# The Virtual Annotation System

Reid Harmon,[a] Walter Patterson,[b] William Ribarsky,[a] and Jay Bolter,[a,c]
Georgia Institute of Technology

## Abstract

*Annotation is a key operation for developing understanding of complex data or extended spaces. We have developed a flexible set of annotation tools that can be placed in a variety of applications. These tools offer a full set of capabilities for inserting, iconizing, playing back, and organizing annotations in a virtual space. They also have an intuitive and easy-to-use interface for employing these capabilities while immersed in the virtual environment. We illustrate the annotation system with two diverse examples: a general data visualization/ analysis application and an architectural walkthrough.*

## Overview

Scientists, engineers, planners, designers – anybody involved in a complex or extended project – feels the need to make notes as part of the process of organizing his/her thoughts, grasping the structure of the project, and ultimately gaining understanding. These workers often associate their notes with artifacts of the project; they may write on plots, images, text, diagrams, spreadsheets, and so on. These jottings are important for developing understanding. In fact, recent studies of scientists confirm that note-taking that preserves context and is descriptive is a key part of the analysis process.[1]

If one were building an information visualization/analysis system that purported to provide complete support for the investigative or design process, one would undoubtedly include support for note-taking or annotation.

Yet most data visualization systems provide little support for annotation, permitting little more than the attachment of labels to 2-D static images. In particular, the systems do not provide support for organizing annotations for review or for sharing with other project workers.

When one deals with 3-D data, one needs the capability to associate notes with objects or happenings in the 3-D space. This is part of the important process of preserving contextual information during analysis.[1] This need is heightened if the analyst must navigate through an extended 3-D space or must follow the extended dynamics of a system (as from a simulation). If handled correctly, such notations can provide signposts that enable users to recall detailed steps in even quite complicated investigations. This process of locating semantic objects in visualizations has been a long tradition indeed. Orators in ancient Greece and Rome used spatialization to memorize speeches as imaginary walks through 3 dimensional structures.[2]

The need to make notes while immersed in a 3-D space extends well beyond data visualization. Architectural walkthroughs or extended design spaces are enriched by the ability to post notes on walls, furniture, and other objects. Users of immersive training environments that may involve many rooms (such as shipboard emergency response trainers for the Navy) or even extend over several miles in virtual space (such as command-level tactical trainers for the Army) could benefit greatly from directions, instructions, or tactical information placed at key points and times in the virtual environment (VE). Complex 3-D interfaces with many tools and modes of interactions could themselves be annotated to speed the user's learning of the interface. These annotations might not only be text but could also be verbal or images (e.g., video) showing the tools in action. Indeed explorers of any 3-D environment that is extended, complex in space or time, involving several variables, or populated with objects and behaviors that may change depending on location or time would significantly benefit from the ability to make annotations. In addition those who employ collaborative or distributed environments would find effective uses for 3-D notes that could be shared or left to be retrieved by others.

In this paper we describe a simple but workable system for making and handling annotations while immersed in a 3-D space. The system addresses the needs of both information visualization/analysis systems and exploratory walk- or fly-throughs. Since we see the need for note-taking arising in a multitude of areas, we have tried to design our system so that it is generally applicable. To show this generality we present here two widely different examples of the system's use.

## Previous work

There has been surprisingly little work in the

[a] Graphics, Visualization, and Usability Center
[b] School of Civil Engineering
[c] School of Literature, Culture, and Communication

development of tools for annotation in 3-D spaces. The direct forbear of this project is work by Verlinden, Bolter, and van der Mast [3] on annotation and verbal communication in virtual reality (VR). This work uses a conventional mouse, with 6-D trackers attached, to position and orient a 3-D cursor in the VE. Upon selection of an object, the user can make a voice annotation simply by pressing a mouse button; an annotation marker then appears that can be subsequently selected for playback. The Virtual Annotation (VAnno) system uses a similar mechanism for storing voice annotations but allows longer recordings, appending, and discarding. It also goes beyond the work of Verlinden *et al.* in providing a complete interface including a 3-D control panel from which one controls these features by direct manipulation. Among other new features, the VAnno system also allows the organization of annotations for review, sharing, or guided tours of the annotated data.

To our knowledge there are no other annotation systems for VEs. However, there has been work done in windowed environments. Recently, Loughlin and Hughes (LH) [4] have developed an excellent system for annotating fluid flow visualizations. They use a "post-it" metaphor for embedding annotations into the 3-D data space. An important feature of the LH system is to use a Magic Lens filter [5] to create specialized views of the data, and a unique capability is to allow users to sketch and then annotate 3-D volumes. The main differences between this system and our annotation system are, of course, that ours is an immersive rather than a windowed environment and that LH uses written annotations (mostly text) whereas we use voice annotations. (Our annotation structure could easily be generalized to include written and image annotations.) The immersive characteristics permit us to emphasize direct manipulation controls and tools and high interactivity. Our system has certain new features such as extended modes for annotation organization and guided tours, whereas the LH system has extensions such as the Magic Filter and annotated volume. In addition, our system is general with a range of applications.

Outside of the systems mentioned above, there is little support for annotations in scientific visualizations. In dataflow and point-and-click environments, annotations consist mostly of attaching labels, markers, etc. to plots or images. There is no capability for full embedding of these annotations in the 3-D space.

## Use of sound

The usual input devices, such as the mouse and keyboard, are not easily available to the user while immersed in a VE with head mounted display. The resolution of the most widely used head-mounted displays is limited to a picture quality no better than a standard NTSC (TV) signal. Although HMDs with significantly higher resolution are available on the market, these cost several times as much as the lower resolution products and remain out of the price range of most VE developers. This cost disparity may continue for the foreseeable future. The lower resolution displays are unsuited for the presentation of text other than short strings (such as control words or menu text).

For these reasons the use of audio or voice annotations within a virtual environment presents a desirable solution. Users encumbered by physical restrictions and limited by their inability to employ traditional input or output devices while immersed in a VE can take advantage of speech as input and can trigger the playback of prerecorded annotations or other messages as output.

Much of the current research on the use of sounds in a VE focuses on 3-D sounds that appear to the user to have a definite source, direction, or location. Brewster *et al.* [6] have experimented with the use of "earcons" to impart complex messages. Earcons are abstract synthetic tones that can be used alone or in structured combinations to create sound messages. The authors concluded that sound can be used to increase the total amount of information received or reduce the amount that has to be received solely through the visual channel. Thus, greater quantities of more complex information can be transferred and understood with less strain on the subject. In fact, the authors point out that psychological evidence suggests that using different sensory modalities to convey information can actually improve task performance [6].

Although the research performed concerning earcons focused primarily on nonverbal sound, several of the resulting guidelines for using earcons could prove useful in constructing an annotation system. For example, researchers recommend that the volume intensity be kept within a close range so that if the user changes the volume no sound will be lost and that individual sounds be kept at more or less the same relative intensity. They also recommend that a delay of 0.1 seconds be used between playing different earcons so the user can tell when one sound ends and another begins. This is also appropriate for a gap between voice annotations if a user activates several in rapid succession.

Also pertinent to this work is the VoiceNotes system [7], which employs a hand-held microcassette recorder with a voice interface designed to make recording and playback of annotated linear voice notes easy and productive. One of the problems with this non-immersive system was that voice annotations are quick for the creators, allowing them to input a large amount of information in a short period of time. For the listener, on the other hand, voice notation can be more cumbersome than written text, which can be quickly scanned for meaning. By creating a system that is intuitive for the creator of the notes and allows the user to search for key words and topics within the recorded

notes, the developers were able to make the best use of a voice system for recording lists, thoughts, and other material. [7] They conclude that using multiple input and output modalities (e.g., VoiceNotes, voice and button input, speech and non-speech output) combines the capabilities of each, while minimizing limitations of a particular modality. We face the same obstacle with cumbersome output in our Virtual Annotation system and have formulated a plan to implement similar means and multiple modalities to overcome it. However, the VAnno system also has the advantage that the user can introduce some structure to the recorded notes by using specific icons for particular types of displayed data or objects. The system itself also can organize the notes according to the author, date, time, type of object annotated (e.g., data or control widget), and so on.

## Elements of a useful virtual annotation system

To provide maximum benefit, an annotation system must be a fully integrated component of a virtual environment. This complete integration allows users to freely use the annotation system when needed, without the need for special modes of operation. A voice annotation system must primarily provide a quick means for recording and playing back speech. Also needed is the capability to easily edit, remove, or save annotations. Finally, the annotations must be stored or indexed in a fashion that allows users to retrieve specific annotations or classes of annotations grouped together in a way meaningful to the user.

Recording of annotations can be accomplished by providing the user of the VE with a microphone connected to an audio input port on the computer. Controls are needed to activate the recording process, and to signal when the end of a recording has been reached. These controls could be virtual controls within the VE, such as iconic tape-player shuttle controls. Alternately, these controls could be hand-held devices physically manipulated by the user. Examples of possible physical devices include a mouse, joystick, keyboard, foot pedals, or a button box. However, the desire for immersion in the virtual environment precludes the use of devices that cannot readily be manipulated without the use of the eyes. The device must be simple or familiar enough so that the user can literally operate it with his eyes closed, such as with the brake, accelerator, and turn signals in a car.

For the recording process to be complete, the user will often need to specify what aspect of the environment the annotation applies to. For example, the user may wish to annotate a specific place in the environment, a specific object, or a specific time or range of times within a simulation or animation. Some means of providing this association with the recorded speech must be included. Additionally, the user may wish to assign properties such as color, shape, or size to any geometric representation of the recorded annotation within the virtual environment.

Playback of annotations in the VE must be accomplished through similar means. The user needs some control to choose which annotation to hear. More sophisticated controls might allow the user to pause, rewind, and fast-forward the annotation, and to display relevant information such as which user originally made the annotation, when the it was recorded, and which objects, locations, or time steps within the virtual environment are associated with the annotation.

If a 3-D graphical icon is used to represent the annotation in the virtual environment, then the user can select the icon for playback of the recorded annotation in the same way that other objects are selected or manipulated for other purposes in the environment. For example, a pointer could be used in conjunction with a hand-tracking device. Alternate methods could use other types of gesturing or picking devices, or simple proximity algorithms.

The voice annotations recorded in the VE could be linked to more traditional textual annotations made outside the VE. If the voice annotation could be associated with a specific text file, then users within the virtual environment could view this text while remaining immersed. Similarly, users browsing the text from a more traditional command-line approach could automatically choose to hear the voice annotation made by a user in the virtual environment. The annotation icons could also be linked, in a straightforward way, to images or even to animation sequences. These could be special pre-recorded notes; when the user chose them she would see the image or animation and then be returned to the virtual scene.

## Our annotation system

VAnno is centered around the recording and playback of voice annotations for two reasons. First, sound, even through standard *SGI Indigo* audio equipment, provides a representation with more information content than text in current HMDs. Secondly, a user can make voice recordings while wearing a HMD, without the need for the keyboard. The primary drawback to the use of voice annotations is that editing is not as conceptually or technically simple as text editing, and we have not directed our efforts towards developing VE audio editing capabilities. The user can append a note to an existing annotation, re-record it, or discard it altogether, but is otherwise restricted in audio editing ability.

The system supports two types of annotation: *object annotation* and *view annotation*. Object annotation refers to notes on specific objects in the VE. When an annotation is made for an object, an *annotation icon* is created and placed in the VE, and a line connects the icon to the object. (Figure 1 shows a sample icon that

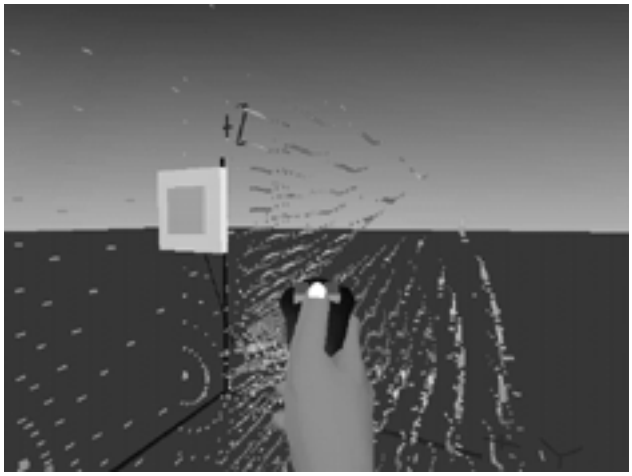

Figure 1: Object annotation icon marking annotated data from a molecular dynamics simulation. VAnno has been inserted into the Virtual Data Visualizer application.

marks an annotation in a molecular dynamics system in the VDV.) The line allows the icon to be placed away from data objects, which is important since the user will not want to obscure the data. If an icon of this type is moved, the system saves its new position and adjusts the line to the annotated object accordingly. We decided to use icons to illustrate an annotation rather than altering the appearance of the annotated object since, in data visualization applications, many or all of the object's appearance attributes carry semantic information. (For example, color or shape may be controlled by a data variable.) The VAnno system supports object annotation both with and without icons. In the latter case, the annotated object itself will have a special color or other attribute. This mode is useful in the architectural walkthrough example described below.

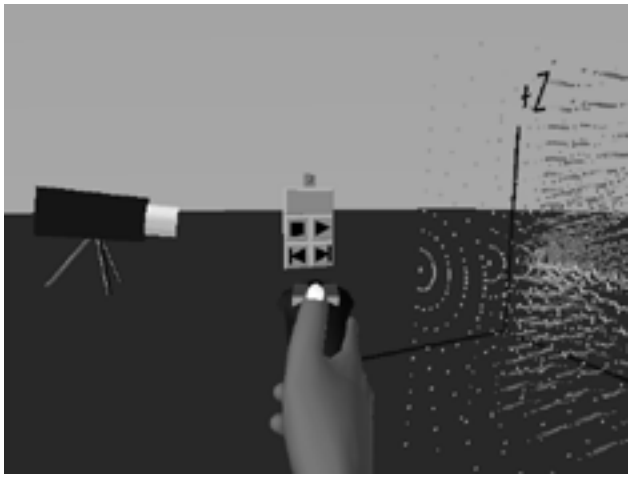

Figure 2a: The camera icon marking an annotated view of the molecular dynamics simulation.

Each user of the system is registered when he/she runs the application for the first time, and a default icon for that user is chosen from a static grab-bag. The user's default icon appears when an object annotation is made. When a view annotation is made, an annotation icon is created and placed at the 6-D coordinates of the user. This icon is not the user's default icon, but a camera on a tripod, pointing in the user's view direction. Note, however, that the owner information for this icon is retained in the annotation file for later search and retrieval. (Figure 2a shows the camera icon in the VDV. Figure 2b shows the view that the user chose and that she sees when the icon is selected.) The camera icon may also be moved, but the system retains its original position in the annotation file.

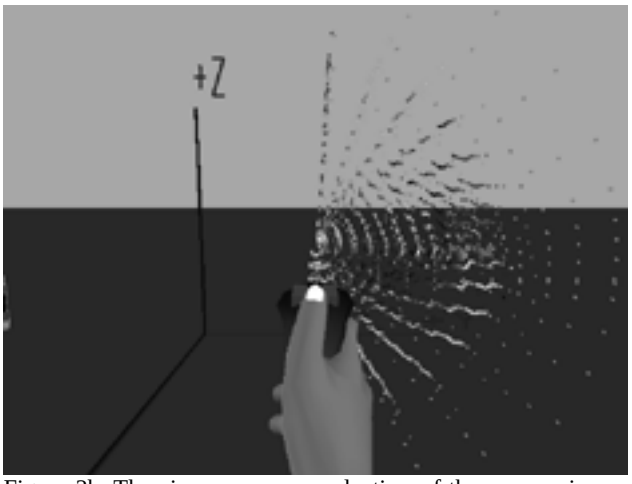

Figure 2b: The view seen upon selection of the camera icon.

In addition to the annotation icons, the VAnno system adds many objects to its host application. These are the *annotation mode switch* (depicted in Figure 3 as a blue letter A against a gray square), which toggles the annotation mode on and off, the *annotation menu* (covering the right half of Figure 3), which lists all annotations in the system (and each menu item is its own object), and 4 *annotation control* objects (set under the annotation mode switch in Figure 3). Internally, the annotations are stored in a linked list of annotation specifications, of type *AnnotSpec*. Each *AnnotSpec* stores a pointer to its icon object, its menu item object, and, optionally, the application object, as well as the time and date of creation, the user who created the annotation, the user's 6-D position at the time of creation (stored as a 4x4 transformation matrix), the name of the AIFF file containing the audio recording, and the annotation's *frame number*. The frame number refers to the discrete time in a simulation for which the annotation is active. This may be quite important in data visualization applications that may show a large number of time frames from a simulation. Both it and the user's 6-D position are optional values. If the annotation does not have a frame number, then it spans all simulation time. An icon in the system is only visible if both the annotation mode is on and the

annotation is active for the current application simulation time. Thus the appearance/disappearance of annotation icons indicates the time steps at which they were made, and they can be turned off entirely to remove clutter from a scene.

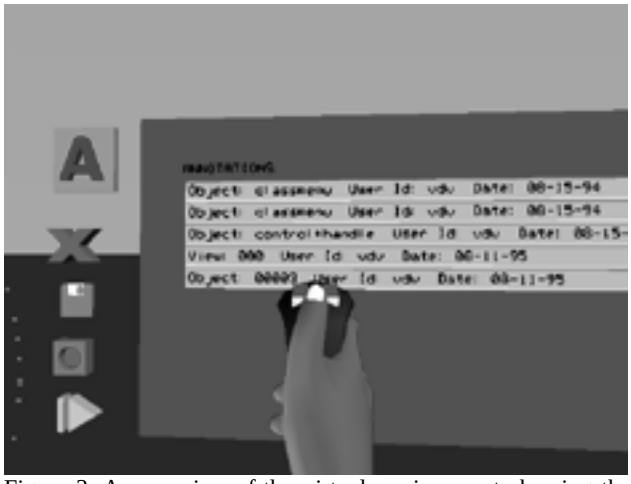

Figure 3: An overview of the virtual environment showing the annotation controls and menu.

Externally, annotations are stored in an *annotation* file. The format of the annotation file is zero or more *user entries* followed by zero or more *annotation entries*. A user entry has the userid plus the name of a default icon file. The annotation entry has several fields specifying the object annotated, time, date, etc. At initialization, the annotation system reads the annotation file and stores the information in an internal form. This internal form is modified while the application program runs, and then the annotation file is rewritten when the application program exits. The simple, modular control and file structure of VAnno make it quite flexible and easy to attach to application programs.

When the annotation mode switch is off, all annotation objects except for the switch itself are invisible, and when the switch is on, all annotation objects are visible except for icons whose annotations are outside the current simulation time. When an annotation menu item (or a camera icon) is selected, the relevant annotation specification is found in the list, and then the user is moved to the 6-D position of creation (if the annotation has one), the simulation time is set to the frame number (if the annotation has one), and the AIFF recording is played. This is to put the user in the same frame of mind as the creator of the annotation. (The transitions to the new 6-D position and simulation time are at present instantaneous, but we are investigating using a smooth transition instead.) Additionally, by selecting annotations from the menu in sequence the user can take a "tour" of the application and its annotations.

The four annotation controls are *record*, *play*, *save*, and *cancel*, represented respectively by a gray square inset with a red circle, a green arrow, a pink floppy disk, and a red X. (See Figure 3.) An annotation is made by first selecting an object, or by selecting no object (for view annotations). The user's 6-D position is marked at this time. Then *record* is selected and held, and the user speaks into a microphone. When the user releases the record control, the audio recording stops. The user may select *play* to replay the recorded audio or re-record indefinitely. If *cancel* is selected, the system removes the recorded audio file. If *save* is selected, however, the annotation specification is created and added to the internal list, and the annotation icon and annotation menu item appear.

## Annotations in the Virtual Data Visualizer

The Virtual Data Visualizer [8] is a highly interactive, immersive environment for visualizing and analyzing data. VDV is a set of tools for exploratory data visualization that does not focus on just one type of application. It employs a data model with data arranged hierarchically in classes that can be modified by the user within the virtual environment. The class structure is the basis for bindings or mappings between data variables and glyph elements, which the user can make, change, or remove. The binding operation also has a set of default operations so that the user can quickly display the data. We have installed VAnno within VDV; Figures. 1-3 show this environment. We have used this environment for a variety of data including the molecular dynamics simulations, as shown, and also simulations of unsteady fluid flow. VAnno fits nicely with the VDV controls; the user can run through a simulation and annotate views or particular data elements. This provides a unique capability to interpret the data, even down to the level of individual data elements, in space and time. As an extension to VAnno, the bindings of data variables to glyphs--3-D graphical objects whose elements (e.g., position, size, shape, color, orientation, etc.) are bound to data--are saved along with the annotation.

## Annotations in the architectural walkthrough

We can easily insert VAnno into an entirely different application. We have set up walkthroughs or flythroughs for buildings and shipboard spaces. These include some extended spaces such as a shipboard environment with several levels, the Olympic stadium in Atlanta, and a representation of the Georgia Tech campus including accurate topography and buildings. We have implemented an annotated flythrough of the latter using VAnno tools. As an example of a specific implementation, Figures. 4 and 5 show scenes from an annotated walkthrough of the GVU Center. Fig. 4 shows the GVU conference room ; the table is being selected to playback a note describing the function of

the space. Fig. 5 shows a workstation in one of the GVU research areas whose note describes its capabilities and functions. Objects throughout the GVU space are annotated, allowing the user to browse through the space and discover the uses, contents, and capabilities of its parts. In this case VAnno is used without icons; the objects themselves are selected. As an option the objects can have distinctive colors or flash to indicate that they hold annotations.

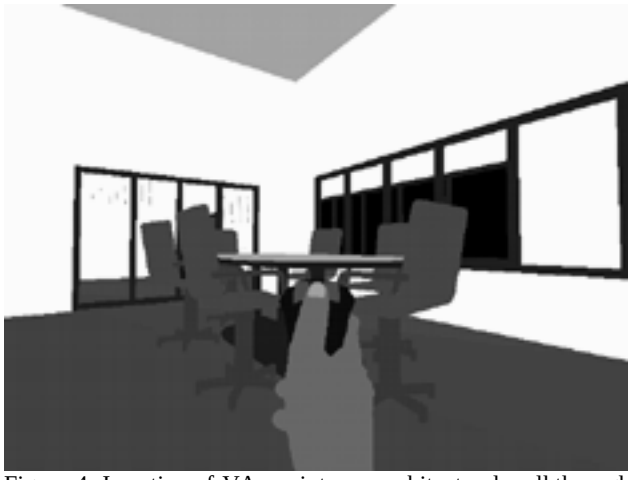

Figure 4: Insertion of VAnno into an architectural walkthrough of the GVU Center. An annotation attached to the table and describing the GVU conference room has been selected. In this case the objects themselves are annotation icons.

Figure 5: Selection of an annotated workstation in the GVU Center. The annotation describes the configuration and uses of the workstation.

Most of the objects and the modes of interaction in both the architectural walkthrough and VDV are built using the Simple Virtual Environment (SVE). [9] The SVE is a toolkit developed at Georgia Tech that allows one to set up a virtual environment simply and quickly. It provides a stable and ever-expanding environment for building applications since new features from applications are often incorporated into SVE if they are of general use. Building on SVE insures that object identifications and modes of navigation and interaction assumed in VAnno are available in the environment.

## Usability

We did some preliminary usability tests involving several users in environments with and without annotations. In a flyover of the Georgia Tech campus, users had difficulty locating where they were and identifying campus buildings. The annotations improved both the location and identification processes. The same was true in the VDV environment where users found annotations identifying the various tools helpful, too. The initial interface was judged too difficult to use, so the present one with clear iconic buttons replaced it. We plan more formal and extensive usability tests for the VDV tools.

## Future work

We will want to look closely at VE interface issues. The annotation system supports more capability than what is yet possible within the VE itself. For example, a user's default icon may be changed by directly editing the annotation file, but there is not yet a means in the VE to do this. A possible approach would be to present a board on which all supported icons are visible, and the selection of one of these icons would change the current user's default. But should this board always be visible when the annotation mode is on (like the annotation menu), or should the user pop it up? Besides the AIFF file, links can also be made to other files that may contain images, animations, or even extended text. Should these be presented in some window within the environment or perhaps on a virtual screen? We will explore the content and use of these files as they can be of great help in training environments.

At the moment the annotation menu is organized by creation time. We shall extend selection capability so that the user can organize the menu by userid, annotation location, data or object type, annotation content, or other parameters. This will be a very useful capability if the environment is complex, there are several users making notes, or the list of annotations is long. Although controls or buttons for doing this can readily be implemented within SVE, we wish to perform usability tests to optimize this design and, indeed, the whole VAnno interface. These tests will also give us an idea of the effectiveness of the Annotation system and its range of utility for different applications.

## Conclusions

We have presented VAnno, which is a complete set of tools for making, finding, and reviewing annotations while immersed in the VE. We have shown

how a simple collection of controls, icons, and menus can make an intuitive, easy-to-use interface for these annotations. VAnno is also flexible and can be inserted into a range of applications, which we have shown with two quite different examples. VAnno makes it easy to relate a comment to a data feature, a time step, or a viewpoint. Since any object in the scene can be annotated (including controls and menus), the system supports the development of training environments. It also can easily be extended to displaying annotations containing text or images. By selecting annotations organized in the annotation menu, the user can review important points or take tours of the data and the virtual space. Users can also review notes left by collaborators and add to them.

VAnno offers a flexible system for analysis and explanation that grows in power as the data becomes more complex in space and time and as it is analyzed by multiple collaborators. Indeed we are now inserting annotation capability into an immersive, 3-D geographical information system. In this system, since we have implemented novel methods for managing large stores of topographic data and related information, users can fly for hundreds of miles over high resolution terrain. The use of annotations will be critically important for this case.

# References


1. R. Springmeyer, M. Blattner, and N. Max. A Characterization of the Scientific Data Analysis Process. *Proc. Second IEEE Conference on Visualization*, pp. 351-352, 1992.
2. Frances Yates. The Art of Memory. University of Chicago Press, Chicago, 1966.
3. J. Verlinden, J. Bolter, and C. van der Mast. Voice Annotation: Adding Verbal Information to Virtual Environments. *Proc. European Simulation Symp.*, pp. 60-69, 1993.
4. Maria Loughlin and John Hughes. An Annotation System for 3D Fluid Flow Visualization. *Proc. IEEE Visualization '94*, pp. 273-279, 1994.
5. E. Bier, M. Stone, K. Pier, W. Buxton, and T. DeRose. Toolglass and Magic Lenses: The See-through Interface. *Proc. SIGGRAPH '93*, pp. 73-80, 1993.
6. Stephen A. Brewster, Peter C. Wright, and Alistair Edwards. An Evaluation of Earcons for Use in Auditory Human-Computer Interfaces. *InterCHI '93*, pp. 222-227, 1993.
7. Lisa J. Stifelman, Barry Arons, Chris Schmandt, and Eric Hulteen. VoiceNotes: A Speech Interface for a Hand-Held Voice Notetaker. *InterCHI '93*, pp. 179-186, 1993.
8. R. van Teylingen, W. Ribarsky, and C. van der Mast. The Virtual Data Visualizer. GVU Center Technical Report GIT-GVU-95-16, 1995.
9. Jouke C. Verlinden, Drew Kessler, Larry F. Hodges. The Simple Virtual Environment (SVE) Library: Users Guide. GVU Center Technical Report GIT-GVU-93-24, 1993.