

Enabling Scientific Workflows in Virtual Reality

Oliver Kreylos* Gerald Bawden† Tony Bernardin* Magali I. Billen‡ Eric S. Cowgill‡ Ryan D. Gold‡
Bernd Hamann* Margarete Jadamec‡ Louise H. Kellogg‡ Oliver G. Staadt* Dawn Y. Sumner‡

Abstract

To advance research and improve the scientific return on data collection and interpretation efforts in the geosciences, we have developed methods of interactive visualization, with a special focus on immersive virtual reality (VR) environments. Earth sciences employ a strongly visual approach to the measurement and analysis of geologic data due to the spatial and temporal scales over which such data ranges. As observations and simulations increase in size and complexity, the Earth sciences are challenged to manage and interpret increasing amounts of data. Reaping the full intellectual benefits of immersive VR requires us to tailor exploratory approaches to scientific problems. These applications build on the visualization method's strengths, using both 3D perception and interaction with data and models, to take advantage of the skills and training of the geological scientists exploring their data in the VR environment. This interactive approach has enabled us to develop a suite of tools that are adaptable to a range of problems in the geosciences and beyond.

CR Categories: 1 [Fundamentals in Virtual-Reality Continuum]: Scientific visualization in virtual-reality continuum; 4 [Applications]: Virtual-Reality Continuum in Geology, Geography and GIS

Keywords: Virtual reality, scientific visualization, workflow, geosciences

1 Introduction

The typical workflow of a geoscientist generates insight through observation and measurement. To use an example from the work of one of our group [Gold et al. 2006]: a geologist researching the behavior of an active fault observes terrain in the field, measures the position, orientation, offset, and other properties of fault lines and records them in a map, and then generates a model of the local tectonic structure based on those measurements (see Figure 1). More generally, a geoscientist first detects features, then measures them, and finally generates a model explaining the features. This visual approach to science is powerful, as the human brain excels at visually identifying patterns. As Edward Tufte wrote two decades ago: “At their best, graphics are instruments for reasoning about

quantitative information. Often the most effective way to describe, explore, and summarize a set of numbers – even a very large set – is to look at pictures of those numbers” [Tufte 1983]. Unlike many engineering applications in which the exact shape of a structure is known, geologic structures often have complex three-dimensional shapes, that additionally may only be sampled at irregularly distributed locations. It is therefore desirable to apply the feature detection skills of a trained geoscientist to problems that are not accessible for “in situ” inspection, such as remote locations, the ocean floor, the interior of the Earth, or the surface of a different planet. Moreover, many geologic features exhibit scale invariance [Turcotte 1997]; i.e., phenomena are mathematically similar at vastly different scales. Although it is not possible to directly observe the topography of a crystal surface at the molecular scale, nor the structure of tectonic boundaries at the global scale, the same geoscience observation and analysis skills apply to datasets collected at these widely different scales.

To improve the return of scientific insight from geoscience data, we have created (immersive) visualization software and measurement/analysis tools that allow scientists to use real-world skills and methods inside virtual environments. At the core of this approach are techniques to display geoscience data at high detail and the frame rates required for immersive visualization. Our experience indicates, however, that virtual reality (VR) visualization alone is not sufficient to enable the processes scientists employ in the real world. Interaction with, measurements of, and manipulation of the displayed data are important for feature detection, and they are essential for quantitative analysis. In our example, a scientist must be able to measure and record the orientation of surfaces (strike and dip angles) from 3D topography models. For many applications, interaction, such as interactively fitting model parameters to observations, is also desirable in the model generation stage. Even if an automatic fitting method is available, observing the response of a model to parameter changes can lead to new insight into the model's governing equations.

To summarize: our approach is to create immersive visualizations of geoscience data, whether the scale of the natural system is millimeters or hundreds of kilometers, in a user's near field, i.e., within arm's reach, and to provide users with the tools needed to touch, manipulate, and measure their data. Our software is aimed at stereoscopic head-tracked visualization environments with six-degree-of-freedom (6-DOF) input devices, and it operates effectively in a wide variety of environment types from CAVEs, to display walls and workbenches, to standard desktop PCs (of course with reduced effectiveness towards the lower end). We discuss the visualization and interaction challenges posed by contemporary geoscience applications, our approach to create truly portable VR software, and the perceived impact of our software on geoscience research.

1.1 Case Studies

To illustrate our approach with concrete examples, we have selected three driving geoscience problems.

*Institute for Data Analysis and Visualization (IDAV), Department of Computer Science, University of California, Davis

†US Geological Survey, Water Resources Division

‡Department of Geology, University of California, Davis

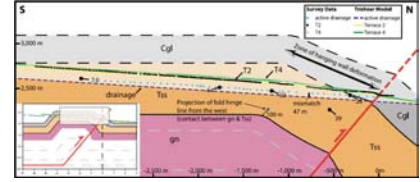
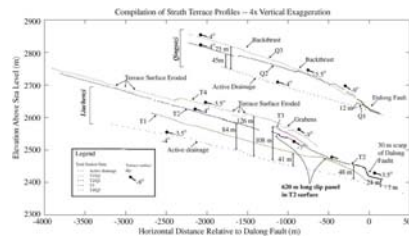


Figure 1: Scientific workflow of field-based geologic mapping. From left to right: observations made in the field; measurements, recording elevations across a fault; and insight, in the form of the inferred cross-section above and below current ground elevation at a mapped location.

Geological Mapping Geological mapping is the identification of structures such as faults, folded layers of rock, and geomorphic features from data collected in the field or from digital elevation data and multi-spectral satellite or photographic imagery [Gold et al. 2006]. Reconstructions of geologic features are then used to interpret present and past deformation of the Earth's crust as it responds to the forces of plate tectonics and is modified by processes of erosion and deposition. A geologist has to make detailed observations over large areas by viewing the region of interest from many perspectives and at different scales, by detailed analysis of focus regions, and by direct measurement of the location and orientations of often complex planar and undulatory 3D structures, defined solely by their intersection with the 3D surface topography.

Displacement Analysis High-resolution laser scanners offer a new method to accurately measure the deformation of the Earth's surface and of natural or man-made structures due to geological events such as landslides, floods, or earthquakes [Bawden et al. 2005; Kreylos et al. 2005]. A tripod-based terrestrial LiDAR (light detection and ranging) scanner can measure the 3D position of points on surfaces at accuracies in the millimeter range, and can gather surface data sets containing several million points in only a few hours. Geologists can measure deformations by comparing two LiDAR scans taken at different times, e.g., before and after an earthquake. Due to the randomness of sampling and inherent noise, these comparisons cannot be performed point-by-point. Instead, they require the identification of features common to both scans, and the measurement of derived properties of these features, such as plane or cylinder equations or intersection lines or points.

Plate Subduction Simulation Geoscientists employ finite-element-method (FEM) fluid dynamics simulations to investigate the fate of tectonic plates entering the Earth's mantle in the vicinity of subduction zones such as the Aleutian chain [Billen and Gurnis 2003]. The observed data used to generate initial configurations for the simulation are the shape of a subduction zone observed via ocean floor topography, and the location of the subducting slab in the Earth's mantle reconstructed from the locations of deep earthquakes. To successfully run a simulation, researchers have to convert these line and scattered point data into smooth 3D temperature and viscosity fields. This setup process involves numerous conversion and filtering steps, and unsatisfactory initial models will lead to convergence failure during simulation, thus wasting tremendous amounts of researcher's time and CPU cycles.

2 Related Work

As geoscientists often need to analyze complex 3D data for their research, the 3D perception offered by immersive visualization could

be highly beneficial for their workflows. In fact, there are many previous approaches, with a focus on oil and gas exploration [Evans et al. 2002].

Lin and Loftin [Lin and Loftin 1998; Lin et al. 1998] presented visualization techniques for 3D seismic data in CAVE [Cruz-Neira et al. 1993] environments. They focused in particular on interaction techniques such as selection, manipulation, range selection, and parameter input. Results of a user evaluation of these techniques were presented in [Lin et al. 2000]. They concluded that VR provides a better 3D visualization environment, but verification and interpretation of 3D seismic data for well planning was less effective than in desktop environments. This can be attributed possibly to a lack of design and validation methodology for interactive applications in immersive VR environments as pointed out by van Dam et al. [van Dam et al. 2000] and Johnson [Johnson 2004]. Gruchalla [Gruchalla 2004] investigated the benefits of immersive VR for well-path editing. He reported speed and accuracy improvements of immersive systems over desktop system, based on a study with 16 participants. Simon [Simon 2005] presented the VRGEO Demonstrator project for co-located interactive analysis of complex geoscience surfaces and volumes in immersive VR systems.

VR technology has also been used for remote sensing data exploration. Di Carlo [Di Carlo 2000] developed a VR toolkit that allows users to interact with different 3D representations of the same data simultaneously in an immersive environment. Chen [Chen 2004] presented a non-immersive application that allows multiple users to explore and interact with 3D geographical data of the Tibet Plateau across the internet. Gardner et al. [Gardner et al. 2003] developed another non-immersive system for analysis of LiDAR data. They noted that being able to visualize LiDAR data directly in 3D greatly helped the analysis of complex structures, but that better analytical tools for creating, editing, and attributing features need to be developed.

Harding et al. [Harding et al. 2000] developed a system for geoscientific data exploration. They integrated interactive 3D graphics, haptics, and spatial sound into a multimodal user interface. Recently, Head et al. [Head, III et al. 2005] presented ADVISER, an immersive visualization system for planetary geoscience applications. This system combines cartographic data and interactive terrain visualization with *virtual field tools* to, for example, analyze the north polar-layered terrain on Mars.

In addition to immersive projection environments such as the CAVE [Cruz-Neira et al. 1993] or the blue-c [Gross et al. 2003], the GeoWall [Steinwand et al. 2002], with its greatly reduced cost at the price of reduced immersion, has gained increasing popularity within the geoscience community.



Figure 2: Observation in VR environment. Left: Observation of the topographic expression of valleys draining into Lake Tahoe. Center: Examination of the UC Davis water tower and Mondavi Center for the Performing Arts, measured by several LiDAR scans. Right: Comparison of surface features to the distribution of subsurface earthquakes in the Earth’s crust and mantle.

3 Virtual Reality Workflows

The major goal of our work is to provide researchers with the tools they need to apply the same processes they use in the real world to new problems in VR environments. In order to reach this goal, we need to create integrated software to implement three main components:

Real-time Visualization The starting point of our workflows is observation. To enable observation, we need to provide both real-time visualizations of large and highly detailed data sets, and intuitive navigation methods. This combination allows researchers to inspect their data in the same way they make observations in the field. For example, to detect faults in a terrain model, a user might want to walk around in an immersive visualization, crouch to observe a terrain contour, and then pick up the terrain and scale it to obtain a better detail view. To create convincing visualizations in head-tracked environments and avoid “simulator sickness,” our algorithms must be able to sustain frame rates upwards of 48 Hz per eye [Kreylos et al. 2001]. This constraint typically requires use of multiresolution and out-of-core visualization methods.

Direct Manipulation Once users detect a feature by observation, they must be able to measure or record the feature for further analysis. Our goal is to provide direct manipulation tools, i. e., tools that allow users to “touch their data” or manipulate data “at their fingertips.” This is in contrast to indirect tools such as buttons, dials or sliders provided through graphical user interfaces. For example, when detecting a fault line in a terrain model, a user should be able to directly sketch the line onto the 3D model using a 6-DOF input device. Our geoscience co-authors believe that this direct interaction is a key feature for effective VR visualization, and is necessary for the acceptance of VR as a scientific tool. Interaction also poses an algorithmic challenge, in that it must not interfere with the measurement process. To not break “suspension of disbelief,” an application must show the effects of an interaction within a time frame of about 0.1 s [Kreylos et al. 2001]. This requires the use of incremental visualization techniques such as seeded slices or isosurfaces, and real-time update of multiresolution data structures.

VR Infrastructure Complex VR applications cannot be developed from scratch; we need a powerful VR toolkit that hides implementation details from developers, offers a unified programming interface for a wide variety of VR environments, and offers methods to develop applications that implement a common “look and feel.” Our main goal was to create a toolkit that supports truly portable applications, from high-end immersive VR environments

down to standard desktop and portable computers. We found that most existing VR toolkits [Cruz-Neira et al. 2002; Stu n. d.], while successfully hiding the display setup of a VR environment – such as number and position of projected screens, rendering distribution, and view frustum generation, do not hide the input device environment at a high enough level. Although all contain or use drivers that hide the particular input device hardware [Taylor, II et al. 2001; Reitmayr and Schmalstieg 2001], most applications are still written for particular device layouts (such as CAVE-style wand, two data gloves, spaceball, joystick, etc.). As a result, an application developed for a two-glove environment will usually work at severely diminished capacity, if at all, in environments with different input device setups. Furthermore, while most toolkits offer “simulator modes” to run on standard computers, those are merely intended for debugging and do not allow scientists to use applications effectively. We decided instead to provide an abstraction that maps application functionality to “atomic interactions” that are defined by the toolkit and implemented by environment-specific plug-ins. These plug-ins support applications that are truly device-independent, and in many cases the desktop versions of applications developed for immersive VR are as usable as applications specifically designed for the desktop. Additionally, the ability to develop or choose custom plug-ins allows users to adapt the look and feel of all applications to their tastes, e. g., by choosing their favorite navigation or selection methods. In some aspects, our approach is similar to the data flow networks described by Shaw et al. [Shaw et al. 1993].

4 Implementation Challenges

Next, we describe the methods we used to provide real-time visualization and direct manipulation for each of our three case studies, and elaborate on how our VR toolkit supports portability across a wide range of environments.

4.1 Real-time Visualization

Our three application scenarios have in common that each requires scientists to inspect very large data sets. Due to VR’s stringent real-time constraints and the memory limitations of commodity graphics workstations, this typically requires using out-of-core multiresolution methods for visualization. Furthermore, due to our focus on interactivity, we must also ensure that the selected methods allow interactive manipulation of the displayed data.

In the geological mapping example, the source data are topography models (3D heightfields) and draped color images typically generated from satellite or aerial photography. To make accurate

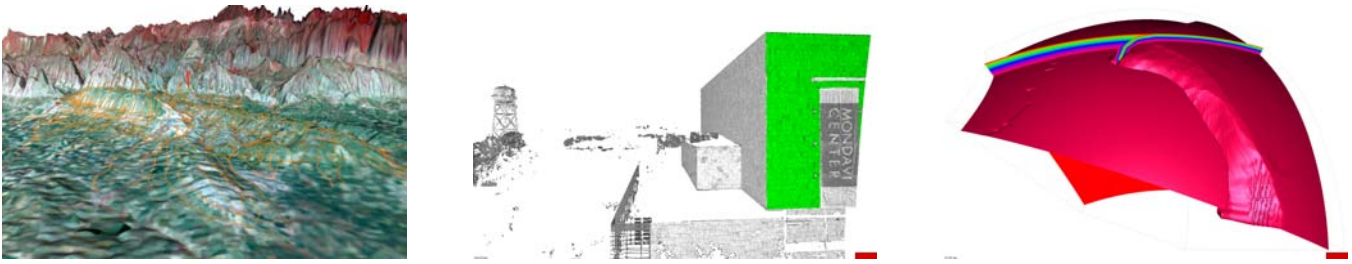


Figure 3: Measurements in VR environments. Left: Locations of faults, folds, and drainages precisely mapped directly onto the virtual topographic surface. Center: Selected point set (green) in 3D LiDAR scan to extract plane equation of the Mondavi Center wall, to measure precise relationships between objects or changes in location over time. Right: Color-mapped vertical slice and isosurface extracted from simulation data to characterize the geometry and dynamics of the subduction of ocean crust into the Earth’s mantle.

observations, researchers need models with around 1–10 m resolution and extents of several hundred kilometers. To enable real-time visualization of these multi-GB data sets, we employed an out-of-core multiresolution method based on a quadtree subdivision of a terrain model’s domain. Although other, and arguably more efficient, methods exist (see [Hwa et al. 2004] for an extensive list of references), the quadtree’s simplicity made it easier to integrate topography and color imagery, and to allow geologists to interactively create and manipulate geological maps directly on the 3D terrain model. These mapped lines and polygons are represented as sets of 2D vector primitives (annotated polylines), and are displayed as 3D geometry draped over the 3D terrain model. As the terrain’s displayed level-of-detail changes in response to user navigation, the 2D primitives are converted to 3D geometry on-demand. Raw input data are converted to a quadtree by first covering the heightfield’s or image’s domain with identical square tiles, and then building a hierarchy bottom-up by downsampling four adjacent tiles into a parent tile half the resolution until the entire domain is covered by a single root tile. The heightfield and image trees are stored in separate files, and are accessed in a top-down fashion during visualization. Whenever a displayed tile’s resolution becomes too low, its children are brought in from external storage by a background thread and displayed as soon as they are available. Two levels of caching (external storage to main memory, and main memory to graphics card memory) are employed to optimize rendering performance. The decision when to split a node is based on the projected size of the node (to ensure that model triangle size is on the order of display pixel size), whether a node intersects the view frustum (to enable frustum culling), and on a node’s distance from any interaction cursor (to enable “focus+context” visualization).

In the displacement analysis example, the source data are unsorted sets of attributed 3D points generated from merged laser scans. Measuring displacements of small objects with high accuracy requires hundreds of sample points per object; this results in very large point sets containing several million points. The UC Davis water tower data set contains about 15 M points; another data set of a landslide contains about 50 M points. Point-based visualization is an active research area [Amenta and Kil 2004; Fleishman et al. 2003; Alexa et al. 2001], but most approaches rely on sampling density assumptions that are typically not met by LiDAR data. To avoid introducing bias, we chose not to attempt to reconstruct surfaces from the point data, but instead to visualize the point sets directly using an out-of-core multiresolution approach based on octrees. These octrees are created in a pre-processing step by first assigning all points to a root node covering the entire point set’s domain, and then recursively splitting nodes that contain more than a preset number of points. Once the entire tree is constructed, lower-resolution point sets are created for interior nodes by randomly subsampling the union of their children’s point sets. The resulting octree

is stored in a file and accessed in a top-down fashion during visualization. Whenever the average point distance in a node becomes too large, its children are brought in from external storage and displayed instead. The caching schemes used to speed up rendering, and the decisions when to split nodes, are very similar to the terrain rendering algorithm.

The source data in the plate subduction simulation example are FEM grids produced by fluid dynamics simulations run on a remote computation cluster. Due to the grids’ irregular structures, we have not yet implemented out-of-core or multiresolution methods for their visualization; currently, data size is limited by the memory size of the display computers. Fortunately, the biggest data set produced by the simulations right now contains about 20 M nodes and fits easily into 1 GB of memory (leaving enough room for extracted visualization primitives such as isosurfaces). We were able to leverage pre-existing software to represent these data, and the main work has been to add new visualization functionality to the component framework described in [Kreylos et al. 2001]. Our approach generally follows the methods used in the Virtual Windtunnel [Bryson and Levit 1991]. The techniques we use to visualize these 3D data are color-mapped planar slices, and isosurfaces computed using an incremental version of the Marching Cubes algorithm [Meyer and Globus 1993; Lorensen and Cline 1987].

4.2 Direct Manipulation

We believe that immersive visualization is only one aspect of the benefits offered by VR, and we have therefore focused on developing intuitive ways to explore data by interacting with it. Interaction methods are naturally not only highly problem-specific, but also depend on the particular workflows employed by individual researchers. Our general goal was to provide *direct manipulation* where possible, i. e., to allow users to manipulate data by touching it in 3D space using 6-DOF input devices, as opposed to manipulating it indirectly using buttons, dials, or sliders via a graphical user interface. The ability to touch data requires a “near-field” approach to VR, where the data – or at least the user’s region of interest – appears in arm’s reach of a user (literally implementing the “data at your fingertips” mantra). This approach has typically been implemented using workbench-style VR environments; it is usually incompatible with theater-like immersive displays such as 3D IMAX and especially curved-screen displays. We found that flat or segmented tiled display walls and CAVEs are optimal environments due to their large fields of view and the accurate perspective provided by the combination of high-accuracy tracking systems and flat projection surfaces.

In our geological mapping application, the main interaction is to

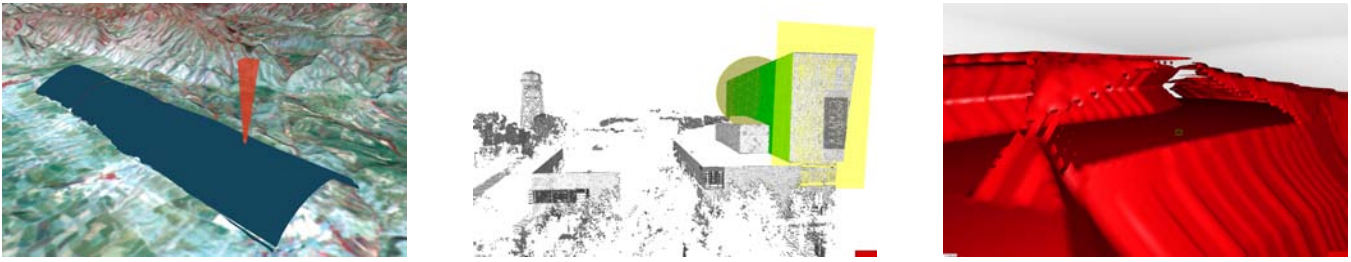


Figure 4: Insight gained from visualization in VR environments. Left: A 3D fold surface calculated from the virtually mapped data in Figure 3. Center: An extracted plane equation (yellow) compared to a second surface defined by the selected point set (green). Right: Isosurface showing aliasing in the simulation viscosity field that prevents the simulation from converging.

record observations by drawing vector-based mapping elements, such as polylines, directly onto a 3D terrain model. The interaction method is very straightforward: by pressing a button on a 6-DOF input device, users can create new vertices (or select and drag existing vertices) at the projection of the position of the input device onto the terrain surface. This projection can either be performed along the model's z (height) direction, or along a ray emanating from the input device in an appropriate pointing direction. The most important design objective is to allow users to switch seamlessly between navigation and mapping; observations are made by picking up the model and inspecting a region of interest from many different angles to perceive silhouettes, and then record those observations immediately. This process is intuitive using a multi-button device such as a CAVE wand; the optimal case seems to be an environment with two input devices where users can use one to pick up the model and the other to manipulate mapping elements at the same time.

The main interaction step in the LiDAR-based displacement measurement application is to extract features, by selecting the subset of sample points defining the surface of a particular feature. After a point set has been selected, users can request the computation of derived quantities, such as plane equations, sphere centroids and radii, or cylinder axes and radii, depending on the shape of the selected feature. Assuming that scanning adds random noise to the real position of surface points, quantities derived from sets containing several hundred points have higher accuracy than each individual point and allow displacement calculations between several scans of a single scene taken at different times. The end result of computing displacements of multiple features is a displacement vector field that leads to insight into the process that caused the displacement, such as an earthquake or landslide. The main difficulty in feature extraction is that features of interest are often small and have surface variabilities on the order of the sample spacing, and can be obscured by "point dust" resulting from undersampled surfaces such as foliage. As a consequence, point sets are typically selected in refinement steps: close examination of a region of a scan shows several points that should belong to the set; these points are then selected (and highlighted, as shown in Figures 3 and 4, center), and the result is examined again to improve selection. To facilitate this tight coupling of observation and selection, we followed a "sweep selection" paradigm that affixes a 3D brush, e. g., a sphere of appropriate radius (the yellow sphere shown in Figure 2, center), to a 6-DOF input device, and selects all points within the brush while an input device button is pressed. By changing the size of the brush and the selection mode (add vs remove), selected point sets can be refined in a very intuitive way. This is especially powerful in environments with two input devices such as data gloves: a user can affix the selection brush to one device, and use the other device to pick up and move the entire data set.

The typical process in exploring 3D volume data such as results

from plate subduction simulations is to extract visualization primitives such as slices, isosurfaces, or streamlines (as shown in Figures 3 and 4, right). Proper placement of these primitives can provide insight into the local or global structure of the examined data. In typical desktop visualization programs, extraction is guided by indirect interaction (such as entering an isovalue into a text field or selecting it via a dial or slider). Once extraction is initiated, the program typically blocks until the complete primitive is extracted and can be displayed. Depending on primitive type and data size, this can take several seconds or even minutes. The main problem with this approach is that primitives only show the structure of the examined data very close to the primitive. To obtain a more global understanding, users have to extract and display many primitives, which is time consuming and can lead to cluttering. An alternative approach is to support interactive extraction, where a user selects the parameters guiding primitive extraction via direct manipulation, and the (partial) result of extraction is displayed immediately – typically within 0.1 s. For example, an isosurface can be specified not by its isovalue, but by a point on its surface [Kreylos et al. 2001; Meyer and Globus 1993]. Using a 6-DOF input device, users can pick a point of interest inside the examined data set's domain, and the program will start extracting the isosurface containing that point starting at the picked point using *seeded isosurface extraction* performed by a background thread. After a preset amount of time, the partial extraction result is displayed. If the user has moved the input device in the meantime, the program immediately starts extracting a new isosurface; otherwise, it keeps expanding the current surface. This approach allows users to interactively drag an isosurface through a data set's domain, thereby gaining a more global understanding of a data set's structure without introducing clutter. The same approach can be used to extract slices based on the position and orientation of an input device, or to drag stream lines through a vector field. In our experience, observing the behavior of these "dynamic primitives" offers quick insight into a data set during initial exploration. Although the interaction itself is very straightforward, implementing incremental versions of common primitive extraction algorithms can be a challenge.

4.3 Portability

The wide range of different VR environment layouts and capabilities poses major challenges for developers wishing to create applications that run unchanged on a variety of platforms. The variability can be roughly classified into three somewhat orthogonal areas: (1) distribution model (single display, single-CPU/multi-pipe, multi-CPU/multi-pipe, cluster-based); (2) display model (single screen, multi-screen, head-mounted); and (3) input environment (mouse/keyboard, desktop devices such as spaceballs or joysticks, single 6-DOF device, multiple 6-DOF devices). The possible com-

binations are limitless. For example, IDAV has a cluster-based head-tracked multi-screen environment with a single 6-DOF input device and an additional head-mounted display for multi-person viewing. In order to manage this combinatorial explosion, a developer must rely on VR toolkits with high powers of abstraction. There are many existing toolkits, and most of them do an excellent job of hiding variety in areas (1) and (2). However, our emphasis on interactivity makes our software depend heavily on the available input environment, and we found existing toolkits to be lacking in this area. Their input abstraction is limited to hiding low-level differences in input device hardware, but they do not shield an application from the differences between an environment having two 6-DOF input devices and one having, say, a mouse/keyboard and a joystick. Due to the lack of input environment abstraction in available toolkits, we decided to base our VR software on *Vrui* (VR user interface), a toolkit that has been under development at IDAV for several years.

Vrui follows the typical approaches of hiding distribution and display models. When a VR application is started, it determines the configuration of the local environment by reading a system-wide configuration file. The application adapts itself to the distribution model by starting additional threads in a multi-CPU system, or by replicating itself onto all cluster nodes in a cluster-based system. Afterwards it establishes communication between all application threads or processes, and initializes the application itself. Next, it creates rendering resources (windows and OpenGL contexts) for all displays of each thread or process, and finally enters the main loop where program state updates in each process are interleaved with render cycles for all displays. Thus, by default, *Vrui* implements a “split-first” model where a distributed application is synchronized by running identical processes and providing identical input data. However, applications are free to request private communications channels to enable “split-middle” models, where an application splits itself into a master process and several render processes. This enables better utilization of computing resources and easy integration of higher-level abstractions like shared scene graphs. In theory, *Vrui* supports “split-last” approaches by running an application in single-CPU mode on top of a distributed OpenGL implementation such as Chromium [Humphreys et al. 2002], but this approach usually cannot compete due to interconnect bandwidth limitations.

The novel component of *Vrui* is its management of input devices. At the lowest level, it contains a device driver daemon that bundles any number of physical input devices into a single input device stream, and offers this stream to applications connecting locally or remotely via a socket. *Vrui* currently has its own device daemon, but it contains interfaces to talk to standard daemons such as *vrpn* [Taylor, II et al. 2001]. The difference to other toolkits is how applications connect to input devices. Usually, an application requests an input device by name, e.g., “wand” or “left glove,” and then installs callbacks to be notified of device events or polls the device in its inner loop, leading to non-portable applications. *Vrui*, on the other hand, introduces a middle layer of so-called “tools” that map input device events to higher-level *semantic events* such as “navigate,” “select,” or “drag.” From an application’s point of view, tools offer callback hooks that can call application functions when, for example, the user selects a point in space. The trick is that the mapping from input devices to semantic events is hidden from the application, and can be prescribed externally through configuration files, and even be changed by a user during a program’s run-time. Thus *Vrui* can provide tool implementations optimized for the local environment. For example, an intuitive navigation method using a single 6-DOF input device is to pick space by pressing a button and then to drag space by attaching the navigation coordinate frame to the input device’s coordinate system. In a mouse/keyboard environment, on the other hand, an agreed-upon navigation method is to

attach a virtual trackball, a screen-plane panner, and a virtual zoom slider to several mouse buttons. Using tools, interactions are performed in the optimal way for a given environment. Furthermore, this approach allows users to pick their own favorite method of performing particular interactions, and thus to customize the “look and feel” of the VR applications they use.

The tool approach has another important benefit: it overcomes the scarcity of buttons available on typical 6-DOF devices. When using direct manipulation, interactions are typically initiated by pressing a button on an input device. Complex programs offer a large palette of interactions, and these cannot all be mapped to individual buttons. The alternative of using GUI elements such as menus to select actions is often cumbersome and interferes with usage patterns where users need to seamlessly switch between several operations. Using tools, users can assign arbitrary application functions to arbitrary buttons and devices at run-time. For example, if a user anticipates that she has to perform a long sequence of selection and navigation interactions, she can assign a navigation tool to one button and a selection tool to another one, and these assignments can later be changed to accommodate changing usage patterns. It is important to note that this flexibility is entirely handled by the toolkit, and completely invisible to the application. As a side effect, the tool interface often leads to application designs that offer unintended but useful interactions, such as a user of our plate subduction simulation application being able to drag a dynamic slice and isosurface at the same time using two input devices.

5 Results

We have evaluated the effectiveness and efficiency of our approaches to enabling scientific workflows for VR by recording the experiences of our geoscience collaborators and collecting feedback from other users. Due to the specialized applications and the narrow target user base (geoscientists), we have not yet performed a formal user study, but anecdotal evidence shows that our approaches are in fact useful. In the following, we list the benefits that the geoscience co-authors on this paper and other users have identified when comparing our VR applications to their previously available approaches.

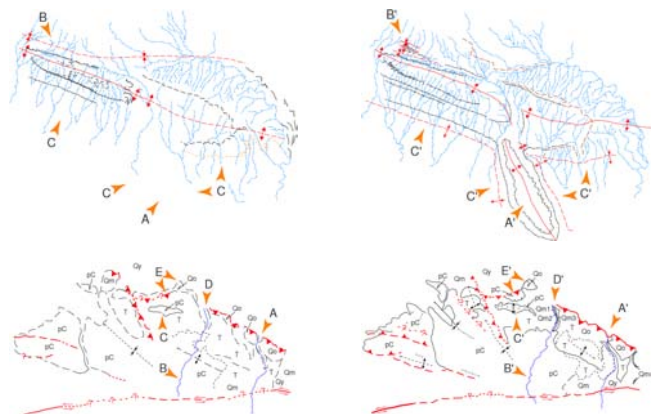


Figure 5: Results from comparison between Stereo Analyst (left column) and our VR mapping application (right column). Top row: sensitivity test comparing the accuracy and confidence of maps created with unlimited time. Bottom row: time test comparing the number of observations made in the same amount of time. The gold arrows highlight key differences between the result maps.

To determine the impact of using VR methods on geological mapping, we have performed experiments where several of our geology co-authors created a map of the same area using their previous system (Stereo Analyst [Ste n. d.]) and our new system. Our users had to perform two tasks: record all observations that could be made over an unlimited amount of time, and record all observations that could be made in a limited amount of time. The first test was to judge the effectiveness of our new system, i.e., the accuracy and confidence of the mapping (see Figure 5, top row); the second test was to judge its efficiency, i.e., the number of observations made in time (see Figure 5, bottom row). This informal study was not quantitative due to the small number of participants and the fact that each participant performed all tests, but it supports our hypothesis that VR visualization enables scientists to make more accurate observations in less time, and to be more confident about their observations.

Displacement analysis using LiDAR data essentially requires VR visualization and interaction. Since the sampled point sets are typically not high enough in resolution to reconstruct surfaces of small features, they can only be visualized as point sets to prevent the introduction of bias. Without stereoscopic display a user only sees an amorphous point cloud, and it is very hard even for experienced users to identify objects in still images. Users typically rely on motion parallax to provide depth clues by constantly spinning or panning the model, but this interferes with accurate point selection. In addition, selecting non-regular 3D point sets using 2D input devices is inherently difficult. Using stereoscopic displays, we find that we and most users immediately perceive 3D objects, and can identify the points defining features accurately. In combination with the intuitive point set selection, this results in a more effective and efficient process, where smaller features can be extracted more accurately in less time.

We have only recently started using our 3D data visualization system to explore plate subduction simulation data. The first results have come from using it as a debugging tool for FEM simulations. One of our co-authors was facing a convergence failure in her simulations of plate subduction in the Aleutian region, and could not find the cause of the problem using her previous set of tools. By exploring her data in the VR application, we quickly found several regions where one component of the simulation input exhibited severe aliasing, and were able to trace it back to the algorithms used when converting plate surface data reconstructed from subsurface earthquake locations into a 3D temperature and viscosity model. We have hence devised an alternative approach to model generation that will hopefully solve the problem (results are still pending). We view this as anecdotal evidence that our VR application is more effective than the previously used program, in that we were able to pinpoint a problem that had not been identified before.

6 Conclusion

We have presented an approach for turning immersive visualization into a scientific tool. We focused on transferring real-world skills into a virtual environment by providing real-time visualization and direct interaction, thereby allowing scientists to use their existing workflows in VR, and applying their skills to new problems. We have illustrated our approach for three distinct geoscience applications, which have in common the need for the human interaction with data to make scientific progress. The most important factor in the success of our described efforts has been the willingness of the team members to cross disciplinary boundaries in an effort to understand the driving scientific questions posed by large geoscience data sets and the capabilities and limitations of interactive VR. This

successful collaboration between geoscientists and information scientists has resulted in new approaches to visualization that take advantage of the skills of geoscientists while also exploiting the full capabilities of a VR environment.

We emphasize that the methods that make data exploration possible are not confined to state-of-the-art VR systems, but are adapted to a wide range of other visualization systems. Previously developed VR software has typically been limited to portability between VR systems with very similar input devices, limiting the ability of a research group with a low-end visualization system to scale up in a cost-effective manner. The value of immersive, interactive data exploration is growing more important with the explosion of large datasets created by imaging, large observational efforts, and high-resolution computer simulations. Great opportunities for further development exist especially for simulation applications. The ability to rapidly create complex objects for use in models allows the use of more realistic boundary conditions and objects in Earth science modeling. One of the most difficult aspects of developing forward models and simulations of earth science processes is identifying the spatial distribution of critical behaviors and the temporal framework of changes. Proper resolution is critical to modeling realistic behaviors. An ability to interactively adjust critical parameters in 3D models substantially increases the appropriateness of boundary conditions during model development, promoting rapid advances in model sophistication, accuracy, and relevance to natural Earth processes.

Acknowledgments This work was supported by the National Science Foundation under contract ACI 9624034 (CAREER Award), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, through the National Partnership for Advanced Computational Infrastructure (NPACI), and a large Information Technology Research (ITR) grant. We gratefully acknowledge the support of the W.M. Keck Foundation provided to the UC Davis Center for Active Visualization in the Earth Sciences (KeckCAVES), and thank the members of the Department of Geology and the Visualization and Computer Graphics Research Group at the Institute for Data Analysis and Visualization (IDAV) at the University of California, Davis.

References

- ALEXA, M., BEHR, J., COHEN-OR, D., FLEISHMAN, S., LEVIN, D., AND SILVA, C. T. 2001. Point set surfaces. In *Proceedings of the Conference on Visualization 2001 (VIS-01)*, IEEE Computer Society, Piscataway, NJ, T. Ertl, K. Joy, and A. Varshney, Eds., 21–28.
- AMENTA, N., AND KIL, Y. J. 2004. Defining point set surfaces. In *ACM, ACM Press, 1515 Broadway, New York, NY 10036*, vol. 23 of *ACM Transactions on Graphics (TOG)*, ACM, 264–270.
- BAWDEN, G. W., SCHMITZ, S., HOWLE, J. F., LACZNAK, R. J., BOWERS, J., OSTERHUBER, R., AND IRVINE, P. 2005. Four-dimensional surface deformation analysis, snow volume calculation, and fault mapping with ground based tripod LiDAR. *Eos Trans. AGU* 86, 52, Fall Meet. Suppl., Abstract G33D-07.
- BILLEN, M. I., AND GURNIS, M. 2003. A comparison of dynamic models in the Aleutian and Tonga-Kermadec subduction zones. *Geochemistry, Geophysics and Geosystems* 4, 4, 1035, doi:10.1029/2001GC000295.
- BRYSON, S., AND LEVIT, C. 1991. The Virtual Windtunnel: An environment for the exploration of three-dimensional unsteady flows. In *Proc. of Visualization '91*, IEEE Computer Society Press, Los Alamitos, CA, 17–24.
- CHEN, S. 2004. A prototype of virtual geographical environment (VGE) for the Tibet Plateau and its applications. In *Geoscience and Remote Sensing Symposium, 2004. Proceedings. IEEE International*, vol. 5, 2849–2852.

- CRUZ-NEIRA, C., SANDIN, D. J., AND DEFANTI, T. A. 1993. Surround-screen projection-based virtual reality: The design and implementation of the CAVE. In *Proceedings of SIGGRAPH 93*, Computer Graphics Proceedings, Annual Conference Series, 135–142.
- CRUZ-NEIRA, C., BIERBAUM, A., HARTLING, P., JUST, C., AND MEINERT, K. 2002. VR Juggler: An open source platform for virtual reality applications. In *40th AIAA Aerospace Sciences Meeting and Exhibit*.
- DI CARLO, W. 2000. Exploring multi-dimensional remote sensing data with a virtual reality system. *Geographical & Environmental Modelling* 4, 1 (May), 7–20.
- ERLEBACKER, G., YUEN, D. A., AND DUBUFFET, F. Current trends and demands in visualization in the geosciences. *Electronic Geosciences*.
- EVANS, F., VOLZ, W., DORN, G., FRÖHLICH, B., AND ROBERTS, D. M. 2002. Future trends in oil and gas visualization. In *VIS '02: Proceedings of the conference on Visualization '02*, IEEE Computer Society, Washington, DC, USA, 567–570.
- FLEISHMAN, S., COHEN-OR, D., ALEXA, M., AND SILVA, C. T. 2003. Progressive point set surfaces. In *ACM Transactions on Graphics*, J. C. Hart, Ed., vol. 22(4). ACM Press, 997–1011.
- GARDNER, J. V., WARNER, T., NELLIS, M. D., AND BRANDTBERG, T. 2003. Virtual reality technology for lidar data analysis. In *Geo-Spatial and Temporal Image and Data Exploitation III*, N. L. Faust and W. E. Roper, Eds., vol. 5097 of *Proceedings of SPIE*, 48–57.
- GOLD, R. D., COWGILL, E., WANG, X.-F., AND CHEN, X.-H. 2006. Application of trishear fault-propagation folding to active reverse faults: examples from the Dalong fault, Gansu province, NW China. *Journal of Structural Geology* 28, 200–219.
- GROSS, M., LAMBORAY, E., NÄF, M., WÜRMLIN, S., SPAGNO, C., KUNZ, A., LANG, S., STREHLKE, K., VANDE MOERE, A., ENGELI, M., VAN GOOL, L., KOLLER-MEIER, E., SVOBODA, T., AND STAADT, O. 2003. blue-c: A spatially immersive display and 3d video portal for telepresence. *ACM Transactions on Graphics* 22, 3 (July).
- GRUCHALLA, K. 2004. Immersive well-path editing: investigating the added value of immersion. In *Virtual Reality, 2004. Proceedings. IEEE*, 157–164.
- HARDING, C., KAKADIARIS, I. A., AND LOFTIN, R. B. 2000. A multimodal user interface for geoscientific data investigation. In *Advances in Multimodal Interfaces - ICM I 2000, Third International Conference, Beijing, China, October 14-16, 2000, Proceedings*, 615–623.
- HEAD, III, J. W., VAN DAM, A., FULCOMER, S. G., FORSBERG, A., PRABHAT, ROSSER, G., AND MILKOVICH, S. 2005. ADVISER: Immersive scientific visualization applied to mars research and exploration. *Pothogrammetric Engineering & Remote Sensing* 71, 10 (October), 1219–1225.
- HUMPHREYS, G., HOUSTON, M., NG, R., FRANK, R., AHERN, S., KIRCHNER, P. D., AND KLOSOWSKI, J. T. 2002. Chromium: a stream-processing framework for interactive rendering on clusters. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, 693–702.
- HWA, L. M., DUCHAINEAU, M. A., AND JOY, K. I. 2004. Adaptive 4-8 texture hierarchies. In *Proceedings of IEEE Visualization 2004*, Computer Society Press, Los Alamitos, CA, IEEE, 219–226.
- JOHNSON, C. 2004. Top scientific visualization research problems. *Computer Graphics and Applications, IEEE* 24, 4, 13–17.
- JORDAN, K. E., YUEN, D. A., REUTELER, D. M., ZHANG, S. X., AND HAIMES, R. 1996. Parallel interactive visualization of 3D mantle convection. *IEEE Computational Science & Engineering* 3, 4, 29–37.
- KREYLOS, O., BETHEL, E. W., LIGOCKI, T. J., AND HAMANN, B. 2001. Virtual-reality based interactive exploration of multiresolution data. In *Hierarchical Approximation and Geometrical Methods for Scientific Visualization*, G. Farin, H. Hagen, and B. Hamann, Eds. Springer-Verlag, Heidelberg, Germany, 205–224.
- KREYLOS, O., BAWDEN, G. W., AND KELLOGG, L. H. 2005. New visualization techniques to analyze ultra-high resolution four-dimensional surface deformation imagery collected with ground-based tripod LiDAR. *Eos Trans. AGU* 86, 52, Fall Meet. Suppl., Abstract IN43B–0332.
- LIN, C.-R., AND LOFTIN, R. B. 1998. Application of virtual reality in the interpretation of geoscience data. In *VRST '98: Proceedings of the ACM symposium on Virtual reality software and technology*, ACM Press, New York, NY, USA, 187–194.
- LIN, C.-R., LOFTIN, R. B., AND STARK, T. 1998. Virtual reality for geosciences visualization. In *Proceedings of 3rd Asia Pacific Computer Human Interaction Conference*, 196–201.
- LIN, C.-R., LOFTIN, R. B., AND NELSON, JR., H. R. 2000. Interaction with geoscience data in an immersive environment. In *Virtual Reality, 2000. Proceedings. IEEE*, 55–62.
- LORENSEN, W. E., AND CLINE, H. E. 1987. Marching Cubes: A high resolution 3D surface construction algorithm. In *Proc. of SIGGRAPH '87*, ACM, 163–169.
- MEYER, T., AND GLOBUS, A. 1993. Direct manipulation of isosurfaces and cutting planes in virtual environments. Tech. Rep. CS–93–54, Brown University, Providence, RI.
- REITMAYR, G., AND SCHMALSTIEG, D. 2001. OpenTracker: An open software architecture for reconfigurable tracking based on XML. In *Virtual Reality 2001 Conference*.
- ROSS, L. E., KELLY, M., AND SPRINGER, A. E. 2003. GeoWall use in an introductory geology laboratory: Impacts in student understanding of field mapping concepts. *Eos Trans. AGU* 84, 46, Fall Meet. Suppl., Abstract ED31A–07.
- SHAW, C., GREEN, M., LIANG, J., AND SUN, Y. 1993. Decoupled simulation in virtual reality with the MR toolkit. *ACM Transactions on Information Systems* 11, 3 (July), 278–317.
- SIMON, A. 2005. First-person experience and usability of co-located interaction in a projection-based virtual environment. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology, VRST 2005, Monterey, CA, USA, November 7-9*, 23–30.
- SPRAGUE, K. B., AND DE KEMP, E. 2005. Interpretive tools for 3-D structural geological modelling. Part II: Surface design from sparse spatial data. *Geoinformatica* 9, 1 (Mar.), 5–32.
- Stereo Analyst for ArcGIS.
<http://gis.leica-geosystems.com/Products/StereoAnalyst/>.
- STEINWAND, D., DAVIS, B., AND WEEKS, N. 2002. GeoWall: Investigations into low-cost stereo display systems. Tech. Rep. 03–198, USGS Open File Report.
- The Studierstube home page.
<http://www.cg.tuwien.ac.at/research/vr/AVS/html/>.
- TAYLOR, II, R. M., HUDSON, T. C., SEEGER, A., WEBER, H., JULIANO, J., AND HELSER, A. T. 2001. VRPN: a device-independent, network-transparent VR peripheral system. In *Proceedings of the ACM symposium on Virtual reality software and technology*, ACM Press, New York, NY, 55–61.
- THURMOND, J. B., DRZEWIECKI, P. A., AND XU, X. M. 2005. Building simple multiscale visualizations of outcrop geology using virtual reality modeling language (VRML). *Computers & Geosciences* 31, 7 (Aug.), 913–919.
- TUFTE, E. 1983. *The Visual Display of Quantitative Information*.
- TURCOTTE, D. L. 1997. *Fractals and Chaos in Geology and Geophysics*, 2nd ed. Cambridge University Press.
- VAN DAM, A., FORSBERG, A. S., LAIDLAW, D. H., LAVIOLA, J. J., J., AND SIMPSON, R. M. 2000. Immersive VR for scientific visualization: A progress report. *Computer Graphics and Applications, IEEE* 20, 6, 26–52.