains. The interviews were performed to obtain feedback on both our designed techniques but also on the display itself. Feedback from potential users on the technology is important for validating our belief that it can serve as a platform for collaborative usage. We met with three anatomy professors and one professional landscape architect. The interviews lasted about 60-90 minutes. The entire system was demonstrated and feedback on our techniques and the display was obtained throughout the interview.

The reactions in each of our interviews were quite positive. The overall response from all of the interviews was that the system had "tremendous potential". The anatomy experts said the system would be great for education, diagnosis, and in particular, surgical planning, due to the number of elements involved with complex 3D relationships. The architect said the system had a "huge range of prospects", with "almost no end of encouraging future applications", and that it would be great to incorporate in the design process for understanding relationships.

Another high-level response was the appreciation for a truly 3D virtual platform which could replace or at least complement current physical processes. One anatomy professor discussed how "anatomy museums", which display physical specimens inside enclosures for educational purposes, are burdened by security and storage costs, leaking fluids, plus legislative issues, since they involve biological specimens. A volumetric display would not suffer from these drawbacks, while maintaining the affordances of an enclosed physical 3D specimen. The architect commented that scaled-down physical models provide a necessary 3D viewing modality, but often prohibit creative design, because any mistake is costly, with respect to time and the cost of materials. If the model was instead displayed virtually inside a volumetric display, then the designer could worry less about making specific mistakes, and concentrate more on the creative design, while still obtaining the desired 3D viewing mode. We also obtained comments about specific features:

- Anatomy experts liked the rotation, as it replicated the physical Lazy Susan's used in anatomy museums.
- The architect liked the ability to physically walk around the display, and said it would allow designers to use their "innate biological resources" to understand the spatial relationships of a model. This comment is validation for our *location-sensitive interaction* design goal.
- The architect liked the mark-up tools and said it would allow designers to "analyze and interrogate" a 3D model.
- An anatomy expert appreciated the highlighting tool, and mentioned that it is often difficult to point out a feature to a student when a specimen is in an enclosure.
- Both the architect and anatomy experts said the culling functionality of the highlighting tool would be great to reveal inner relationships during a virtual 3D dissection.
- One anatomy expert liked that text labels could be added, moved, and deleted, because it would allow the labeling of a model to change over time. In a physically encased model, the inner labels are static and cannot be changed.
- The architect liked the idea of supporting subtle manipulations, as it would allow a designer to edit a model during the demonstrations which are typically done with rigid physical 3D models.

Finally, a number of new features were suggested. An anatomy professor wanted to see support for remote collaboration, so a surgeon in a different country could help plan a procedure, or a remote group of students could attend a virtual dissection. The ability to record usage sessions for future playback was also suggested. The architect mentioned that it would be interesting if the system was much larger, providing a more immersive viewing angle.

In summary, the experts all felt the system had great potential, saw numerous potential applications, and were impressed by a number of the features which were implemented. The fact that so many possible usage scenarios were suggested informs us that we were successful in developing interaction techniques which could be applicable to various usage domains.

IMPLICATIONS, LIMITATIONS, AND CONCLUSIONS

Although our prototype was developed for two users, in general the techniques and application which we implemented could be used by an arbitrary number of users without modification. An exception is the scene splitting functionality, which would require new considerations. A user may want a split scene which is shared by some, but not all of the other users. The system could also support the splitting of a viewport which was already split but shared. One feature that would not be possible with more than two users is the privacy viewing mode for text flags, since a text flag could only be hidden from one user.

Our location-aware assumption requirement was more critical to our application. While the Vicon markers gave us extremely accurate locations for the users, most of the system functionality would still work fine if precise positions were unknown. Alternative non-intrusive technologies which would provide a lower grade estimate of the user's location could be implemented. For example, stereo computer vision, sonar, or a pressure activated floor mat could all provide approximate body locations and would be invisible to the user. In the absence of these technologies, the input device locations could be used to infer the user locations. The only feature of our system which would significantly degrade with such less accurate position information is the hidden surface removal.

In conclusion, we have explored the issues surrounding collaborative interaction with volumetric displays. We discussed and implemented a number of new interaction techniques, adhering to three design goals, and outlined the results of usage observations and expert interviews with our system. The encouraging observations made during the usage sessions, in combination with the positive feedback received during our expert interviews, indicate that our work can serve as a baseline for the future development of collaborative applications for volumetric displays.

REFERENCES

1. Accot, J. and Zhai, S. (1997). Beyond Fitts' Law: Models for trajectory-based HCI tasks. *ACM CHI*. p. 295-302.
2. Agrawala, M., Beers, A. C., McDowall, I., Frohlich, B., Bolas, M. and Hanrahan, P. (1997). The two-user Responsive Workbench: support for collaboration through individual views of a shared space. *ACM SIGGRAPH*. p. 327-332.
3. Balakrishnan, R., Fitzmaurice, G. and Kurtenbach, G. (2001). User interfaces for volumetric displays. *IEEE Computer*. 34(3): p. 37-45.
4. Bier, E. A. and Freeman, S. (1991). MMM: a user interface architecture for shared editors on a single screen. *ACM UIST*. p. 79-86.
5. Bowman, D. and Hodges, L. (1997). An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. *ACM I3D*. p. 35-38.
6. Bowman, D. A. and Wingrave, C. A. (2001). Design and Evaluation of Menu Systems for Immersive Virtual Environments. *Virtual Reality*. p. 149-156.
7. Favalora, G. (2005). Volumetric 3D Displays and Application Infrastructure. *IEEE Comp*. 38(8): p. 37-44.
8. Favalora, G., Napoli, J., Hall, D, Dorval, R., Giovinco, M., Richmond, M. and Chun, W.S. (2002). 100-million-voxel volumetric display. *SPIE* V*ol. 4712*. p. 300-312.
9. Greenberg, S. and Marwood, D. (1994). Real time groupware as a distributed system: concurrency control and its effect on the interface. *ACM CHI*. p. 207-217.
10. Grosjean, J., Burkhardt, J. M., Coquillart, S. and Richard, P. (2002). Evaluation of the Command and Control Cube. *IEEE ICMI*. p. 473-478.
11. Grossman, T. and Balakrishnan, R. (2006). The Design and Evaluation of Selection Techniques for 3D Volumetric Displays. *ACM UIST*. p. 3-12.
12. Grossman, T. and Balakrishnan, R. (2006). An evaluation of depth perception on volumetric displays. *AVI*. p. 193-200.
13. Grossman, T., Wigdor, D. and Balakrishnan, R. (2004). Multi finger gestural interaction with 3D volumetric displays. *ACM UIST*. p. 61-70.
14. Grossman, T., Wigdor, D. and Balakrishnan, R. (2007). Exploring and Reducing the Effects of Orientation on Text Readability in Volumetric Displays. *ACM CHI*. p. 483-492.
15. Mine, M. (1995). Virtual environment interaction techniques. *UNC Chapel Hill Department of Computer Science Technical Report TR95-020.*
16. Morris, M. R., Ryall, K., Shen, C., Forlines, C. and Vernier, F. (2004). Beyond "social protocols": multi-user coordination policies for co-located groupware. *ACM CSCW*. p. 262-265.
17. Pierce, J., Conway, M., Dantzich, M., and Robertson, G. (1999). Toolspaces and glances: storing, accessing, and retrieving objects in 3D desktop applications. *ACM CHI* p. 163-168.
18. Poupyrev, I., Billinghurst, M., Weghorst, S. and Ichikawa, T. (1996). The go-go interaction technique: non-linear mapping for direct manipulation in VR. *ACM UIST*. p. 79-80.
19. Rosen, P., Pizlo, Z., Hoffmann, C. and Popescu, V. S. (2004). Perception of 3D spatial relations for 3D displays. *Stereoscopic Displays XI*. p. 9-16.
20. Ryall, K., Forlines, C., Shen, C. and Morris, M. R. (2004). Exploring the effects of group size and table size on interactions with tabletop shared-display groupware. *ACM CSCW*. p. 284-293.
21. Shen, C., Vernier, F., Forlines, C. and Ringel, M. (2004). DiamondSpin: An extensible toolkit for around the table interaction. *ACM CHI*. p. 167-174.
22. Stewart, J., Bederson, B. and Druin, A. (1999). Single display groupware: A model for co-present collaboration. *ACM CHI*. p. 286-293.
23. Tsang, M., Fitzmaurice, G. W., Kurtenbach, G., Khan, A. and Buxton, B. (2002). Boom chameleon: simultaneous capture of 3D viewpoint, voice and gesture annotations on a spatially-aware display. *ACM UIST*. p. 111-120.
24. Viega, J., Conway, M. J., Williams, G. and Pausch, R. (1996). 3D magic lenses. *ACM UIST*. p. 51-58.
25. Wigdor, D. and Balakrishnan, R. (2005). Empirical Investigation into the Effect of Orientation on Text Readability in Tabletop Displays. *ECSCW*. p. 205-224.
26. Wigdor, D., Shen, C., Forlines, C. and Balakrishnan, R. (2006). Effects of display position and control space orientation on user preference and performance. *ACM CHI*. p. 309-318.
27. Wu, M. and Balakrishnan, R. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. *ACM UIST*. p. 193-202.
28. Zanella, A. and Greenberg, S. (2001). Reducing interference in single-display groupware through transparency. *ECSCW*. p. 16-20.