# Rendering Volumes in WebXR

Maxim Kuznetsov, University of California Santa Cruz

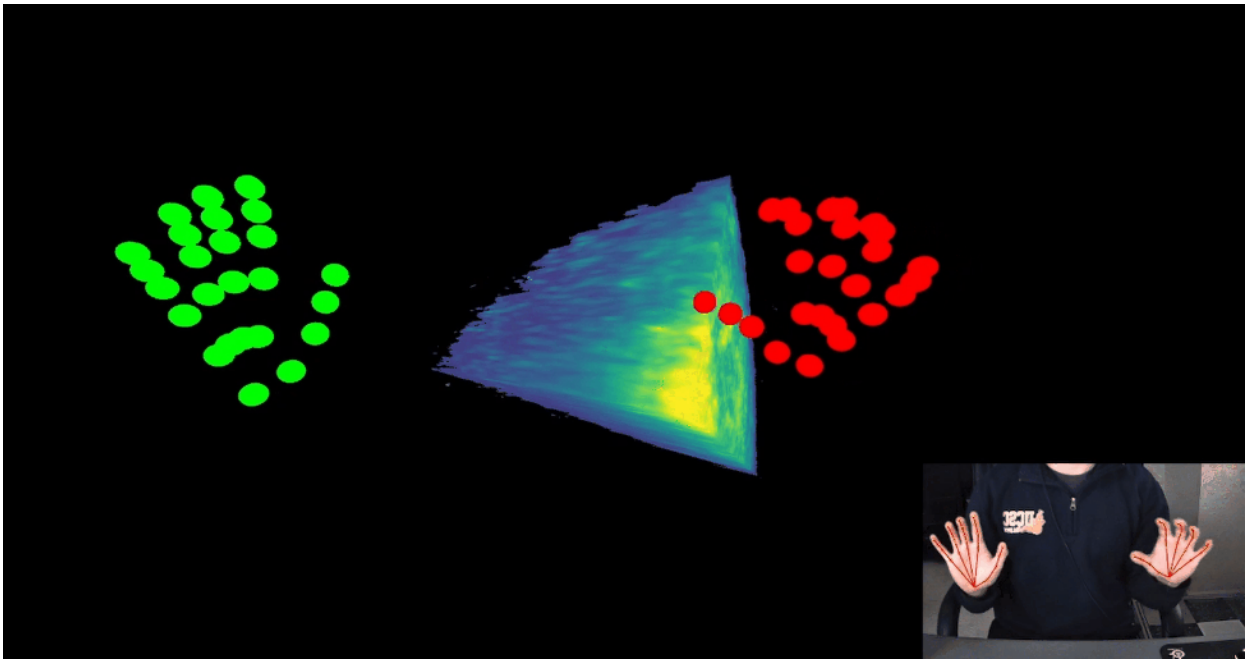

Fig. 1. Volume analysis with free hand motion tracking.

**Abstract**—Visualization and analysis of volumetric data is essential in biological research. The current tooling used in these visualizations need great processing power to be used efficiently in real time. The high requirements of these tools cause underfunded researchers to be unable to perform the same level of analysis as researchers with the proper resources available. On top of this, almost none of the tooling available takes advantage of immersive XR, leaving the user to analyze the data on a traditional 2D surface. This system is an attempt to pair the two. On a simple static page, users are able to analyze data both in traditional surfaces and in immersive VR using WebXR, and the performance is constant across devices by varying the level of detail.

**Index Terms**—Volume rendering, WebXR, multiplatform

## 1 Introduction

Visualization of biological data plays an important role in the innovation of new research within medicine. In areas such as protein analysis and epidemiology [1, 2, 4, 5], proper visualizations are key in keeping the rate of research high. The visualization techniques often require large amounts of processing power and do not take advantage of new developments in human-computer interaction. When interacting with 3D data on a traditional 2D surface, it can be difficult to annotate and analyze the projected data in a natural way. We propose a system for visualizing volumetric data that can presents effective results on both low and high powered systems, with immersive WebXR capabilities.

...

## 2 Background

...

## 3 Results

WebGL was chosen as the base platform to build this system. As the performance levels of WebGL are reaching closer to native OpenGL [3], it is becoming an increasingly attractive platform for open-source tools and visualizations. Visualization techniques that require high real-time performance are starting to become possible to accomplish through WebGL. Three.js was chosen as the higher level API between the the developer and WebGL. Three.js is capable of managing much of the boilerplate required to setup a usable WebGL program while also being very extensible. ...

### 3.1 Volumetric Rendering and WebGL

This system is currently capable of loading and displaying Nearly Raw Raster Data (NRRD) files. These files contain stacks of two-dimensional images, assembled together into a three-dimensional format. The NRRD loader included in Three.js is used to load the file into memory as a 3D texture in a format that the GPU is capable of reading.

For rendering the volumes, the Three.js volume shader was chosen as the base shader to build the platform off of. The shader in Three.js is a single-pass shader created for volumetric rendering, with a ray being cast and pixel intensity being determined directly in the fragment shader. In each step of casting the ray in the fragment shader, the 3D texture that was created from the NRRD file is sampled. Since the entire 3D texture is loaded into memory, efficient random access of the 3D texture data is possible. ...

### 3.2 WebXR

All components of VR integration are done through the WebXR API. The WebXR API allows us to interface with many different VR devices in a way that is platform agnostic by supplying VR features that are generalizable to all VR devices. This allows our system to work with any device that is capable of interfacing with WebXR. No platform-dependant code is written in this system.

WebXR integration required modifications to the Three.js volume shader to work properly. The original shader was built with an orthographic camera projection in mind. WebXR requires a perspective camera projection, and without modifications the perspective projection causes major warping in the original shader. The fragment shader was modified to support a perspective camera by changing the view direction to be determined by the direction of the vector from the camera position to the fragment position. This small change removes the warping that occurred in the original shader and displays the volume accurately with a perspective projection.

Locomotion is handled by translating and rotating the user on a rig that circles the volume loaded. Movement in VR can be accomplished using hand tracking or with a controller's joystick. Moving the rig around the volume itself simulates rotating and scaling the object for the user, even allowing traversal within the volume itself. ...

#### 3.2.1 VR Locomotion

...

#### 3.2.2 Hand Tracking

Hand control of the volume was implemented using the new WebXR Hand Input API, which is being interfaced by Three.js. WebXR support for hand input is still in its infancy, with few devices offering support. Currently, the WebXR Hand Input API can be used with the Oculus browser[1] or using the Hand branch of the Mozilla WebXR emulator[2], which is capable of implementing hand tracking using only a webcam. Because the WebXR Hand Input API is not platform-specific, this system will be able to support any future devices that implement WebXR-capable hand tracking.

When hand input is detected to be present, a model of the hands is attached to the same rig as the VR camera. The hand tracking allows the user to rotate and translate towards or away from the volume without requiring any extra hardware. When the user is detected to be touching their index fingertip and thumb fingertip on their main hand, which is toggled by listening to the "pinching" flag set by Three.js, the camera unlocks its position. In this state, any change in the X or Y position of the main hand results in translation of the camera around the volume which mimics rotation of the volume itself. The user may also use a secondary hand to translate the camera closer or further away from the volume, which is tracked using the change in Y position of the secondary hand. Once the user wants to stop movement, they can simply stop pinching, locking the camera's position. This allows for full three dimensional analysis of the loaded volume with only the user's hands, no longer needing a controller for movement. ...

### 3.3 Annotation

...

## 4 Discussion

...

## 5 Conclusion

...

## References


[1] L. N. Carroll, A. P. Au, L. T. Detwiler, T.-c. Fu, I. S. Painter, and N. F. Abernethy. Visualization and analytics tools for infectious disease epidemiology: A systematic review. *Journal of Biomedical Informatics*, 51:287–298, Oct. 2014. doi: 10.1016/j.jbi.2014.04.006

[2] N. T. Doncheva, K. Klein, F. S. Domingues, and M. Albrecht. Analyzing and visualizing residue networks of protein structures. *Trends in Biochemical Sciences*, 36(4):179–182, Apr. 2011. doi: 10.1016/j.tibs.2011.01.002

[3] H. Kim, S. Nam, J. Park, and D. Ko. Direct canvas: Optimized webgl rendering model. In *2018 IEEE International Conference on Consumer Electronics (ICCE)*, pp. 1–3. IEEE, 2018.

[4] A. Porollo and J. Meller. Versatile annotation and publication quality visualization of protein complexes using POLYVIEW-3D. *BMC Bioinformatics*, 8(1):316, Dec. 2007. doi: 10.1186/1471-2105-8-316

[5] Z. Yang, K. Lasker, D. Schneidman-Duhovny, B. Webb, C. C. Huang, E. F. Pettersen, T. D. Goddard, E. C. Meng, A. Sali, and T. E. Ferrin. UCSF Chimera, MODELLER, and IMP: An integrated modeling system. *Journal of Structural Biology*, 179(3):269–278, Sept. 2012. doi: 10.1016/j.jsb.2011.09.006


[1] https://developer.oculus.com/webxr/

[2] https://github.com/MozillaReality/WebXR-emulator-extension/tree/Hand