Interactivity for Immersive Volume Visualization

Chahat Kalsi
Stony Brook University
NY, US
ckalsi@cs.stonybrook.edu

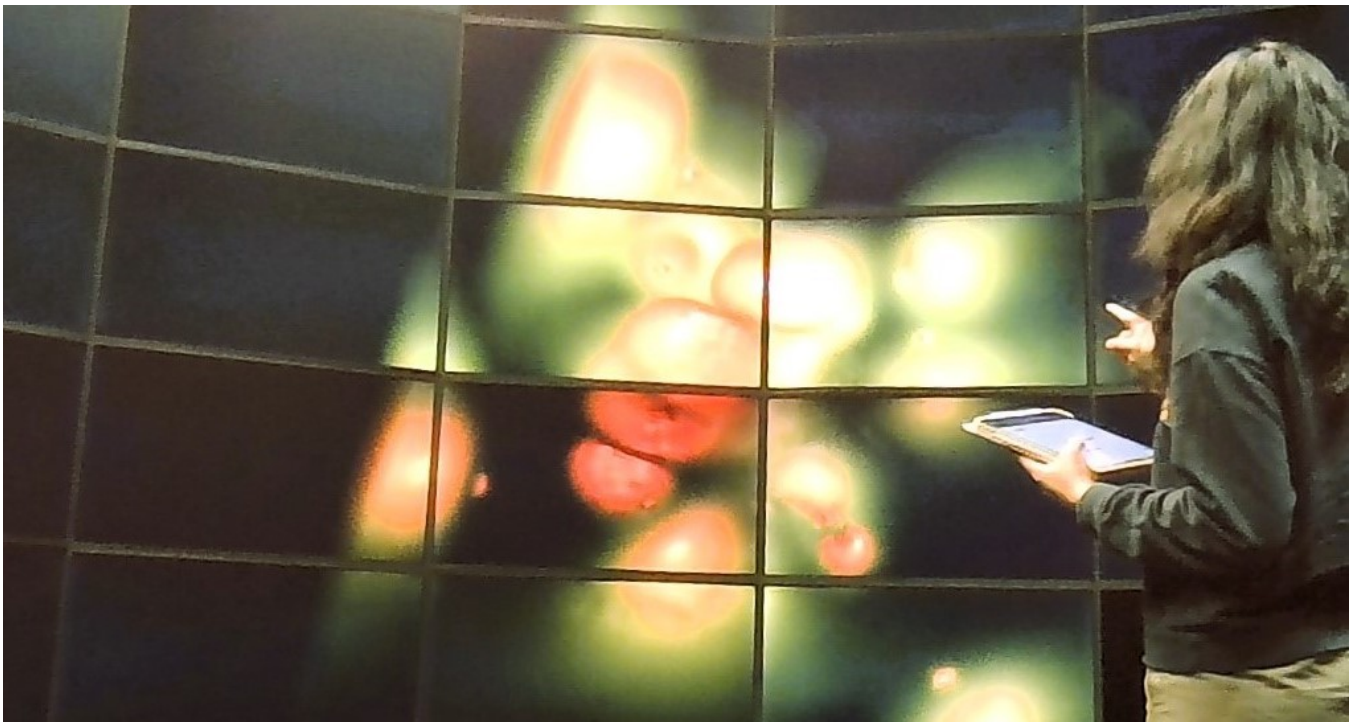

Figure 1: A user investigating the volume visualization of the electron potential of an iron protein molecule on Silo - an immersive facility - along with the interaction interface on an iPad device.

Abstract

This paper investigates the design principles and essential considerations for effective volume visualization in immersive environments. It presents an interactive interface optimized for touch-based devices, facilitating key volumetric interactions, including navigation, application of transfer functions, adjustment of blend modes, and management of applications within immersive facilities. By leveraging the intuitive and portable nature of touch interfaces, the framework enhances user engagement and control in immersive settings. The proposed solution leverages Silo, a fully stereoscopic cylindrical tiled-display visualization facility, which provides a near 360-degree field of regard, serving as an exemplary platform to demonstrate the approach. A user study is conducted that compares the efficacy of the proposed solution versus the popular volume visualization tool ParaView [1].

1 Introduction

Immersive facilities present unique interaction challenges that require special considerations. The scale and field of regard they provide demand interaction methods that can accommodate large, multi-user displays and diverse user perspectives. Moreover, interaction interfaces must balance precision with usability, enabling both novice and expert users to engage meaningfully with the visualization [9].

A promising approach to address these challenges is the development of tablet-based, touch-driven interfaces that integrate multiple interaction modalities [2]. Such interfaces combine the portability and familiarity of touchscreens with the ability to provide fine-grained control over complex volumetric interactions. By consolidating functionalities like navigation, transfer function editing, and application management into a unified interface, tablet-based systems offer an intuitive and powerful way to interact with immersive environments.

To validate this approach, we conducted a user study with domain experts comparing our tablet-based interface in an immersive Silo environment against a traditional setup using the popular volume visualization tool ParaView as shown in Figure 3. The study revealed promising insights: while completion times varied, our interface demonstrated consistently higher accuracy, particularly

in complex manipulation tasks. For instance, in isosurface identification, participants achieved a 100% accuracy rate using our interface compared to 83% with ParaView. This empirical evidence underscores the potential of our human-computer interaction (HCI) approach in enhancing volumetric data exploration.

This paper presents the design and development of an interaction framework for immersive volume rendering, tailored for use in facilities like Silo (Figure 2). It highlights the need for HCI-driven solutions in this domain and demonstrates how a tablet-based interface can meet the unique demands of immersive visualization systems, supported by initial user study findings that suggest significant usability improvements.

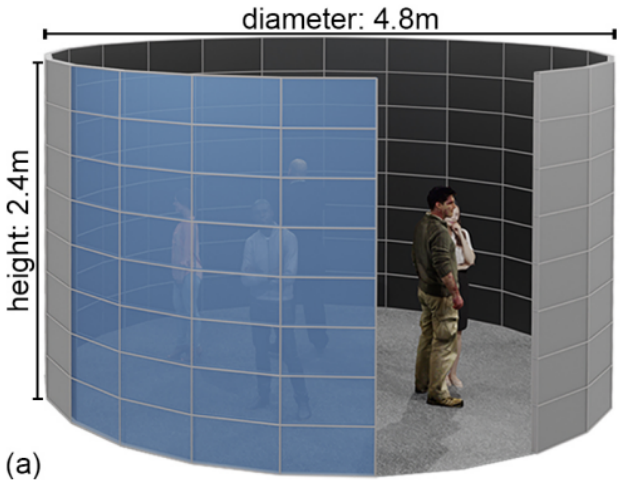

Figure 2: Silo is a immersive visualization facility, a 3D rendering of which is shown here. It is constructed in the shape of a cylinder with 21 columns, with each column being 8 displays tall, bringing the total number of displays in this facility to 142.

2 Related Works

Immersive volume rendering is a rapidly evolving domain that merges advanced rendering techniques with innovative interaction paradigms to enable intuitive exploration of volumetric data. Significant advancements have been made in immersive technologies such as head-mounted displays (HMDs) [3] and large-scale visualization systems, but gaps remain in understanding how to integrate real-time interactivity, particularly in CAVE-like [5] environments such as the Reality Deck [7] Silo. This section reviews related work on immersive facilities, volume rendering, and interaction techniques, while highlighting the unique challenges that remain unaddressed.

2.1 Immersive Visualization Facilities

Immersive visualization facilities have been a cornerstone of scientific visualization, offering environments that enhance spatial perception and provide deeper insights into complex datasets. Systems like the CAVE (Cave Automatic Virtual Environment) [6] by the Electronic Visualization Laboratory (EVL) have pioneered this space by combining large-scale projection environments with stereoscopic shutter glasses and head tracking to create highly immersive experiences. Extensions of this concept, such as the Silo and similar tiled-display systems, provide massive visual resolutions for exploring intricate datasets, making them invaluable for domains such as volume visualization.

2.2 Interactivity for Volume Rendering

Interaction is pivotal to making volume rendering truly accessible and insightful. Existing approaches for immersive interaction have largely focused on HMDs, leveraging hand tracking, gaze-based controls, and touch interfaces to manipulate volumetric parameters such as transfer functions, blend modes, and camera navigation. For example, Taibo and Iglesias-Guitian [10] introduced a real-time system for editing volume appearance and transfer functions in VR, showcasing the potential for interactive cinematic rendering.

In contrast, interaction in CAVE-like systems remains an open challenge. While these systems offer unmatched collaborative and spatial capabilities, traditional interaction paradigms, such as mouse and keyboard inputs, are ill-suited to their immersive nature. Jadhav and Kaufman [4] proposed a workbench for radiologists that integrates head tracking and shutter glasses for interacting with DVR visualizations in large projection environments, but real-time interactivity with volume rendering in CAVE-like systems has yet to be comprehensively addressed.

2.3 Volume Rendering Tools

ParaView. Paraview [1], developed using the Visualization Toolkit (VTK) [8], stands as one of the most widely used open-source tools for scientific visualization and volume rendering. Despite its popularity, ParaView exhibits several significant limitations that constrain user interaction and visualization flexibility. Notably, the tool provides only a single, fixed camera speed, which restricts users' ability to navigate volumes at their preferred pace. In contrast, our developed interface allows dynamic speed control, enabling more fluid and personalized camera movements.

The transfer function editing in ParaView further demonstrates these constraints. Users are limited to selecting from preset color maps and cannot directly assign colors to individual control points, a significant usability barrier for precise visualization. Our interface addresses this limitation by providing an intuitive color picker that allows users to assign specific colors to each control point, offering unprecedented granularity in transfer function design.

Perhaps most critically, ParaView is fundamentally designed for traditional desktop environments and lacks native support for immersive visualization systems. This limitation necessitates an entirely new interaction paradigm for large-scale, multi-display facilities like Silo. Our research directly tackles this challenge by developing a touch-based, tablet-driven interface specifically engineered for immersive volume rendering environments, bridging a crucial gap in current visualization technologies.

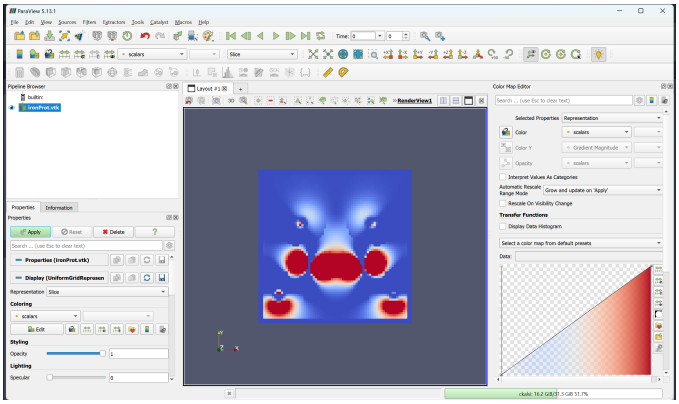

Figure 3: Volume visualization on ParaView.

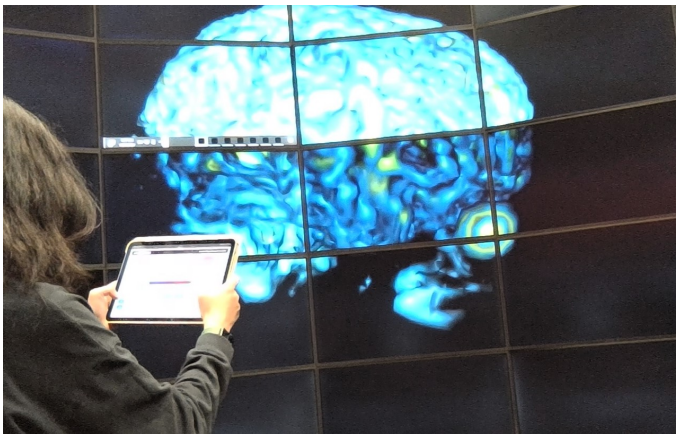

Figure 4: A user interacting with the MRI scan of a human brain, dynamically adjusting the transfer function.

3 Software and Design

Software Architecture. To develop volume rendering applications for the Silo, we implemented a framework based on the Visualization Toolkit (VTK). This framework employs a replicated execution model [14], where individual instances of the application run independently on each Silo node. A master instance is responsible for synchronizing key components, such as application controls, tracking, and communication, across all nodes and instances.

The framework uses a configuration file to define the physical layout of the Silo displays, the initial position of the virtual scene, and the camera viewport for each application instance. Additionally, system-specific information, such as the machine name and allocated GPU for each node, is specified. Assigning GPUs to individual instances ensures that their respective OpenGL contexts utilize the designated GPU for rendering and processing, optimizing performance.

To minimize synchronization overhead on visualization nodes, a dedicated node—equipped with the same computing capabilities—acts as the master or "head" instance. This head instance manages application synchronization and hosts the Flask backend server, which provides a touchpad-compatible frontend interface. This interface facilitates interaction with the volume rendering application by relaying user inputs to the appropriate rendering client instances. To ensure scene coherence across all nodes, the master periodically broadcasts synchronization data to client instances. This mechanism ensures that any out-of-sync instances are promptly updated, maintaining a consistent rendering experience across the entire Silo environment.

The touchpad-based interface, illustrated in Figure 5, includes tools for controlling the virtual camera's position and orientation, a graphical transfer function editor for real-time adjustments to the transfer function, and client management utilities. This design enables users to seamlessly interact with the volume rendering application and customize their visualization experience dynamically.

Applications. VVolume rendering is used to visualize complex 3D data sets. This technique is used in a variety of fields, including medical imaging, scientific visualization, and VFX. One application is shown in Figure 4, where the user is investigating the MRI scan of a head inside the immersive environment Silo.

4 Interactivity Interface

The volume rendering interactions are enabled through an interface optimized for touch-based tablet devices. This design choice considers that such devices are the most practical and intuitive options for interaction within immersive facilities like the Silo. The components of the interface are detailed in the following subsections.

4.1 A. Client Manager

The Client Manager provides an overview of the currently connected volume rendering instances running in an immersive environment, such as the Silo. It includes tools to quickly launch or terminate the execution of these instances for any specific environment.

A1. The "Env" dropdown, labeled A1 in Figure 5, allows users to easily launch volume rendering instances for any specific environment, such as the Silo.

A2. Instances are labeled as "Node: x, Instance: x," where x indicates the node and instance number, respectively. This labeling, shown as A2 in Figure 5, helps users identify which instance is being referenced. For example, "Node: 1, Instance: 1" corresponds to the first instance running on the first node. Each instance has a red cross icon next to it, enabling users to terminate that particular instance immediately.

4.2 B. Transfer Function Editor

Transfer functions play a critical role in volume rendering by determining how voxel intensities are mapped to color and opacity, offering vastly different representations of the same dataset. The Transfer Function Editor is designed to facilitate dynamic and intuitive adjustments.

B1. Opacity Editor. The Opacity Editor is a D3.js-based graph where voxel intensities are represented on the X-axis and their corresponding opacities on the Y-axis. Users can dynamically add

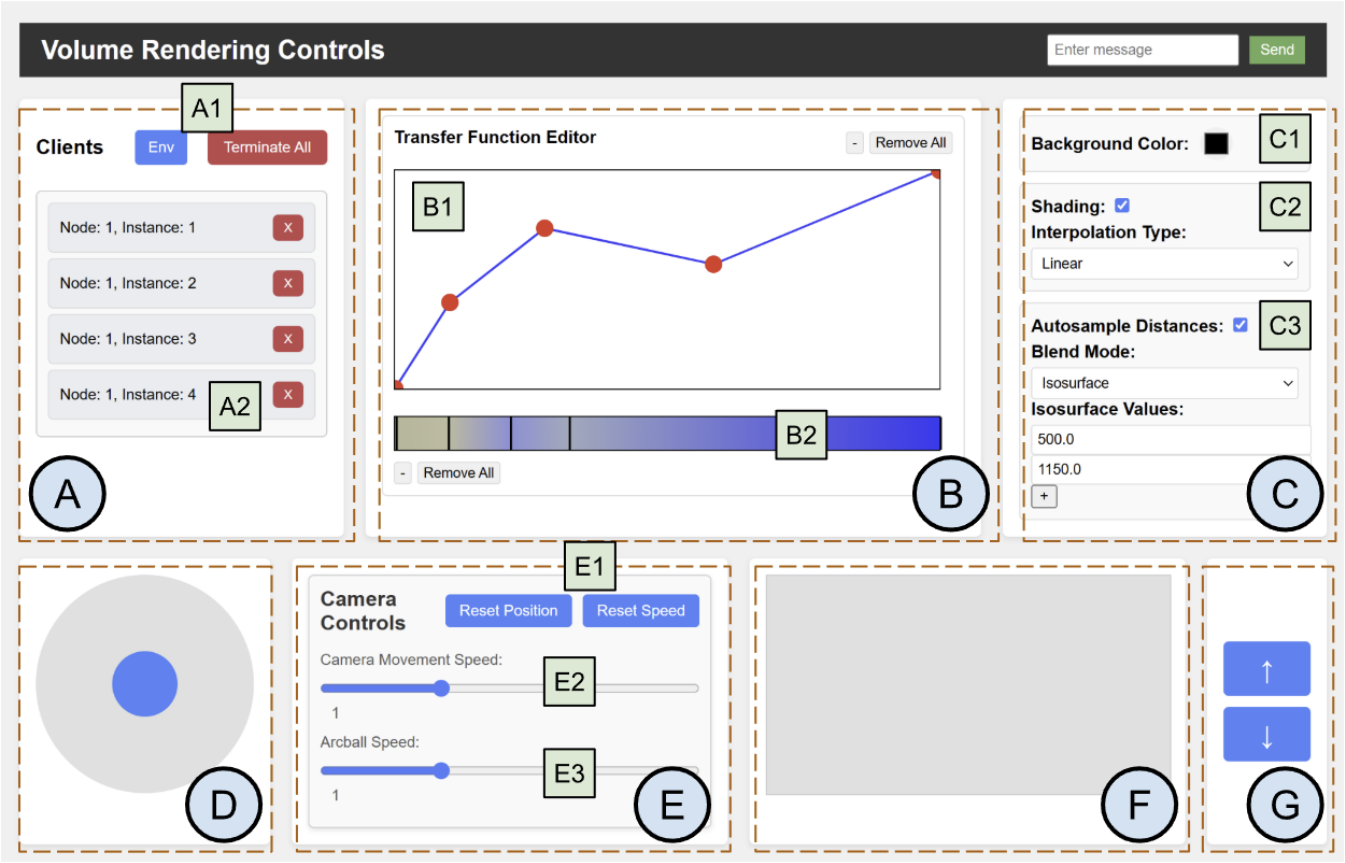

Figure 5: The touch-based interface for immersive volume visualization, showcasing immersive client management, navigation and transfer function editing tools. The Client Manager (A) shows the volume rendering client instances currently controlled by the server. The Transfer Function Editor (B) gives an interactive interface to change opacity and color of the volume. The Volume Controls panel (C) allows the dynamic change of various volumetric parameters. The Free Camera (D) and (G) allow for camera movement along all three primary axes. The Arcball Camera (F) provides camera movement focused on the volume's center. The Camera Controls panel (E) provides extra controls over camera movement and position.

control points to define specific intensity-opacity mappings. Each new control point is connected by a line segment, visually cuing the user to the opacity gradient. By default, two control points are provided: one at the minimum intensity with zero opacity and another at the maximum intensity with full opacity. Users can interact with the graph by selecting a control point (indicated in green), dragging it to a new position, or removing it using the '-' button that appears in the top-right corner of the Opacity Editor. Additionally, a "Remove All" button resets the graph to its default state.

B2. Color Editor. The Color Editor is a rectangular bar aligned with the Opacity Editor. It allows users to dynamically change the voxel colors by adding control points, represented as partitioning lines. By default, the left (minimum intensity) and right (maximum intensity) ends are colored red (#FF0000) and blue (#0000FF), respectively. Users can add new partitions by clicking on the bar. A newly created partition adopts the color of its immediate left neighbor but can be changed using an HTML color picker that appears upon selection, as shown in Figure 5. Colors between partitions are linearly interpolated, offering smooth transitions across the intensity range.

4.3 C. Volume Controls Panel

The Volume Controls Panel enables users to adjust various parameters related to volume rendering.

C1. Background Color. An HTML color picker allows users to change the background color dynamically. This is particularly useful for improving visibility or creating aesthetically pleasing representations for specific volumetric renderings.

C2. Shading, Interpolation Types and Sample Distances. Shading enhances the perception of depth and structure in volumes by applying lighting effects. For example, shading is useful for highlighting intricate structures but can be disabled for techniques like minimum or maximum intensity projections. The shading is performed

on the client side using VTK's default volumetric lighting model, with a light source attached to the camera for consistent visibility.

Volumetric interpolation determines how voxel values are calculated between grid points. The Interpolation Type dropdown offers two options:

(1) Linear Interpolation (Default). Smoothly interpolates values between neighboring voxels, producing a more realistic and continuous appearance.
(2) Nearest Neighbour Interpolation. Uses the value of the closest voxel, resulting in a blocky but computationally efficient representation, useful for large datasets.

Sample distance refers to the spacing between sampling points along each ray during volume rendering. Smaller distances improve rendering quality but increase computational cost. By default, the "Autosample Distances" checkbox dynamically adjusts sample distances based on the volume's properties. Unchecking this option allows expert users to manually input a custom sample distance.

C4. Blend Modes. Blend Modes determine how voxel intensities are combined during volume raycasting. The interface supports the following modes:

(1) Min Intensity. Retains the lowest intensity encountered along each ray.
(2) Max Intensity. Retains the highest intensity encountered along each ray.
(3) Average Intensity. Computes the average of all intensities along the ray.
(4) Additive. Sums voxel intensities along the ray.
(5) Composite (Default). Combines intensities along the ray based on their opacity values.
(6) Isosurface. Visualizes specific intensity thresholds as surfaces.

For the Isosurface blend mode, additional input fields allow users to specify intensity thresholds. Users can add multiple thresholds with the '+' button, and their colors and opacities can be adjusted via the Transfer Function Editor. This Blend Mode is shown in Figure 5.

4.4 D. and G. The Free Camera Controls

The Free Camera provides unrestricted movement along the X, Y, and Z axes. The controls are ergonomically placed at the bottom-left (D) and bottom-right (G) of the interface for easy thumb access.

D. Joystick. The joystick simulates movement along the X and Z axes. Up-down movements correspond to in-out motion (Z-axis), while left-right movements control lateral motion (X-axis). Diagonal movements are a combination of the two.

G. Up and Down Buttons. These buttons enable vertical movement along the Y-axis. Holding the buttons allows continuous motion.

4.5 F. The Arcball Camera

TThe Arcball Camera focuses on a specific target point, allowing users to rotate around it intuitively. The touchpad-shaped panel lets users interact with the volume by dragging their fingers. Pinching gestures enable zooming in and out, offering fine control over the viewing distance.

4.6 E. Other Camera Controls

The interface includes additional tools to streamline camera interactions:

E1. Reset Position and Reset Speed. The "Reset Position" button returns the camera to its default position, calculated based on the volume's center and bounds. The "Reset Speed" button restores the Free and Arcball camera speeds to their default values of 1.

E2. Free Camera Speed. A slider allows users to adjust the Free Camera speed dynamically. The default speed is 1, and the current value is displayed beneath the slider.

E2. Arcball Camera Speed. Similar to the Free Camera, this slider modifies the Arcball Camera speed, with a default value of 1 and the current speed displayed beneath the slider.

The interface thus combines intuitive touch-based controls with advanced features like transfer function editing, camera navigation, and real-time parameter adjustments to provide a seamless and efficient interaction experience. Designed specifically for immersive environments like Silo, it bridges the gap between user accessibility and the complexity of volume visualization, enabling users to explore and analyze volumetric data effectively.

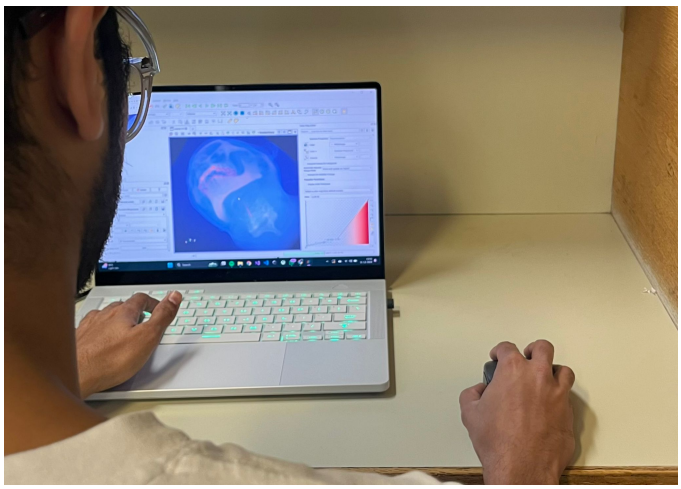

Figure 6: User Study task T1 on ParaView - a user is navigating through the spinal cord in the volume on ParaView.

5 User Study

We conducted a user study to evaluate the effectiveness of the developed interactive interface for volume rendering tasks in immersive environments, compared to a single-computer display using ParaView. Specifically, the study analyzes quantitative metrics to assess how these environments impact user performance and interaction during navigation and manipulation tasks.

5.1 Study Design

Our user study involved two tasks, detailed below. Participants were instructed to complete each task as efficiently and accurately

Figure 7: User Study task T2 on Silo - a user is visualizing isosurfaces for the skin, skull bone and tooth enamel.

as possible, reflecting typical activities in the domain of volume rendering. The immersive environment the touch based interface is being tested on is Silo. The interface frontend was hosted on an iPad which the user took inside Silo to perform the study tasks. For ParaView, an ASUS RoG Zephyrus laptop with dimensions of 31.1 x 22.0 x 1.59 1.63 cm (12.24" x 8.66" x 0.63" 0.64") is being used.

Table 1: User Study Tasks

Task	Description
T1: Navigation and Exploration	Participants were asked to navigate through a CT scan of the head and count the number of vertebrae in the spine during the process.
T2: Manipulation and Adjustment	Participants were tasked with identifying isosurfaces roughly corresponding to skin, skull bone, and tooth enamel, and rendering them as distinct layers.

The study employed a within-subject design, where each participant completed the tasks using the developed interactivity interface on an iPad within Silo and then on ParaView on a laptop device. To mitigate potential learning effects or biases, the order of tasks was randomized for each participant.

For the quantitative analysis, we measured two key metrics: the time taken to complete each task (Completion Time) and the correctness of the responses (Accuracy). Accuracy for **T1** was determined by accurately counting the number of vertebrae in the spine. For **T2**, a more comprehensive scoring system was used to assess effectiveness. Participants earned:

- 1 point for correctly identifying each isosurface (skin, bone, enamel).
- 2 points if all three isosurfaces were rendered distinctly.
- 1 point if only one or two isosurfaces were rendered distinctly.

This scoring system resulted in a total possible score of 5 for the second task, reflecting the user's ability to accomplish the required objectives.

5.2 Participants

The study was conducted with 3 participants (1 woman and 2 men) aged between 23 and 26, all of whom were domain experts in volume rendering. Participants had between 1 and 2 years of experience in volume rendering, with an average of 1.3 years. They were well-versed in terminology such as "isosurfaces," which was relevant to the tasks in the study. All participants were screened to ensure normal or corrected-to-normal vision.

All participants had prior experience with ParaView, ranging from 1 to 6 months, with an average of 2.6 months of familiarity with the tool. However, since they were not previously acquainted with the developed interactivity interface, they were introduced to it a month before the study and encouraged to familiarize themselves with its features. Prior to conducting the study, participants self-assessed their proficiency with the new interface, with all reporting a proficient level of skill.

The participants were given the option to entirely suspend the study if they experienced cybersickness or take a break if preferred. None of the participants opted to do so.

5.3 Results

We assess the user performance in terms of task completion times and success rates. Overall, the user studies conclude that the task completion times using the developed interface are higher, however, the accuracy of the tasks performed is consistently higher as well, achieving a 100% success rate for **T2**.

To evaluate the statistical significance of the observed variations between environments, we performed a one-way ANOVA for both completion times and accuracy.

Completion Time. Figure 8 illustrates the completion times for Task 1 (T1), revealing that the developed interface on Silo consistently performed similarly to ParaView on a laptop. The average task completion time on Silo was 245 seconds, compared to 228 seconds on the laptop. For Task 2 (T2), as depicted in Figure 10, the interface demonstrated more efficient performance, with an average completion time of 407 seconds versus 494.67 seconds for ParaView.

Accuracy. The accuracy results showed notable differences between the two platforms. For T1, the interface using Silo achieved an average accuracy of 88.67%, compared to 78% for ParaView, as visualized in Figure 9. Even more pronounced was the performance in T2, where the developed interface reached a perfect 100% accuracy, while ParaView achieved 83%, as shown in Figure 11.

ANOVA. An analysis of variance (ANOVA) was conducted to compare task completion times across the two environments: Silo with the interactive interface and a laptop using ParaView. The results showed no statistically significant difference between the environments, $F(1, 14) = 0.27, p = 0.63$. The proportion of variance explained by the environment (η^2) was 0.062, indicating that only 6% of the variance in task completion time is attributable to the environment. A separate ANOVA was performed to analyze accuracy scores for **T1** across the environments. The results showed no statistically significant difference, $F(1, 14) = 0.74, p = 0.44$. The environment explained 15% of the variance in accuracy ($\eta^2 = 0.15$). Further, ANOVA was conducted to compare task completion times for **T2**

across the two environments: Silo with the interactive interface and a laptop using ParaView. The results showed no statistically significant difference between the environments, $F(1, 14) = 2.74, p = 0.17$. However, the proportion of variance explained by the environment ($\eta^2 = 0.41$) suggests that 41% of the variance in task completion time is attributable to the environment, indicating a potentially meaningful effect that warrants further investigation. A separate ANOVA was performed to analyze accuracy scores for **T2** across the environments. The results were not statistically significant, $F(1, 14) = 3.06, p = 0.16$. The environment explained 43% of the variance in accuracy ($\eta^2 = 0.43$), suggesting a medium effect size despite the lack of statistical significance.

Boxplots showing the completion times and accuracies for both **T1** and **T2** in both environments are presented in Figure 12 and Figure 13.

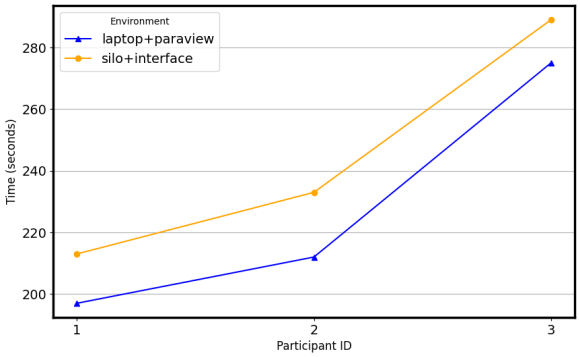

Figure 8: Task 1 completion times. The orange line shows the completion times for the developed interface on an iPad when used in Silo, and the blue line is for completion times on a laptop device using ParaView.

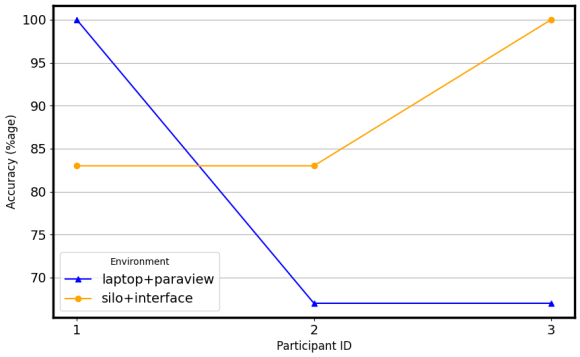

Figure 9: Task 1 accuracies. The orange line shows the completion times for the developed interface on an iPad when used in Silo, and the blue line is for completion times on a laptop device using ParaView.

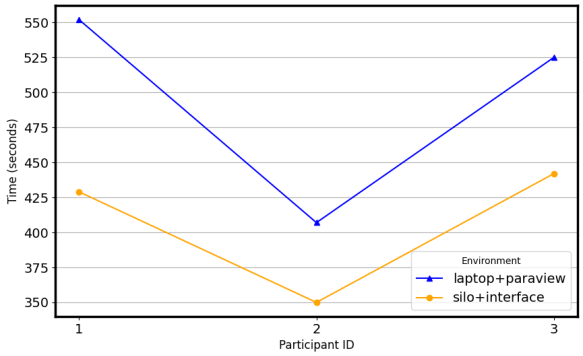

Figure 10: Task 2 completion times. The orange line shows the completion times for the developed interface on an iPad when used in Silo, and the blue line is for completion times on a laptop device using ParaView.

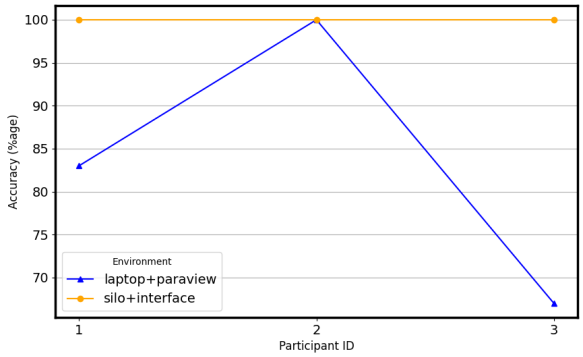

Figure 11: Task 2 accuracies. The orange line shows the completion times for the developed interface on an iPad when used in Silo, and the blue line is for completion times on a laptop device using ParaView.

5.4 Discussion

For **T1**, the navigation task, the completion time on Silo using the interface is consistently longer than that on ParaView. However, this increased time was observationally mostly due to the camera speed controls provided in the interface. Users chose to slow the speed down and navigate carefully through the spine, whereas in ParaView, where such controls are not provided, users had no choice but to pass through the volume at the quick interaction speed of the ParaView camera. This deliberation on the part of users can be witnessed in their accuracy, where the developed interface consistently performs better than ParaView.

For **T2**, both the completion times and accuracy are lower for the developed interface compared to ParaView. This is due to the easier usage of the color editor in the interface compared to ParaView. While ParaView only allows users to pick between specific color maps, users had to apply various color maps to produce the most

visually distinct isosurfaces after finding the isosurface values. In contrast, in the developed interface, users were quickly able to assign colors directly to control points, which increased the ease of the task and resulted in better completion times and accuracies.

Although ANOVA shows that the observed variation is not statistically significant, this failure to reject the null hypothesis may be a result of the small number of users involved in the study. ANOVA suggests that especially **T2** could potentially show greater significance in variance with more study participants.

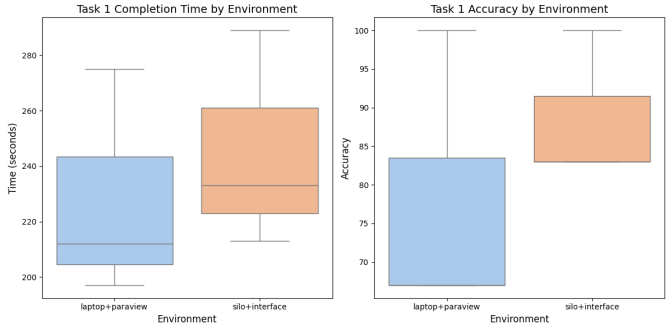

Figure 12: Boxplots for Task 1 completion times and accuracies.

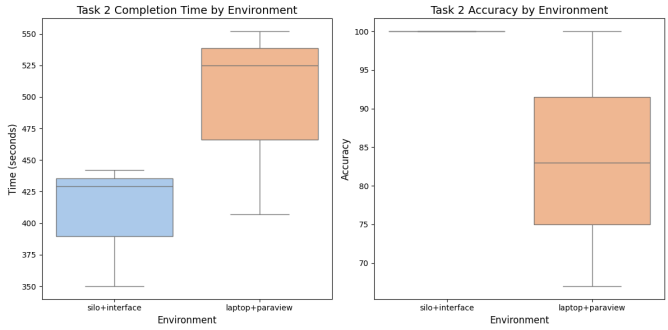

Figure 13: Boxplots for Task 2 completion times and accuracies.

6 Future Work

Even though this represents a significant milestone for advancing HCI-based research in immersive volume rendering, much remains to be explored.

Regarding the interactivity interface, numerous potential enhancements can be considered. These include a dropdown menu allowing users to cycle through various volume demonstrations, range sliders in the opacity or color editor for more precise color selections, the ability to specifically render NaN values with custom colors, and improved grouping of instances by nodes in the client manager. An intriguing potential addition is the integration of Extended Reality (XR), where users could gain additional context by pointing their iPad towards the virtual world. Head-tracking-based movements also present an exciting avenue for future development.

Exploring more sophisticated methods to enhance focus and context within volumes represents another promising research direction. This could involve allowing users to place markers within the volume or developing intelligent pathfinding algorithms between these markers.

Transfer functions emerge as another compelling area of research for future investigation. While the current interface demonstrates a simple two-dimensional transfer function, exploring multi-dimensional transfer functions could significantly extend the system's functionality.

From a user study perspective, more comprehensive research is needed. This could include rigorous comparisons testing the effectiveness of immersive versus non-immersive flat environments for volume rendering, or conducting heuristic-based evaluations of the interface. Such approaches would be crucial steps towards developing more naturalistic and human-centered interactions.

7 Conclusion

This research introduces an innovative touch-based interactive interface for immersive volume rendering, specifically designed for large-scale visualization facilities like Silo. By addressing the critical human-computer interaction challenges inherent in immersive environments, the developed interface demonstrates the potential for more intuitive and engaging volumetric data exploration.

The user study revealed promising insights into the interface's performance. While the completion times varied across tasks, the developed interface consistently showed higher accuracy, particularly in complex manipulation tasks. The camera speed controls and direct color assignment capabilities highlighted the interface's potential to provide more deliberate and precise interactions compared to traditional volume rendering tools.

The framework's modular design, leveraging VTK and a replicated execution model, offers a flexible approach to synchronizing and rendering volumetric data across large-scale display environments. By integrating touch-based interactions with advanced features like transfer function editing and comprehensive camera controls, the research demonstrates a significant step towards making immersive volume visualization more accessible and user-friendly.

As volumetric datasets continue to grow in complexity and size, the need for intuitive, interactive visualization tools becomes increasingly critical. This research contributes to the emerging field of human-centered design in scientific visualization, offering a foundation for future developments in immersive data exploration technologies.

8 References

References

[1] Utkarsh Ayachit. *The ParaView Guide: A Parallel Visualization Application*. Clifton Park, NY, USA: Kitware, Inc., 2015. ISBN: 1930934300.

[2] Paul George et al. "Nomad devices for interactions in immersive virtual environments". In: *The Engineering Reality of Virtual Reality 2013*. Ed. by Margaret Dolinsky and Ian E. McDowall. Vol. 8649. International Society for Optics and Photonics. SPIE, 2013, p. 86490I. DOI: 10.1117/12.2008451. URL: https://doi.org/10.1117/12.2008451.

[3] Claudia Hänel et al. "Visual Quality Adjustment for Volume Rendering in a Head-Tracked Virtual Environment". In: *IEEE Transactions on Visualization and Computer Graphics* 22.4 (2016), pp. 1472–1481. DOI: 10.1109/TVCG.2016.2518338.

[4] Shreeraj Jadhav and Arie E. Kaufman. "MD-Cave: An Immersive Visualization Workbench for Radiologists". In: *IEEE Transactions on Visualization and Computer Graphics* 29.12 (2023), pp. 4832–4844. DOI: 10.1109/TVCG.2022.3193672.

[5] Siddhesh Manjrekar et al. "CAVE: An Emerging Immersive Technology – A Review". In: *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation* (2014), pp. 131–136. URL: https://api.semanticscholar.org/CorpusID:15840690.

[6] Muhanna A. Muhanna. "Virtual reality and the CAVE: Taxonomy, interaction challenges and research directions". In: *Journal of King Saud University - Computer and Information Sciences* 27.3 (2015), pp. 344–361. ISSN: 1319-1578. DOI: https://doi.org/10.1016/j.jksuci.2014.03.023. URL: https://www.sciencedirect.com/science/article/pii/S1319157815000439.

[7] Charilaos Papadopoulos et al. "The Reality Deck–an Immersive Gigapixel Display". In: *IEEE Computer Graphics and Applications* 35.1 (2015), pp. 33–45. DOI: 10.1109/MCG.2014.80.

[8] W.J. Schroeder, L.S. Avila, and W. Hoffman. "Visualizing with VTK: a tutorial". In: *IEEE Computer Graphics and Applications* 20.5 (2000), pp. 20–27. DOI: 10.1109/38.865875.

[9] Becky Spittle et al. "A Review of Interaction Techniques for Immersive Environments". In: *IEEE Transactions on Visualization and Computer Graphics* 29.9 (2023), pp. 3900–3921. DOI: 10.1109/TVCG.2022.3174805.

[10] Javier Taibo and Jose A. Iglesias-Guitian. "Immersive 3D Medical Visualization in Virtual Reality using Stereoscopic Volumetric Path Tracing". In: *2024 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*. 2024, pp. 1044–1053. DOI: 10.1109/VR58804.2024.00123.