StereoCell - A Virtual Reality Exploration of 3D Immunoflourescence Data

Chahat Kalsi
ck4106
chahat.kalsi@nyu.edu

Abstract

Cyclic immunofluorescence (CyCIF) is a highly multiplexed imaging technique that can capture 40+ biomarkers on the same tissue sample. Currently, most analyses of these large-scale 3D CyCIF volumes rely on conventional desktop displays. However, prior work in immersive visualization shows that virtual and augmented reality (VR/AR) environments can significantly enhance depth perception and comprehension of three-dimensional spatial relationships. This motivates an investigation of 3D CyCIF data within a VR analytics environment. Key challenges include the high computational complexity of rendering volumetric data of this magnitude on peripheral VR hardware, as well as designing an interactive interface tailored to domain-relevant workflows (such as channel selection, color adjustments, clipping, and spatial navigation). This work aims to address these challenges by designing an application that brings 3D CyCIF data into immersive environments and can serve as a starting point to investigate whether virtual reality can aid histopathological disease diagnosis.

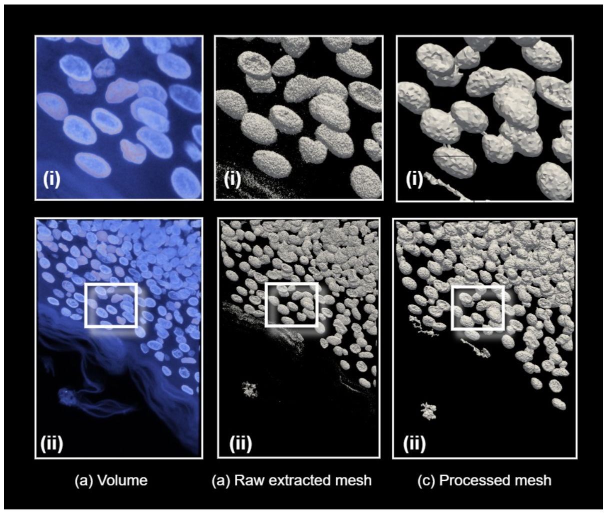

Figure 1. **Automated mesh extraction and processing pipeline for multi-channel volumetric data.** (a) Subsection of the raw volume. (b) Automatically detected and extracted isosurfaces using FlyingEdges3D. (c) Processed and simplified meshes after filtering, hole filling, and alpha wrapping; this pipeline is applied to all channels and the resulting meshes are used for efficient surface rendering in VR.

1. Introduction

Recent advances in three-dimensional cyclic immunofluorescence (CyCIF) [9] have enabled the imaging of tissue specimens with thicknesses of $\sim$30–50 µm. [17] This represents an approximately six-to-ten-fold increase compared to conventional 5 µm thin sections. In contrast to these thin tissue slices, where few, if any, cells remain intact, this thicker-section 3D CyCIF approach permits a more accurate assessment of intact cell-cell interactions. [17] Profiling these interactions has significant applications across biomedical domains, particularly in cancer research, enabling characterization of cellular morphology and immune niches, which in turn can inform immunotherapy strategies. [9, 11, 17]

Despite the advantages that this novel 3D imaging technique has over 2D imaging, tools to visualize and investigate the datasets generated using it are conventionally limited to desktop based displays. [1, 4, 10, 14] These displays project the three-dimensional data onto flat two-dimensional screens, and thus display information using only monocular depth cues. In contrast, more immersive technologies such as head-mounted displays (HMDs) might be more beneficial in terms of preserving the naturalistic perception of depth in the 3D images using stereoscopic rendering. [5] Other ways that HMDs incorporate close to natural depth perception that typical desktop displays can not provide are by reproducing stereopsis and motion parallax. [2, 8, 13]

However, despite immersive visualization of these 3D CyCIF datasets having advantages over flat screens, peripheral HMDs also suffer from a comparative lack of computational resources. Not only do these multiplexed datasets contain a large number of channels, but current technology also allows imaging them at incredibly high resolutions. For example, the 3D melanoma tissue dataset collected and analyzed by Yapp *et al.* [17] comprises 70 channels

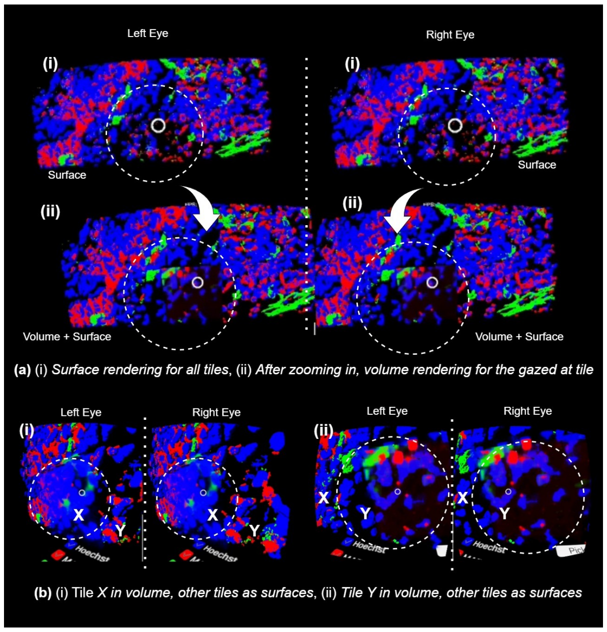
Left Eye
Right Eye
(i)
(i)
Surface
Surface
(ii)
(ii)
Volume + Surface
Volume + Surface
(a) (i) Surface rendering for all tiles, (ii) After zooming in, volume rendering for the gazed at tile
Left Eye
Right Eye
Left Eye
Right Eye
X
Y
Hoechst
(b) (i) Tile X in volume, other tiles as surfaces, (ii) Tile Y in volume, other tiles as surfaces

Figure 2. **Hybrid tiled rendering strategy for interactive VR exploration.** (a) The system switches from surface rendering to volume rendering as the user moves closer to the data. (b) Foveated rendering: only the gazed-at tile is volume rendered while the remaining tiles are rendered as surfaces to reduce computation.

(volumes) across six levels of resolution, with the highest having 194×5,508×10,908 voxels per channel (1.48 TB) and the lowest having 194×172×340 voxels per channel (1.48 GB). Techniques to work around this utilize the fact that human visual perception is most concentrated in the central region of the retina (the fovea), and thus most computational resources can be allocated to rendering that region. As we move away from the foveal region, peripheral regions can be visualized in much less detail. This forms the basis of a fundamental technique in immersive visualization called foveated rendering. [3] Other approaches to optimize the use of computational resources provided by HMDs involve out-of-core rendering of frames [7] and streaming them onto the peripheral device [16], and more recently, using alternative representations such as compressions generated via Gaussian splatting. [6, 15]

In this work, we bring large volumetric microscopy datasets to virtual reality by investigating the use of these techniques to leverage stereoscopic depth cues while visualizing 3D data.

2. Results and Demonstration

The result of this project is a self-contained application that demonstrates a technique for extracting accurate, lightweight meshes from volumetric data and rendering both surfaces and volumes at interactive frame rates on edge devices with limited compute.

Figure 1 shows the output of the fully automated mesh extraction and processing pipeline, and the extracted meshes are used for efficient surface rendering when viewing the dataset from a distance in the virtual reality interface. In (a), we show a subsection of the raw volumetric data. In (b), we show the result of automated isosurface detection and extraction using the FlyingEdges3D algorithm. In (c), we show the result after mesh processing and simplification, including filtering, hole filling, and alpha wrapping applied to the raw meshes. This pipeline is applied to all channels in the multi-volume dataset.

Figure 2 illustrates our rendering strategy for interactive exploration. In (a), the system switches from surface rendering to volume rendering as the user moves closer to the data. In (b), only the user's gazed-at tile is rendered using volume rendering, while the remaining tiles are rendered as surfaces to reduce computation.

Figure 3 demonstrates interaction using gaze (screen center) and touch input (via the cardboard's base button). In (b), we show the user interface, which supports enabling/disabling channels in the multi-volume dataset, changing the color of any channel (updating both mesh materials and the volume transfer function), selecting a specific tile for closer inspection, zooming/translating/rotating the data using touch input, and toggling volume rendering on/off. Panels (c)–(f) show example interactions: (c) rotating the data using single-finger touch, (d) toggling off the blue "Hoechst" channel, (e) changing the previously red "MART1" channel to violet and the previously green "CD31" channel to yellow, and (f) selecting a tile to isolate it for drill-down inspection. Inset (i) shows the selected tile rendered as a volume, while inset (ii) shows the same tile rendered as a surface after disabling volume rendering.

Further methodological and implementation details are provided in Section 3.

3. Implementation

3.1. Hardware and software

The application targets mobile, low-cost VR. It is deployed and tested on a Moto G 5G (xt2417d) inserted into an Mr Cardboard viewer. The project is implemented in Unity 6 (6000.0.034f1) using the Google Cardboard XR Plugin for Unity (v1.31.0), which provides stereoscopic rendering, head-motion tracking, and a viewer-button interaction pathway for mobile VR experiences.

For registering user input, we use (i) **gaze** as the screen center ray from the main camera, and (ii) **touch** as a discrete selection/confirm action triggered by the Cardboard button (which, on touch-based viewers, registers as a standard screen touch event).

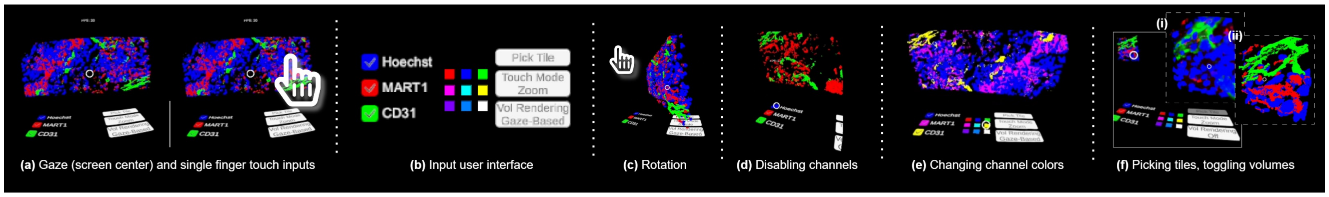

Figure 3. **Interaction techniques and user interface.** Interaction is performed using gaze (screen center) and touch input via the cardboard button. The UI supports channel enable/disable, per-channel color updates (mesh materials and volume transfer functions), tile selection for drill-down inspection, touch-based zoom/translate/rotate, and toggling volume rendering. Example interactions include rotating the dataset, toggling the "Hoechst" channel, recoloring "MART1" and "CD31," and isolating a tile; inset (i) shows the selected tile rendered as a volume and inset (ii) shows the same tile rendered as a surface.

3.2. Dataset

The provided 3D melanoma tissue dataset [17] contains 70 volumetric channels across 6 resolution levels. At the highest resolution, each channel has $10908 \times 5508 \times 194$ voxels (approximately 1.48 TB total), while at the lowest resolution each channel has $340 \times 172 \times 194$ voxels (approximately 1.48 GB total).

Because the full dataset is not feasible for interactive mobile rendering, we visualize only a subset. Concretely, we work with **10 spatial tiles** arranged as a 5×2 grid, where each tile is a cropped subvolume of size $512 \times 512 \times 194$ voxels. At runtime, we display only **3 channels** (rather than all 70). For each tile, we store both (i) per-channel surface meshes, and (ii) the corresponding raw subvolume needed for volume rendering.

3.3. Extracting and processing meshes

All mesh extraction is performed offline using a prebuilt C++ preprocessing script. The goal is to produce clean, lightweight meshes suitable for fast rasterization on a phone GPU.

Automatic iso-value detection. For each channel and tile, we automatically determine an iso-value τ using the method of Pekar *et al.* [12]. This step avoids manual threshold tuning and is especially important because a globally chosen threshold can be suboptimal for local tissue regions; in our pipeline, τ is inferred independently per tile (and per channel).

Isosurface extraction. Given τ, we extract an isosurface using VTK's `vtkFlyingEdges3D`, a highly scalable isocontouring algorithm designed for large structured volumes and optimized via trimming and reduced redundant computations. This produces a raw triangle mesh for each (tile, channel) pair.

Figure 1 shows the raw volume data (a) and the corresponding raw meshes extracted from it (b). The strong visual agreement between the two indicates that the automatically selected iso-value was well suited for this region.

Mesh processing. The raw meshes can contain small disconnected fragments, holes, and noise, and millions of triangles which makes dealing with them at interactive frame rates rather infeasible. We use CGAL for a sequence of cleanup operations: (i) remove small connected components using CGAL's connected-component filtering utilities (keeping only sufficiently large components by face-count and average component diagonal metrics), (ii) fill small holes using CGAL Polygon Mesh Processing hole filling routines, and (iii) apply *alpha wrapping* to obtain a closed, simplified surface controlled by the `alpha` and `offset` parameters, which trade off feature preservation against mesh complexity and tightness. `Alpha` and `offset` are again chosen using average face count and diagonal metrics. The output is a clean, low-polygon mesh per tile and channel that can be rendered efficiently on-device.

3.4. Volume rendering in Unity

For volume rendering, we use Matias Lävik's *UnityVolumeRendering* (Easy Volume Rendering) plugin. The plugin performs direct volume rendering in a fragment shader by raymarching through the volume stored as a 3D texture and mapping density to color/opacity using transfer functions.

Performance optimisation. To keep rendering interactive on a mobile GPU, we: (i) reduce the maximum raymarch sample count to a small budget (we use 100 steps), (ii) enable early ray termination (supported by the plugin) to stop accumulating once opacity saturates, and (iii) restrict expensive volume rendering to only one tile at a time (the gazed-at tile), leaving the remaining tiles in surface mode. Early ray termination and skipping transparent regions are standard, widely used accelerations in volume rendering, and they directly reduce fragment workload. Where supported by the XR pipeline and device, additional system-level optimizations such as foveated rendering can further reduce peripheral shading cost.

3.5. Switching between surface and volume rendering

At runtime, the scene contains a fixed 5×2 tiled layout. Each tile has: (i) three mesh renderers (one per displayed channel) for surface mode, and (ii) one volume-rendered object (a cube proxy with the corresponding 3D texture and transfer function) for volume mode. We implement a hybrid mesh and volume rendering strategy.

Distance-based switching. when the user is far from the data, tiles are rendered using meshes (fast rasterization). When the user moves within a preset distance threshold, the system allows volume rendering for detailed inspection.

Gaze-based switching. we cast a ray from the camera through the screen center to find the currently gazed tile. Only this tile is eligible for volume rendering; all other tiles remain in mesh mode. This acts as an application-level foveation mechanism that keeps fragment shader cost bounded.

3.6. Interaction: gaze dwell and touch

We support two complementary interaction modalities, ie, gaze dwell and touch. These are detailed in this section.

Gaze dwell. Each interactive object (tiles and UI targets) is enclosed by a Unity collider. If the gaze ray intersects a collider continuously for 1.5 seconds, we trigger a dwell event (e.g., select a tile for drill-down). This enables hands-free operation.

Touch input. We use the Cardboard viewer button as a discrete input for toggles and confirmations. For touch-based Cardboard viewers, the button physically presses the phone screen and is exposed to Unity as a standard touch/mouse-click style event, which we map to UI interactions (toggle channel visibility, enable/disable volume rendering, and manipulate transforms).

4. Discussion

The hybrid strategy (surface-from-distance, volume-up-close, and gaze-based single-tile volume rendering) enables interactive exploration on a mobile VR device, but it also exposes a key limitation: the use of fixed $194 \times 512 \times 512$ tiles can make the mesh-to-volume transition feel visually "blocky" and less organic, especially near tile boundaries. This is a common challenge in foveated and multiresolution rendering, where hard boundaries between high- and low-quality regions can be perceptually noticeable unless transitions are smoothed.

A natural next step is to replace the discrete tile-based switch with a more continuous, circular (or gaze-centered) transition region. For example, volume quality could fall off radially from the gaze point using a feathered blending mask (inner high-quality core with a smooth transition ring), reducing the visibility of boundaries while keeping the shading workload bounded.

I also plan to improve performance and scalability by adding more aggressive empty-space skipping in the ray-marcher, building on standard GPU volume rendering optimizations such as early ray termination and skipping transparent regions.

Finally, the current preprocessing and import workflow is external to Unity (C++ scripts for meshing and cleanup, manual asset import). Integrating this pipeline into Unity as an editor tool (e.g., automated mesh/subvolume generation, transfer-function presets, and one-click tile scene construction) would make the system easier to reproduce and extend.

References

[1] AGAVE 3D pathtrace image viewer. 1

[2] Effects of stereopsis on vection, presence and cybersickness in head-mounted display (HMD) virtual reality | Scientific Reports. 1

[3] Towards foveated rendering for gaze-tracked virtual reality | ACM Transactions on Graphics. 2

[4] Dominik Drees, Simon Leistikow, Xiaoyi Jiang, and Lars Linsen. Voreen – An Open-source Framework for Interactive Visualization and Processing of Large Volume Data, July 2022. arXiv:2207.12746 [cs]. 1

[5] Fatima El Jamiy and Ronald Marsh. Survey on depth perception in head mounted displays: distance estimation in virtual reality, augmented reality, and mixed reality. *IET Image Processing*, 13(5):707–712, 2019. _eprint: https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-ipr.2018.5920. 1

[6] Linus Franke, Laura Fink, and Marc Stamminger. VR-Splatting: Foveated Radiance Field Rendering via 3D Gaussian Splatting and Neural Points. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 8(1):1–21, May 2025. arXiv:2410.17932 [cs]. 2

[7] Enrico Gobbetti, Fabio Marton, and José Antonio Iglesias Guitián. A single-pass GPU ray casting framework for interactive out-of-core rendering of massive volumetric datasets. *The Visual Computer*, 24(7-9):797–806, July 2008. 2

[8] J. Adam Jones, J. Edward Swan, Gurjot Singh, Eric Kolstad, and Stephen R. Ellis. The effects of virtual reality, augmented reality, and motion parallax on egocentric depth perception. In *Proceedings of the 5th symposium on Applied perception in graphics and visualization*, APGV '08, pages 9–14, New York, NY, USA, Aug. 2008. Association for Computing Machinery. 1

[9] Jia-Ren Lin, Benjamin Izar, Shu Wang, Clarence Yapp, Shaolin Mei, Parin M Shah, Sandro Santagata, and Peter K

Sorger. Highly multiplexed immunofluorescence imaging of human tissues and tumors using t-CyCIF and conventional optical microscopes. *eLife*, 7:e31657, July 2018. Publisher: eLife Sciences Publications, Ltd. 1

[10] Trevor Manz, Ilan Gold, Nathan Heath Patterson, Chuck McCallum, Mark S. Keller, Bruce W. Herr, Katy Börner, Jeffrey M. Spraggins, and Nils Gehlenborg. Viv: multiscale visualization of high-resolution multiplexed bioimaging data on the web. *Nature methods*, 19(5):515–516, May 2022. 1

[11] Ajit J. Nirmal, Zoltan Maliga, Tuulia Vallius, Brian Quattrochi, Alyce A. Chen, Connor A. Jacobson, Roxanne J. Pelletier, Clarence Yapp, Raquel Arias-Camison, Yu-An Chen, Christine G. Lian, George F. Murphy, Sandro Santagata, and Peter K. Sorger. The Spatial Landscape of Progression and Immunoediting in Primary Melanoma at Single-Cell Resolution. *Cancer Discovery*, 12(6):1518–1541, June 2022. 1

[12] V. Pekar, R. Wiemker, and D. Hempel. Fast detection of meaningful isosurfaces for volume data visualization. In *Proceedings Visualization, 2001. VIS '01.*, pages 223–230, Oct. 2001. 3

[13] Ana Serrano, Incheol Kim, Zhili Chen, Stephen DiVerdi, Diego Gutierrez, Aaron Hertzmann, and Belen Masia. Motion parallax for 360° RGBD video. *IEEE Transactions on Visualization and Computer Graphics*, 25(5):1817–1827, May 2019. 1

[14] Nicholas Sofroniew, Talley Lambert, Grzegorz Bokota, Juan Nunez-Iglesias, Peter Sobolewski, Andrew Sweet, Lorenzo Gaifas, Kira Evans, Alister Burt, Draga Doncila Pop, Kevin Yamauchi, Melissa Weber Mendonça, Jaime Rodríguez-Guerra, Lucy Liu, Genevieve Buckley, Wouter-Michiel Vierdag, Timothy Monko, Carol Willing, Loic Royer, Ahmet Can Solak, Kyle I. S. Harrington, Jacopo Abramo, Jannis Ahlers, Daniel Althviz Moré, Oren Amsalem, Ashley Anderson, Andrew Annex, Constantin Aronssohn, Filippo Balzaretti, Peter Boone, Jordão Bragantini, Matthias Bussonnier, Clément Caporal, Ian Coccimiglio, Jan Eglinger, Andreas Eisenbarth, Jeremy Freeman, Christoph Gohlke, Kabilar Gunalan, Yaroslav Olegovich Halchenko, Hagai Har-Gil, Mark Harfouche, Volker Hilsenstein, Katherine Hutchings, Robert Kozar, Jessy Lauer, Gregor Lichtner, Hanjin Liu, Ziyang Liu, Alan Lowe, Luca Marconato, Sean Martin, Abigail McGovern, Lukasz Migas, Nadalyn Miller, Sofía Miñano, Hector Muñoz, Jan-Hendrik Müller, Christopher Nauroth-Kreß, Horst A. Obenhaus, David Palecek, Constantin Pape, Eric Perlman, Kim Pevey, Gonzalo Peña-Castellanos, Andrea Pierré, David Pinto, David Ross, Craig T. Russell, James Ryan, Gabriel Selzer, MB Smith, Paul Smith, Konstantin Sofiiuk, Johannes Soltwedel, David Stansby, Jules Vanaret, Pam Wadhwa, Martin Weigert, Jonas Windhager, Philip Winston, and Rubin Zhao. napari: a multi-dimensional image viewer for Python, Oct. 2025. 1

[15] Xuechang Tu, Lukas Radl, Michael Steiner, Markus Steinberger, Bernhard Kerbl, and Fernando de la Torre. VRSplat: Fast and Robust Gaussian Splatting for Virtual Reality, May 2025. arXiv:2505.10144 [cs]. 2

[16] Yujie Wang, Praneeth Chakravarthula, Qi Sun, and Baoquan Chen. Joint neural phase retrieval and compression for energy- and computation-efficient holography on the edge. *ACM Transactions on Graphics*, 41(4):1–16, July 2022. 2

[17] Clarence Yapp, Ajit J. Nirmal, Felix Zhou, Alex Y. H. Wong, Juliann B. Tefft, Yi Daniel Lu, Zhiguo Shang, Zoltan Maliga, Paula Montero Llopis, George F. Murphy, Christine G. Lian, Gaudenz Danuser, Sandro Santagata, and Peter K. Sorger. Highly multiplexed 3D profiling of cell states and immune niches in human tumors. *Nature Methods*, 22(10):2180–2193, Oct. 2025. Publisher: Nature Publishing Group. 1, 3