Immersive ParaView: An Immersive Scientific Workflow for the Advancement of Measurement Science

Simon Su*
National Institute of Standards and Technology
William R. Sherman†
National Institute of Standards and Technology
Israel Lopez-Coto‡
National Institute of Standards and Technology
Kamran Sayrafian§
National Institute of Standards and Technology
Judith Terrill¶
National Institute of Standards and Technology

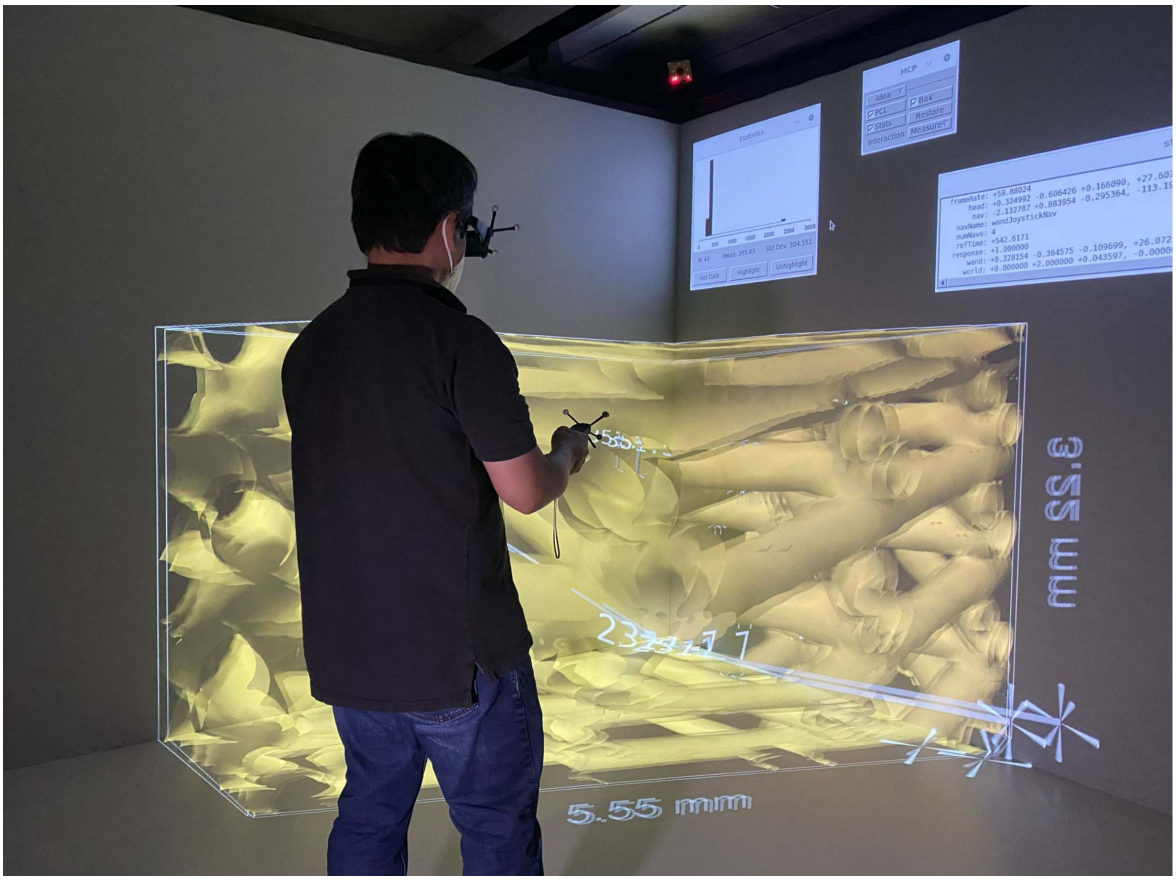

Figure 1: Measuring Tissue Engineering Data in Immersive Environment. Photo: Steven Satterfield, NIST.

ABSTRACT

We present ongoing work to enhance ParaView with new immersive visualization capabilities and demonstrate its use in the development of our scientific workflow in support of visual analytics in an immersive environment to advance measurement science. Two case-studies of complex meteorology and radio-frequency (RF) data as part of the knowledge discovery process reveal the benefits of interactively exploring the three-dimensional data with immersive technologies such as the CAVE and head mounted display virtual reality. By adding support of immersive technology into ParaView, researchers can more naturally investigate the data of three-dimensional simulations. These efforts to provide tools for immersive visualization are guided by the need to advance measurement science, standards, and technology. To make these tools more broadly accessible and impactful, we promote the use of software standards related to visualization and immersive systems.

Index Terms: Human-centered computing—Visualization—Visualization appliation domains—Scientific Visualization ; Human-centered computing—Visualization—Visualization systems and

*e-mail: simon.su@nist.gov
†e-mail: william.sherman@nist.gov
‡e-mail: israel.lopezcoto@nist.gov
§e-mail: kamran.sayrafian@nist.gov
¶e-mail: judith.terrill@nist.gov

2771-1110/22/$31.00 ©2022 IEEE
DOI 10.1109/ISMAR-Adjunct57072.2022.00034

tools—Visualization toolkits

1 Introduction

The latest advancements in commodity graphics hardware and visualization research, together with immersive technologies (virtual reality, etc.), offer ever-increasing opportunities to meaningfully explore scientific results through interactive and immersive interfaces. At the National Institute of Standards and Technology (NIST), we are developing a virtual laboratory where researchers can interactively measure and analyze scientific data through immersive visualization [22]. The virtual laboratory moves normal visual analytics activities scientists use to understand their data into an immersive environment.

Data at NIST comes from a wide variety of sources ranging from tissue engineering [5] to RF energy simulation [12] to meteorological models [9] that exhibit complex dynamics. Our virtual laboratory combines the qualitative with the quantitative in an immersive environment as shown in Fig. 1 where a researcher analyzing data is able to obtain qualitative values using the virtual ruler on the virtual tissue, for example. We use visual representation, interactive selection, quantification, and different visualization renderings to access numerical information from each dataset. Our virtual laboratory illustrates this approach with a variety of software frameworks [10] that demonstrate all these methods used to interactively obtain quantitative knowledge.

The overall challenge in the development of our virtual laboratory remains very similar to the major issue encountered in the development of our Virtual Hydrology Observatory project [19] developed using VRJuggler virtual reality development toolkit [7] and VRFire project [17] developed using FreeVR virtual reality development toolkit [16]. In both projects, three-dimensional interactive visualization of High-Performance Computing generated data were mainly used to visualize the outcome of a computer simulation. The ability to gather quantitative measurement from the experiments in an immersive environment were challenging, which can be attributed to a combination of factors including the scaling factor of the data in a virtual environment, the precision and resolution of the interaction techniques supported, and overall user perception of the experiment in a virtual environment compared to the experiment in the real environment.

Among the challenges in developing a virtual laboratory is determining how to visualize complex data such that it provides support for trusted decision making. Also crucial is to keep on top of new technological frontiers as standards and technologies evolve. Given that data from our NIST collaborators spans a wide range of styles and content, we likewise need access to a variety of multi-modal and high-dimensional data visualization techniques accessible from within our immersive environment. Another important challenge is to address human factors and usability concepts in how scientific measurement can best be useful. While our existing, and established High-End Visualization (HEV) framework (based on DIVERSE [10]) has proven itself over the years, we have found that it only supports re-usability and abstraction at the immersive technology layer. Beyond that, we often must implement a new visualization workflow from scratch for each project. This is a common challenge faced by immersive analytics researchers when selecting software a framework in the development of new projects (further discussed in our Computer in Science and Engineering paper on the topic [20]). The result then, is that a significant level of effort is needed for the High-Performance Computing and Visualization Group to support each science research project at NIST.

Thus, the group is shifting its approach to take advantage of an open-source software that runs on a desktop environment as well as in an immersive environment —ParaView. ParaView is an open-source, multi-platform data analysis and visualization application. It is an integral tool in many national laboratories, universities and industrial settings. Thus, we now address the immersive visualization challenges for the scientific visualization community by adding immersive technology to the already widely used existing scientific visualization software tool. ParaView's immersive additions bring together decades of scientific visualization capabilities with modern advancements in immersive technologies. The result is an easy-to-adopt immersive visualization tool.

Our goal in adopting an approach based on extending the capabilities of already widely used desktop tools is to improve upon how we can address existing and new virtual laboratory development challenges. We are beginning to investigate the use ParaView-generated visualizations as the instrument for exploration and representation of data. As part of our workflow, we provide ways to include analysis and quantitative feedback into the visualization. Furthermore, we are exploring how to express uncertainty values as an integral part of the visualization methods.

2 Related Work

Our current development represents a collaboration between NIST and Kitware Inc. to expand the open-source VTK [15] and ParaView [6] interfaces to immersive systems. VTK was introduced to the visualization community when the 1st edition of the creator's book came out in 1996 [15]. ParaView followed a few years later, spurred by U.S. National Energy Laboratories need for a VTK-based visualization tool with a graphical user interface (GUI). Wider adoption of ParaView began around 2007.

VTK is an open-source visualization toolkit created in 1993. It is a software-rendering application programming interface (API) capable of reading many datasets and producing different styles of graphical outputs, but is not an end-user application. In 1998, VTK began to be maintained by Kitware Inc. while remaining open-source. ParaView is an open-source tool for end-users created as a collaboration between Kitware, Los Alamos National Laboratory and Sandia National Laboratory in a project that began in 2000, with the first public release in 2007.

While virtual reality has only recently been available in the consumer market, there is a long history of VTK and ParaView interfaces to experimental VR systems once only found in larger research institutions. For VTK, a vtkActortoPF [13] class was developed in 1997 to allow developers to translate any vtkActor to a pfGeode which is part of Iris Performer [14]. Through the Iris Performer interface to CAVE-style VR systems, immersive applications could make use of VTK for visualization support. Many other features of Iris Performer could also be applied to VTK scenes, such as rendering on multiple graphics channels, organizing data in scene-graphs, performing intersection testing, and visibility culling. In 2006, the vjVTK [8] API was developed to allow the use of VTK natively within the VRJuggler virtual reality development toolkit. In 2010, vtkFVR was developed as a series of VTK classes for virtual reality environments which use the FreeVR virtual reality development toolkit.

VTK continues to be expanded with new interfaces, such as the somewhat recent (2010) addition of off-axis rendering of the vtkCamera class; and more recently, the 2015 addition of the vtkRenderingExternal [3] class, which supports integration of VTK in external rendering systems (including VR systems). VTK is also available to other virtual reality development tools like Unity through two assets on the Unity Asset Store (VTKUnity-MedicalViewer and VTKUnity-Activiz).

For ParaView, the integration of immersive visualization capabilities started around 2011 with introduction of a *VRPlugin* designed primarily to support immersive visualization in CAVE immersive environments. In addition to adding head position tracking, and some controller interaction, the proper implementation of rendering for CAVE and tiled displays was made possible by the expansion of the vtkCamera class to accommodate off-axis rendering. In 2016,

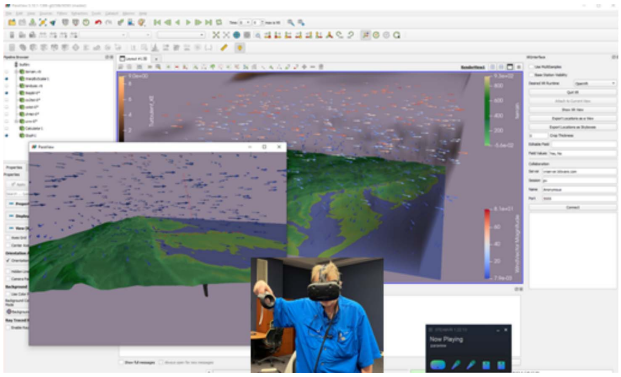

Figure 2: Running OpenXR in ParaView. In the background is the ParaView desktop interface, with the user's view inset at the bottom-left, and a photograph of the immersed user in the bottom-center inset. Photo: Simon Su, NIST.

shortly after the release of the first consumer-ready VR displays, a new plugin to handle OpenVR and Oculus systems [4] was introduced to ParaView for HMD-style VR. Recently the HMD-style plugin has been refactored to support the Khronos OpenXR standard (while maintaining the ability to interface with OpenVR [2]).

3 Immersive ParaView with CAVE-Interaction and XR-Interface Plugins

ParaView is an open-source scientific visualization software for visualizing two-dimensional and three-dimensional data sets. It is a complex software system and internally it uses a pipeline and proxy-based framework to enable multi-system parallel computations. ParaView uses Qt to provide a graphical front-end interface and uses over 2000 VTK C++ classes to produce the visualizations. ParaView runs on both desktop and in immersive mode across multiple platforms. In general, ParaView provides a bridge from researcher desktops to the immersive environments of our virtual laboratory, and beyond that, it also provides access to real time ray tracing and global illumination now possible with modern GPUs.

We started our work on immersive visualization with ParaView by developing an immersive plugin for ParaView by extending ParaView to support off-axis and head-position tracked perspective rendering [18]. (As the first ParaView plugin to provide an immersive interface, this plugin was initially named the *VRPlugin*. This term became ambiguous with the addition of the HMD-style plugin, so as of ParaView version 5.11, the pair of plugins have been renamed the *CAVE Interaction* and *XR Interactor* plugins [2].) The transition of our virtual laboratory workflow is simplified by integrating into ParaView data translators that read data created by our past projects. The combination of a utility to convert the general NIST SAVG format (NIST/HEV internal format) into a simplified subset with a ParaView reader plugin of that subset ensures minimal interruption to existing scientific workflow.

In collaboration with Kitware Inc. (the maintainer of VTK and ParaView), the OpenXR runtime option has been added to the already-existing OpenVR ParaView plugin. This allows ParaView to interact with the Khronos XR interface standard as shown in Fig. 2. The implementation involved a refactoring of the vtkOpenVRRenderer class to a higher level vtkVRRenderer class from which the OpenVR and OpenXR (via vtkOpenXRRenderer) interfaces are derived. As with most vtk classes, these modules have both a C++ and Python interface. By using the latter, quick VR programs can be written in the VTK API. To support the ability to travel through the scene, but also have fixed objects, and objects connected to tracked devices, a new coordinate system choice was added to the vtkActor

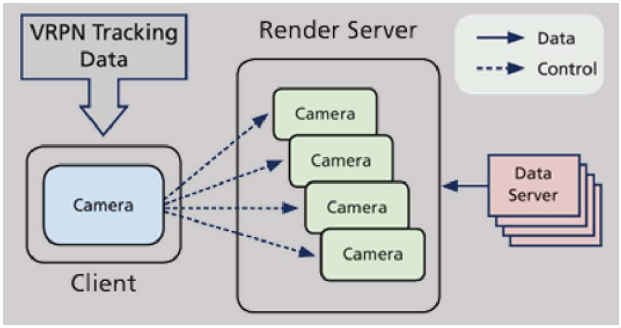

Figure 3: The Architectural Design of the *CAVE-Interaction plugin*.

class that links object representations to one of these coordinate systems.

In addition to single-user interactions, the OpenVR ParaView plugin includes a collaboration mode where several people can be together in the same virtual world. This collaboration mode is also available to the OpenXR runtime, and indeed can work between different choices in runtime. (Presently, the collaboration mode does not work with the *CAVE-Interaction* plugin, though we plan to rectify that soon.) Using the MQTT pub/sub paradigm, one user can run a private server to support the collaborative analysis session. Only minimal data transport is need as the *XR-Interactor* plugin needs just enough information for the location and orientation of all the participants' avatars to be displayed in the collaborative analysis session.

The *XR-Interactor* plugin (for OpenVR and OpenXR) works by duplicating the world from the ParaView Desktop view and rendering it through the VR runtime. It does not, however, include the full ParaView desktop interface within that view. While, of course, the full interface is still available on the desktop itself, a slightly simplified version of the interface can be summoned into the virtual world, and interacted with by pointing the controllers instead of moving the mouse.

The OpenXR option of the XR-Interaction plugin works on both Linux using Monado [1] (Linux open-source OpenXR runtime) and MS-Windows using SteamVR (also MacOS). ParaView operating in CAVE-style with the *CAVE-Interaction* plugin likewise can operate on the Linux, MS-Windows or even MacOS systems.

The ParaView *CAVE-Interaction* (formerly *VRPlugin*) plugin supports immersive visualization for CAVEs, Tiled-walls, Fishtank type of VR systems and consists of two semi-independent components as shown in Fig. 3; *CAVE-Interaction* plugin and Tiled-screen rendering. The *CAVE-Interaction* Plugin provides head and hand tracking along with controller interactions. The plugin obtains positional and orientation tracking data from either a VRPN [21] or a Vrui [11] tracking server. Different interaction "styles" are available, including the basic "Track" style which provides user-perspective rendering. Other styles include the "Grab" style for grabbing and moving objects around, plus a handful of other experimental user interaction styles.

For head-tracked perspective rendering (or even just to align multiple screens for tiled rendering), screen positions are defined with a "PVX" file which is used by ParaView's `pvserver` renderservers to specify the rendering parameters of the VR systems. It leverages VTK off-axis rendering capability (added as part of the initial effort) and also provides stereoscopic rendering support for both active and passive stereo modes.

One difference between the implementation of the *XR-Interactor* plugin with the `pvserver` rendering method is that the `pvserver` has direct access to the internal ParaView desktop rendering, whereas the *XR-Interactor* plugin makes separate copies of the data. One

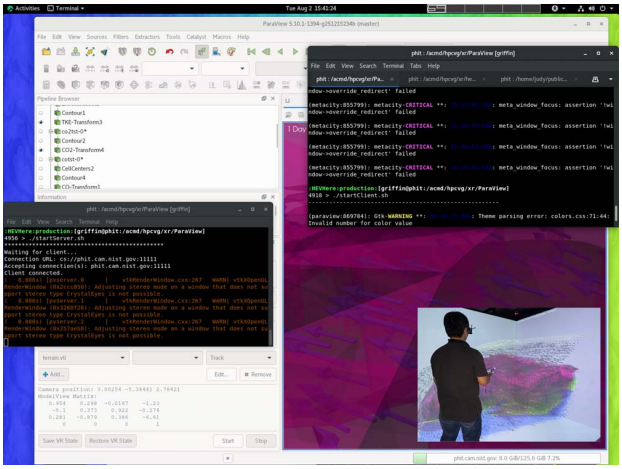

Figure 4: Running ParaView with the *CAVE-Interaction plugin*. Shown here is the visualization of time-series Meteorology Data in the CAVE with visualization pipeline setup in ParaView GUI (client process) connecting to the Render Server (server process) running the displays in the CAVE (lower-right inset). Photo: Steven Satterfield, NIST.

Figure 5: A workable home "mini-CAVE" can be creating using low cost tracking from a consumer VR system. Here a Vive puck worn on the head, along with the standard hand-held controllers enable the user to interact with the ParaView *CAVE-Interaction* plugin. Photo: William Sherman, NIST.

significant difference to the user-experience is that time-series data visualization works well in the *CAVE-Interaction* plugin, but not so well for the *XR-Interactor* plugin. Fig. 4 illustrates visual analytics of our time-series meteorology data on our CAVE systems at NIST using the *CAVE-Interaction* plugin. As mentioned, the *CAVE-Interaction* plugin also works on VR systems running on Linux, MS-Windows, and even MacOS platforms.

One of the in-development features of both plugins is support for different frames-of-reference. Using the new implementation of the physical world, virtual world, and tracked device coordinate systems mentioned above, we add the ability to connect objects in the virtual world to tracked devices, or to travel through the virtual world while some objects (in physical world coordinates) remain stationary. This can be thought of as a proto-scene graph capability.

Another feature of the NIST SAVG format is time-based animation. We have created a translator from these timeline files to the ParaView time animation system through python scripting. Our present HEV timeline methodology is built on a sequence of time deltas with changes in location, orientations and scale that together create a path through a visualization. Interspersed throughout a timeline file are "SAVG commands". These commands alter the visualization by changing the visibility of parts of the visualization (e.g. turning them off or on), or recording an image snapshot at a particular time —snapshots can then be merged into an animated movie file. These features provide ParaView the capability to re-run visualization sessions.

Finally, the `pvserver` "PVX" file combined with the *CAVE-Interaction* plugin and some low-cost head and controller tracking from a consumer VR system, we can also deploy a low-cost hardware setup that provides a home (or on the road) "fishtank-VR" system which functions as a "mini-CAVE" as shown in Fig. 5. This is accomplished by using existing open-source tools along with consumer 6-DOF position tracking systems such as the HTC/Valve Corp "Lighthouse" system. With these tools, collaborators working without the benefit of a full CAVE system can still work with the full user interface of a CAVE benefiting both development and deployment of the immersive visualization features (though perhaps without stereoscopic rendering).

4 Case Studies: Visual Analytics of Meteorology and Radiowave-Organ Interaction Data in the Immersive Environment

A primary goal of the NIST High-Performance and Visualization Group is to improve the scientific workflow by reducing the turnaround time of immersive visualization development. Ideally, we would empower practicing scientists to directly perform their own visual analytics in an immersive environment. Here, we present our experiences working with two different groups of researchers at NIST in pursuit of these goals.

In our collaboration with meteorology researchers, the data is produced by a WRF simulation [9] of one day's atmospheric transport over Chesapeake Region as shown in Fig. 6. The data set consists of 96 time steps with a 15 minute intervals from midnight UTC to midnight UTC of the one day simulation. For the green house gas (CO2) data, we illustrate the concentrations with (pinkish) layered isosurfaces. Wind flow at two layers of the atmosphere are represented by directional arrows with a length proportional to the magnitude. In addition, turbulent kinetic energy is illustrated as brown-ish layered isosurfaces. Representation choices were made working directly with the NIST scientists who ran the simulation, thus leading us to a visualization that best highlights the salient features of the data. Animating data over time demonstrates particular patterns of the simulation such as how there is a boundary to how high the CO2 plume reaches, and where turbulence in the air comes from the mountainous regions as well as through heating of the terrain over the course of the day. The time-series animation also reveals how wind from the Atlantic Ocean brings clean air into the Chesapeake region in the afternoon.

In our present workflow, we manipulate various ParaView features in response to input from the NIST scientist, who knows what aspects of the data are important to see. With ParaView, we rapidly change the visualization parameters which are immediately reflected in the immersive environment. Immediate feedback while the scientist is

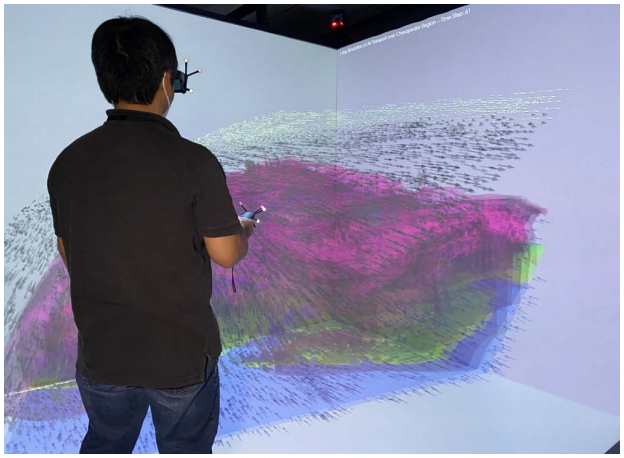

Figure 6: A visualization of meteorology data in the NIST CAVE Immersive Environment using ParaView with the *CAVE-Interaction* plugin. Photo: Steven Satterfield, NIST.

in the CAVE with the data significantly reduces the time needed to develop useful visual analytics.

One issue that arose during the exploration process with the meteorology data were problems with the ParaView volume rendering algorithm when rendered via the `pvserver` which provides the visual output for the CAVE walls. This is the result of the GPU-based volume rendering algorithm not properly adjusting to the continually changing viewpoint based on the user's head movements. (interestingly, the CPU-based rendering is correct, but is too slow for immersion.) Correcting this issue is now part of our collaboration with Kitware. However, the use of layered isosurfaces provides a reasonable alternative visualization.

In another case study, NIST scientists are investigating the feasibility of a simple wearable wireless device that can be used at home to detect or monitor excess fluid in the lungs [12]. Using a 3D computational human body model, propagation of the radio waves at MedRadio frequency band [12] through the human lungs with varying levels of fluid are being studied. Using Poynting vector data obtained through an electromagnetic solver, the path of RF energy between a transmitter on the chest and a receiver on the back side of the torso can be obtained.

The NIST scientists working on this project use ParaView on their desktop computers, and thus can quickly transition to the NIST CAVE through the CAVE-Interaction plugin. As shown in Fig. 7 the (rainbow colored) streamlines show the paths of RF energy between the transmitter and the receiver. With an appropriately designed wearable antenna, the main portion of the RF energy flows through the lungs; however, there is a small leakage that goes around the body. The path that goes through the lung helps to infer information about the fluid build-up in the lungs. For this immersive exploration of the RF data, NIST scientists working on the project used the desktop version of ParaView to setup the visualization pipeline and saved the information into a ParaView state file (`.pvsm`). The same ParaView state file re-creates the visualization in the NIST CAVE through the *CAVE-Interaction* plugin.

While the CAVE facility is conducive to working in collaboration with others who can see and comment on the visualization in the shared physical space, the rise of teleworking drives the need for an at-home immersive tool. For this circumstance, we are also pursuing interfaces to other XR systems. To keep within the ParaView ecosystem, and thus maintain efficiency in the visualization workflow, we are also working to advance the HMD-based *XR-Interface* plugin for ParaView. The ParaView *XR-Interface* allows scientists

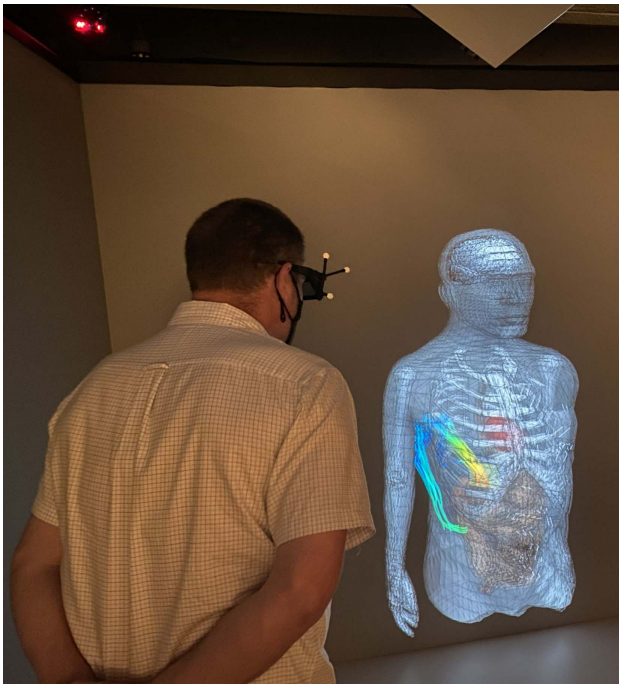

Figure 7: Visualization of simulation of an RF signal interacting with a human torso using the ParaView *CAVE-Interaction* plugin in the NIST Immersive Environment. Rainbow-colored streamlines show the simulated pathways taken by the signal from chest to back. Computer body is double as this is in stereo. Photo: Simon Su, NIST.

to immersively explore their data in smaller footprint VR systems —including in their homes.

The ParaView *XR-Interface* plugin together with the same ParaView state file used on the desktop and in the CAVE, scientists are able to shift between desktop, CAVE and at-home HMD to explore the same visualization, making use of the unique advantages provided by each setup.

These two case examples demonstrate how ParaView, with multiple visualization interfaces empowers NIST scientists to visualize their data in whatever means is best suited for their present circumstance, be it at-home, in their NIST office, or at the NIST CAVE facility. Overall, by collaborating with many domain users, we are able to push the envelop of both the data exploration to gain domain knowledge and also the capability of our immersive visualization tool.

5 Discussion and Future Work

While significant progress has been made to enhance ParaView's visualization capabilities through multiple immersive interfaces, there are additional improvements underway. For example, as mentioned in the case study section, we found ParaView volume rendering functionality to be deficient within CAVE or tiled-screen rendering, which becomes apparent when head-tracking with the *CAVE-Interaction* plugin. Improving the performance of the CPU-based volume rendering is one possible solution, but the frame rate required for immersive visualization incentivizes correcting the issue with the GPU based volume rendering.

While many people are returning to office work, this is often for fewer days a week, and for some telework has become their standard working situation. Navigating the challenges of hybrid work requires a new infrastructure to support open and regular communications among geographically dispersed team members. Among

the new challenges in conducting research collaboration will be the reduced frequency when team members are "just down the hall", and available for impromptu reviews of data using modern scientific visualization tools. Thus, we are now pursuing enhancements to the ParaView collaboration feature such that the ParaView desktop, *CAVE-Interaction* plugin and *XR-Interface* plugin all interface to the same collaboration server allowing researchers working from any interface mode to interact together.

To develop this new collaborative platform, we will refactor the existing visualization functionality of ParaView's XR features that already support head-mounted displays to work with the desktop and CAVE interfaces. The end result will be to enable both immersive and non-immersive users of ParaView (located in different places) to collaboratively visualize, discuss, and manipulate the same scientific dataset as though working in the same room.

These efforts will involve evaluating the new collaborative platform. Tests will include the measured and surveyed components of the utility of the collaboration features during day-to-day working sessions with the NIST RF signal visualization project discussed in the case study section. We will explore best-practice human computer interaction concepts to provide a ubiquitous user interface for data interaction. Creating an interface that works well on both immersive and non-immersive platforms will be particularly challenging as we will need to develop a unified interaction methodology capable of mapping different user interaction styles from heterogeneous 2D and 3D input devices into similar manipulation actions of the scientific data. Given that our platform supports shared visualization of the users connecting from ParaView running on the desktop, Head Mounted Display and CAVE visualization systems, user interactions will require similarly-featured interactions from users of a 2D mouse (desktop) and 3D wand (Head Mounted Display and CAVE).

5.1 Conclusion

Our two case studies demonstrate the successful use of our immersive visualization enhancements in ParaView supported by our immersive analytics scientific workflow at NIST. Leveraging the three decades of VTK/ParaView development in scientific visualization by adding immersive visualization enhancements, we could quickly develop meaningful and insightful immersive analytics applications while empowering other NIST scientist to develop immersive analytics visualization on their own either using the CAVE immersive visualization system at NIST or using a head mounted display system in their teleworking site. Although some visualization algorithms may not (yet) work as intended in the immersive mode, we will continue to work on improving the visualization capability in immersive ParaView.

With the expansion of the immersive collaboration features of ParaView to the desktop and the CAVE, we anticipate further adoption of the technology, especially for research teams with members located at different sites.

Acknowledgments

Certain commercial equipment, instruments, or materials (or suppliers, or software, etc) are identified in this paper to foster understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

References

[1] Monado - Open Source XR Platform. https://monado.dev/. Accessed: 2022-07-31.

[2] Navigation Basics in Virtual Reality with ParaView. https://www.kitware.com/navigation-basics-in-virtual-reality-with-paraview. Accessed: 2022-07-31.

[3] New in VTK: Rendering in external immersive environments. https://www.kitware.com/new-in-vtk-rendering-in-external-immersive-environments/. Accessed: 2022-07-31.

[4] Taking ParaView into Virtual Reality. https://www.kitware.com/taking-paraview-into-virtual-reality. Accessed: 2022-07-31.

[5] D. Arora, G. Babakhanova, and C. G. Simon. Tissue engineering measurands. *ACS Biomaterials Science & Engineering*, 6(10):5368–5376, 2020. PMID: 33320558. doi: 10.1021/acsbiomaterials.0c00475

[6] U. Ayachit and L. Avila. *The ParaView Guide: Updated for ParaView Version 4.3*. Kitware, 2015.

[7] A. Bierbaum, C. Just, P. Hartling, K. Meinert, A. Baker, and C. Cruz-Neira. VR Juggler: a virtual platform for virtual reality application development. In *Proceedings IEEE Virtual Reality 2001*, pp. 89–96, 2001. doi: 10.1109/VR.2001.913774

[8] K. Blom. vjVTK: a toolkit for interactive visualization in Virtual Reality. In *Eurographics Symposium on Virtual Environments (2006)*, pp. 17–19, 01 2006.

[9] I. L. Coto, X. Ren, A. Karion, K. McKain, C. Sweeney, R. Dickerson, B. McDonald, D. Ahn, R. Salawitch, H. He, P. Shepson, and J. Whetstone. Carbon monoxide emissions from the washington, dc, and baltimore metropolitan area: Recent trend and covid-19 anomaly. 2022-01-26 05:01:00 2022. doi: 10.1021/acs.est.1c06288

[10] J. Kelso, L. Arsenault, S. Satterfield, and R. Kriz. DIVERSE: a framework for building extensible and reconfigurable device independent virtual environments. In *Proceedings IEEE Virtual Reality 2002*, pp. 183–190, 2002. doi: 10.1109/VR.2002.996521

[11] O. Kreylos. Environment-Independent VR Development. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, P. Remagnino, F. Porikli, J. Peters, J. Klosowski, L. Arns, Y. K. Chun, T.-M. Rhyne, and L. Monroe, eds., *Advances in Visual Computing*, pp. 901–912. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[12] K. Krhac, K. Sayrafian, U. Bengi, and S. Dumanli. A Wearable Wireless Monitoring System for the Detection of Pulmonary Edema. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–5, 2021. doi: 10.1109/GLOBECOM46510.2021.9685118

[13] P. J. Rajlich. An Object Oriented Approach to Developing Visualization Tools Portable Across Desktop and Virtual Environments. Master's thesis, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 1998.

[14] J. Rohlf and J. Helman. *IRIS Performer: A High Performance Multiprocessing Toolkit for Real-Time 3D Graphics*, p. 381–394. Association for Computing Machinery, New York, NY, USA, 1994.

[15] W. Schroeder, K. Martin, K. Martin, and B. Lorensen. *The Visualization Toolkit: An Object-oriented Approach to 3-D Graphics*. Prentice Hall PTR, 1996.

[16] W. R. Sherman, D. Coming, and S. Su. FreeVR: honoring the past, looking to the future. In M. Dolinsky and I. E. McDowall, eds., *The Engineering Reality of Virtual Reality 2013*, vol. 8649, pp. 47 – 61. International Society for Optics and Photonics, SPIE, 2013. doi: 10.1117/12.2008578

[17] W. R. Sherman, M. A. Penick, S. Su, T. J. Brown, and F. C. Harris. VRFire: an Immersive Visualization Experience for Wildfire Spread Analysis. In *2007 IEEE Virtual Reality Conference*, pp. 243–246, 2007. doi: 10.1109/VR.2007.352491

[18] N. Shetty, A. Chaudhary, D. Coming, W. R. Sherman, P. O'Leary, E. T. Whiting, and S. Su. Immersive ParaView: A community-based, immersive, universal scientific visualization application. In *2011 IEEE Virtual Reality Conference*, pp. 239–240, 2011. doi: 10.1109/VR.2011.5759487

[19] S. Su, C. Cruz-Neira, E. Habib, and A. Gerndt. Virtual hydrology observatory: an immersive visualization of hydrology modeling. In I. E. McDowall and M. Dolinsky, eds., *The Engineering Reality of Virtual Reality 2009*, vol. 7238, pp. 114 – 121. International Society for Optics and Photonics, SPIE, 2009. doi: 10.1117/12.807177

[20] S. Su, V. Perry, L. Bravo, S. Kase, H. Roy, K. Cox, and V. R. Dasari. Virtual and Augmented Reality Applications to Support Data Analysis and Assessment of Science and Engineering. *Computing in Science and Engineering*, 22(3):27–39, 2020. doi: 10.1109/MCSE.2020.2971188

[21] R. M. Taylor, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. VRPN: A Device-Independent, Network-Transparent VR Peripheral System. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '01, p. 55–61. Association for Computing Machinery, New York, NY, USA, 2001. doi: 10.1145/505008.505019

[22] J. E. Terrill, W. L. George, T. J. Griffin, J. G. Hagedorn, J. T. Kelso, M. Olano, A. P. Peskin, S. G. Satterfield, J. S. Sims, J. W. Bullard, J. P. Dunkers, N. S. Martys, A. O'Gallagher, and G. Haemer. Extending Measurement Science to Interactive Visualisation Environments. In R. A. Liere and T. E. Zudilova-Seinstra, eds., *Trends in Interactive Visualization*, pp. 287–302. Springer, London, 2009. doi: 10.1007/978-1-84800-269-2_13