A Virtual Reality Visualization Tool for Neuron Tracing

Will Usher†, Pavol Klacansky†, Frederick Federer, Peer-Timo Bremer, Aaron Knoll, Jeff Yarch, Alessandra Angelucci, and Valerio Pascucci

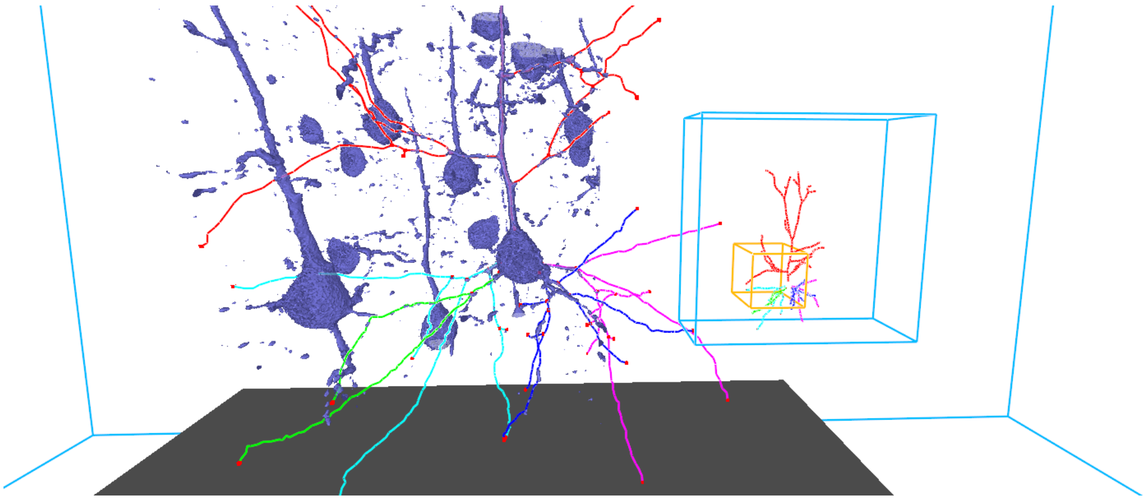


Fig. 1: A screenshot of our VR neuron tracing tool using the isosurface rendering mode. The dark gray floor represents the extent of the tracked space. Users can orient themselves in the dataset via the minimap (right), which shows the world extent in blue, the current focus region in orange, and the previously traced neuronal structures. The focus region is displayed in the center of the space. The 3D interaction and visualization provides an intuitive environment for exploring the data and a natural interface for neuron tracing, resulting in faster, high-quality traces with less fatigue reported by users compared to existing 2D tools.

Abstract—Tracing neurons in large-scale microscopy data is crucial to establishing a wiring diagram of the brain, which is needed to understand how neural circuits in the brain process information and generate behavior. Automatic techniques often fail for large and complex datasets, and connectomics researchers may spend weeks or months manually tracing neurons using 2D image stacks. We present a design study of a new virtual reality (VR) system, developed in collaboration with trained neuroanatomists, to trace neurons in microscope scans of the visual cortex of primates. We hypothesize that using consumer-grade VR technology to interact with neurons directly in 3D will help neuroscientists better resolve complex cases and enable them to trace neurons faster and with less physical and mental strain. We discuss both the design process and technical challenges in developing an interactive system to navigate and manipulate terabyte-sized image volumes in VR. Using a number of different datasets, we demonstrate that, compared to widely used commercial software, consumer-grade VR presents a promising alternative for scientists.

Index Terms—Virtual reality, interaction design, design studies

1 Introduction

Brain function emerges from the coordinated activity of billions of interconnected neurons that form dense neural circuits. A central goal of neuroscience is to understand how these circuits' computations relate to behavior. The field of connectomics is founded on the principle that understanding the precise wiring of these circuits, i.e., the location of neurons and the connections between them, is crucial to comprehending brain function at a mechanistic level. More insight into the fundamental connectivity within the brain also has the potential to lead to breakthroughs in the understanding of brain diseases and open new avenues for treatment.

However, obtaining a comprehensive wiring diagram of even relatively small and simple mammalian brains, such as that of a mouse, is a massive undertaking. Similar projects in species with larger brains that are evolutionarily closer to humans, such as non-human primates (NHPs), take more time and are more complex. To date, the only species whose nervous system has been completely mapped is the nematode Caenorhabditis elegans [45], which is comprised of only 302 neurons. Currently, the majority of connectome efforts are focused on mapping the mouse brain [8, 10]. However, with recent advances in high-resolution tissue labeling [27], optical tissue clearing [11, 46], and deep tissue imaging [13], mapping the NHP brain at mesoscopic scale is becoming feasible. One major impediment to mapping the NHP brain is the time-consuming, laborious effort of manually tracing labeled neuronal connections through the brain.

For most of the 20th century, reconstructing, annotating, and analyzing neurons has been done by creating hand-drawn images of labeled neurons, traced using an instrument known as *camera lucida* directly from thin brain sections viewed through a microscope. The first computer-aided system for tracing neurons [16] synced the movement of the microscope stage with a plotting board. The user would adjust

• *Usher, Klacansky, Knoll, and Pascucci are with the SCI Institute at the University of Utah, USA. E-mail: {will, klacansky, knolla, pascucci}@sci.utah.edu*
• *Federer, Yarch, and Angelucci are with the Moran Eye Institute at the University of Utah, USA. E-mail: freddieneuron@gmail.com, j.yarch@utah.edu, alessandra.angelucci@hsc.utah.edu*
• *Bremer is with Lawrence Livermore National Laboratory, USA. E-mail: bremer5@llnl.gov*

† *Usher and Klacansky are both first authors*

Manuscript received 31 Mar. 2017; accepted 1 Aug. 2017.
Date of publication 28 Aug. 2017; date of current version 1 Oct. 2017.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2017.2744079

the stage to select points along the neuron to be plotted, and press a foot pedal to record each point on the board and then measure distances between them. This system ultimately evolved into NeuroLucida [29], the current industry standard for neuron tracing. NeuroLucida allows scientists to draw lines along neuronal axons and dendrites using either tissue sections mounted on a glass slide or image stacks of scanned tissue. The software moves the microscope (or image) to keep the viewpoint and tracing aligned while the user navigates the data. Tracing labeled neurons manually is tedious and time-consuming, and may require months to reconstruct even small portions of the brain [5]. Part of the difficulty in this process is tracing 3D structures such as neurons through a 2D interface, i.e., a computer screen. Neurons often touch or run in parallel, and finding a viewpoint to properly distinguish them may require several non-obvious rotations of the volume. When working with image slices, this process is made more challenging by the fixed viewpoint. Automatic techniques for neuron reconstruction fail on complex and noisy data, and often the results must be corrected manually. In fact, Peng et al. [35] report that the clean-up process may take longer than manual tracing. In practice, most neuron reconstruction is still done manually [31].

Beyond the mechanics of tracing, another challenge is that microscopy technology is rapidly outpacing the supporting tools in terms of raw data size. State-of-the-art microscopes regularly produce terabytes worth of images, yet few existing tools are capable of handling data at this scale. Notably, the TeraFly [9] plugin for Vaa3D [36] supports paging in hierarchical volume data to explore large datasets. Other tools are often limited by the RAM capacity of the system. In this paper, we present a design study on how off-the-shelf virtual reality (VR) systems coupled with state-of-the-art data management and visualization solutions can improve the workflow of connectomics researchers. Working with trained neuroanatomists, we explore different rendering, interaction, and navigation methods, as well as the use of force feedback to improve the quality and speed of neuron tracing. We demonstrate that given a high enough frame rate and appropriate rendering techniques, a 3D interface substantially improves the overall user experience by allowing neuroscientists to directly interact with their data. Our contributions in detail are:

- A design study on using consumer-grade VR technology for neuron tracing;
- A flexible and scalable backend framework that allows neuron tracing in datasets that are orders of magnitude larger than currently feasible with existing approaches; and
- A comparison of the reconstruction accuracy and speed of our tool compared to the industry standard.

2 Background and Related Work

To provide context for the task of neuron tracing, we first discuss recent work in neuron tracing and the current state of the art in the connectomics tracing workflow. We then discuss related work in immersive environments and direct 3D interaction.

2.1 Neuron Tracing

Automated methods for neuron reconstruction continue to improve; however, neuroscientists often find the results of these algorithms unsatisfactory [26]. Thus, tracing neurons remains primarily a manual task. Meijering [31] noted that data quality was the primary reason these algorithms fail in practice, as the current state-of-the-art methods provide error-free results only in highly optimal conditions. For a full review and comparison of recent methods, we refer readers to a recent paper by Acciai et al. [4].

Tools such as Vaa3D [36] and NeuroLucida 360 [30] provide methods for semi-automatic reconstruction, in which the user guides the system along a neuron and the system extracts the 3D structure. The *Virtual Finger* [37], available in Vaa3D, casts rays into the volume to determine the potentially selected objects, e.g., neural structures, as the user draws a line with the mouse. To create a 3D curve from the line, the method searches locally in the data to connect the selected objects, resolving cases in which a ray intersects multiple features. The *Virtual Finger* is inherently view and visibility dependent and may require moving the viewpoint to make the desired selection or correct errors.

NeuroLucida 360 combines manual neuron tracing with automatic algorithms to provide semi-automatic extraction. The user places seed points for the algorithm by clicking or dragging along the neuron, and thereby guides the algorithm in selecting which data to process. This guided extraction improves the speed at which neuron morphology can be traced, but sections with many labeled neuronal processes still need to be manually resolved or corrected. Similar to *Virtual Finger*, this method is also viewpoint and visibility dependent. These methods work well in many cases; however, as the data size increases, the amount of time spent on the challenging subset of cases grows correspondingly.

Independent of VR, volume rendering systems have been employed for visualization, segmentation, stitching, and tracing of neural microscopy data. Jeong et al. [21] combine segmentation and stitching analysis with an out-of-core GPU volume renderer for large datasets. This system was further improved by Beyer et al. [7] to handle larger and more diverse microscopy data. Wan et al. [44] address the problem of classification of multi-channel microscopy data in volume rendering.

2.2 Neuron Tracing Workflow

A typical neuron imaging and tracing workflow proceeds as follows. First, neurons and their processes are labeled using neuroanatomical tracing methods. Modern approaches to labeling neurons in large brains involve the use of viral vectors carrying the genes for fluorescent proteins [27]. These vectors are injected into the brain to induce expression of these genes within neurons, labeling them at high resolution. Current approaches in connectomics then render the brain optically transparent using clearing techniques such as CLARITY [11], PACT [46], or SWITCH [32]. Imaging labeled neurons through brain tissue, either in brain slices or through whole brains or blocks, produces multiple stacks of images ranging in size from gigabytes to terabytes. Finally, neurons are traced on these 2D image stacks to extract the desired neuronal structures. Depending on the analysis being performed, these structures can be used in simulations or overlaid onto functional maps of the brain, in order to understand the connectivity between brain regions or cells within these regions.

Due to noise in the data, the neuron reconstruction process is often entirely manual. In many instances, several trained undergraduate students are responsible for the bulk of the tracing work. Tracing is done on a desktop computer using NeuroLucida [29]. When working on image stacks using this software, the user scrolls through the stack and clicks to mark points along the neuron or to create branches. However, some branches change depth rapidly, cross in complex ways, or have gaps due to imperfections in the labeling or imaging process, making the structure difficult to resolve.

2.3 Immersive Environments

Virtual reality environments such as CAVEs [12] are effective for enhancing visualization tasks related to understanding 3D data. Prabhat et al. [38] performed a comparative study on confocal microscopy data exploration in desktop, fishtank VR, and CAVE VR environments. They evaluated tasks focused on navigation and observation, e.g., locating and counting features or describing some structure. Users tended to perform better on these tasks in the CAVE environment. Laha et al. [24,25] examined how VR system fidelity affects the performance of common visualization tasks by varying field of view, head tracking, and stereo. The tasks studied were similar to those studied by Prabhat et al. [38] involving search and examination. They found that more immersive VR environments improved users' understanding of complex structures in volumes [25] and isosurfaces [24].

The original CAVE used a three-button tracked wand device to manipulate objects [12]. CAVE2 [14] employs a similar wand controller, using a modified PS3 Move controller. CAVE2 also supports a prototype-tracked sphere controller, the CAVESphere, for moving and interacting with the data, along with a tablet controller showing a webview. Although the CAVE is able to provide high-quality VR, it is a large and expensive system, both to purchase and to maintain, making its incorporation into the routine workflow of scientists in small laboratories unlikely. Direct 3D interaction with the data can also be challenging in a CAVE, since users' hands block the display as they work, occluding their selection. Although well-suited for virtual tours of complex datasets with application-centric software (see, for example, [39]), using CAVEs for day-to-day tasks involving frequent manipulation of data would be costly and challenging.

Due to the difficulty of providing input feedback when working with free-form 3D controllers, many studies have evaluated the use of haptics to provide better feedback to the user. Ikits and Brederson designed the Visual Haptic Workbench [20], which combines a stereo display table with a probe arm. The arm is used to interact with the data and provide haptic feedback. For example, when tracing a streamline, the probe will be constrained to follow it. Palmerius et al. [33] described a system of primitives for computing haptics on volumetric data, e.g., for directional or vibration feedback, which can provide a greater sense of touching structures in the environment.

3 Design Process

Independent of the current effort, we developed a technology probe (Section 3.1) with the goal of investigating consumer-grade VR technology for scientific visualization. One of the datasets we used during testing was a large microscopy scan acquired in the laboratory of one of the authors (A.A.), which we down-sampled to fit on the GPU. Positive feedback from an incidental demonstration of this probe prompted us to explore the use of this technology for neuron tracing. Subsequently, we designed our tool in close collaboration with expert neuroanatomists in an open-ended, iterative process, influenced by the nine-stage framework of Sedlmair et al. [42]. Through several iterations, we distilled the fundamental user requirements and added the necessary features to arrive at the tool discussed in Sections 4 and 5.

3.1 Technology Probe

The initial application was designed to explore the potential of using VR systems for generic scientific visualization tasks, and supported volume rendering, isosurfaces, and particle rendering (Fig. 2a). A user sits or stands at a desk and is able to move his or her head to look at the volume, or translate and rotate it using a gamepad. An initial demonstration of this system with microscope scans of labeled neurons encouraged us to further pursue this as a neuron tracing tool. In particular, the neuroscientists on our team noted that, compared to standard 2D interfaces, the VR system allowed better perception of the spatial relations between neurons, one of the key challenges in neuron tracing. However, in our initial investigation, the ability to interact with the data was limited by the restriction of the Oculus DK2 head-mounted display (HMD) [2] and the use of a gamepad as the input device. The Oculus DK2 can track small head movements while the user is facing a webcam style tracker, but does not support walking around a room. Although the gamepad can be configured as a 6 DOF controller, it is not tracked and thus cannot be used to reach out and "touch" the data directly.

The desire for direct 3D manipulation led us to pursue a different interaction paradigm. To this end, we moved to the HTC Vive platform [3], which supports room-scale VR and includes tracked, wand-style controllers. The room-scale tracking allows users to walk around, as well as into the data, and interact with it naturally using their hands. Tilt Brush [17], which uses the same wands to paint in 3D, inspired the first prototype of our tool, which extended the painting metaphor to neuron tracing.

3.2 The Prototype

We designed the first prototype dedicated to neuron tracing to evaluate what different types of interactions would be useful, and explore how they could be mapped to the HTC Vive's control scheme. Based on the available space and hardware setup, we created a medium-sized tracked area, about $2.5m \times 2m$, and placed the data in a $1.5^3 m$ box in the center of the room at about $1m$ above the ground. The prototype used both wands, one to interact with the data and the other to navigate the space.

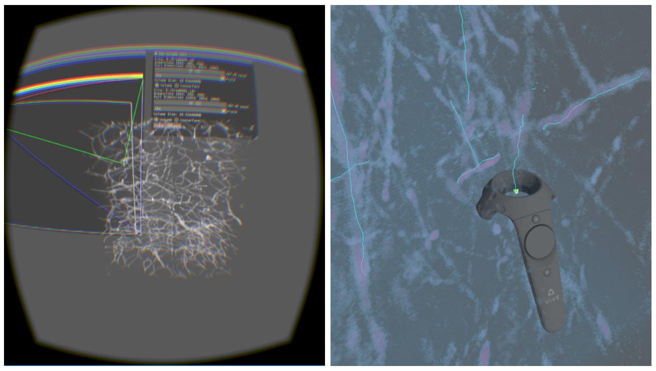

(a) Initial Technology Probe (b) Prototype

Fig. 2: The technology probe and prototype were used to explore different interaction and rendering possibilities for scientific visualization and neuron tracing in VR.

Using the first wand, the user could hold a button and draw a line coming from the tip of a tetrahedron shown in the middle of the wand's loop (Fig. 2b). As only a subregion of the data could be rendered at a sufficient frame rate for VR (see Section 4.2), we rendered a 256^3 subregion of the volume – the *focus region* – in the $1.5^3 m$ box. The second wand could then be used to grab and move the data within this region. To orient users within the dataset, we displayed a minimap of the dataset bounds and the focus region location within it. One notable observation was that given the opportunity to pan, users often preferred to drag neurons closer as opposed to walking toward them.

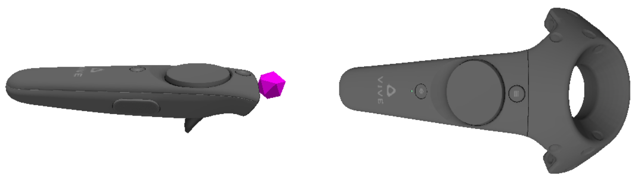
As our target users are not familiar with transfer function design, even in a desktop setting, we chose a preset for the datasets, allowing us to focus on just the task of tracing. Selecting from chosen presets has been found effective in medical visualization and museum installations [47], where users are also unfamiliar with designing transfer functions. Moreover, designing an effective interface for specifying transfer functions in VR is an open and challenging problem.

To evaluate the initial design of the tracing interaction, we asked expert neuroanatomists to trace neurons in some datasets acquired in A.A.'s laboratory. After a short introduction to the control scheme (about 10 minutes), they were free to use the tool as desired. These users noted that the painting metaphor was intuitive. Compared to existing 2D tools, they found the prototype easier to use for exploring the data, allowing them to better resolve complex crossings and spatial relations of neurons in the data.

The prototype, despite being limited to line drawing and simple exploration, provided an initial validation of both the navigation and interaction design. To extend this prototype to a minimally viable tool, we added additional features that are typically used by neuroanatomists in the neuron reconstruction and analysis process in NeuroLucida. For example, color is used to distinguish axons and dendrites, and glyphs to mark areas of interest. NeuroLucida also allows for undoing operations and editing previous traces, which permits review and correction of previously traced neurons. Therefore, we extended our initial prototype by improving the tree drawing system and rendering quality, and added support for undoing and editing, placing markers, selecting line colors, and streaming large volumes from disk. Furthermore, we continued to expand the interaction paradigm by integrating haptic feedback. These improvements are incorporated into the current tool we describe in the following section.

4 Virtual Reality Tracing Tool

The design of the final tool focuses on two key aspects: the process of tracing and navigating (Section 4.1), and meeting the VR rendering performance requirements to provide a high-quality experience and prevent motion sickness (Section 4.2). To analyze how scientists use our system and allow for its use as a training device, we also provide a recording and playback system (Section 4.3) that tracks the user's actions rather than a video stream. Finally, our tool must fit into a larger data processing pipeline, which starts at the acquisition of volume data



(a) Tracing wand (b) Navigation wand

Fig. 3: The wand model shown in VR can be changed from the physical model. On the tracing wand (a), we removed the top loop seen in (b) to avoid occlusion while tracing. The button sticking out underneath (a) is the trigger, and the large circular button is the trackpad. The icosphere brush in (a) is colored to match the selected line color.

from a microscope and ends with the simulation and analysis of the reconstructions in the context of other brain maps. To fit well into the pipeline, our tool loads the IDX [34] volume format used by our collaborators. Once the neurons of interest have been reconstructed, the data is exported in a standard XML format used by NeuroLucida. Furthermore, previously traced neurons in this format can be opened in our tool, allowing for inspection and editing of earlier work.

4.1 Tracing and Navigation

Tracing neurons and navigating the data are the key tasks when reconstructing neurons. Both interactions require the 3D motion to be intuitive, and therefore we map these interactions to the motion of each wand. One is used for tracing and the other for navigation (Fig. 3). Tracing and navigation actions are initiated by holding the trigger button on the corresponding wand. In the VR environment, the tracing wand is displayed with an icosphere at the top, indicating from where the line will be drawn, similar to a paint brush (Fig. 3a). The navigation wand is rendered to match the wand's physical model (Fig. 3b).

A neuron forms a tree that consists of a starting point, branch points, and termination points. Traces created by the user are stored in a graph structure that we update with the user's edits and additions. To trace a neuron (Fig. 4), the user presses and holds the trigger button on the tracing wand, placing a starting point. The user then holds the trigger as he or she follows the neuron through the data, drawing a line from the brush. Releasing the trigger ends the line and creates the termination point. The user is then free to continue the line from the termination point, or trace branches as needed.

Tracing the branches of a neuron correctly is critical to properly recover its connections and structure. Moreover, this task is performed often, and therefore it must be easy to do. To create a branch, the user can start a new line along the current tracing and follow the neuron branch out (Fig. 5a), or start a new line on the neuron branch and reconnect to the parent tree (Fig. 5b). To call attention to the reconnection, we highlight the selected node and send a small vibration to the wand to give a "click" feeling of selecting it. When connecting back to a line, the candidate node that would be created when the trigger is released is displayed as a small cube to indicate where the branching point will be placed (Fig. 5b). The visual and physical feedback provides a clear signal to the user that the connection has been selected as desired.

During the tracing process, mistakes may be made that need to be corrected. For example, the user may have an incorrect initial understanding of a complex crossing, the user's hand could slip, or the system could drop a frame or momentarily lose tracking due to occlusion. Depending on the type of mistake, the user may make an immediate correction or revisit the error later. To correct mistakes, the tool provides two methods of undoing and editing: a quick fine-grained undo operation and the ability to remove entire lines and nodes at any time.

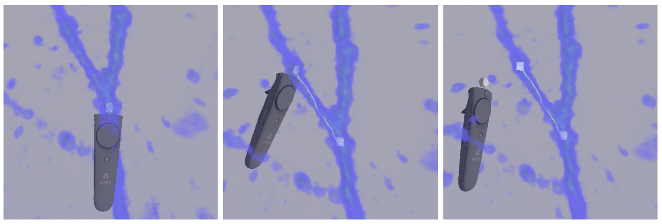
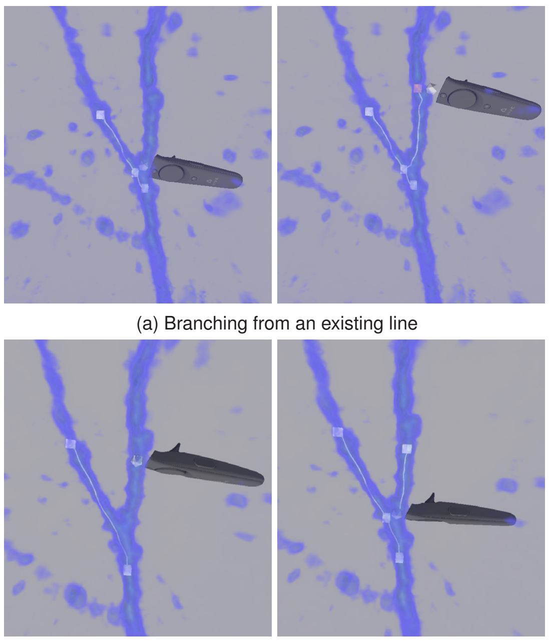


Fig. 4: From left to right: the neuron tracing process begins by finding a neuron. A starting point is placed by moving the brush inside the neuron and pressing the trigger. While holding the trigger, the user follows the neuron with the brush, tracing it. To end the line, the trigger is released.



(a) Branching from an existing line

(b) Connecting a branch back to the parent tree

Fig. 5: A branch can be created by placing the brush close to an existing line, where a candidate branch point will be shown (a), or an existing node, and tracing from it. The branch can also be started as a new line and re-connected to the parent tree (b), in which case the candidate branch point created by the connection is shown.

To immediately correct mistakes, the user can undo lines in the reverse order in which they were created by pressing the trackpad (Fig. 3). This undo is useful for quickly repainting segments where the user is not satisfied with how well the trace follows the neuron. The scope of the undo operation is controlled by placing undo breakpoints along the line every 40 voxels, with each undo operation reverting to a previous breakpoint. Furthermore, any part of the line can be repainted by drawing a new line over the problematic section and reconnecting it after the section. The new line forms a loop in the trace, and the old section will be removed to reduce the graph to a tree. Since a neuron is physically a tree, any loop represents an invalid structure and can be assumed to be an edit.

Scientists may notice errors when revisiting a previously traced section. The undo and line redrawing operations may not be applicable in such instances. Instead, the user can delete specific lines or nodes with the tracing wand. Editing operations are initiated by selecting a line or node with the wand, noted by highlighting the feature and a "click" vibration, and pressing the undo button. The user can then reconnect the disconnected trees as desired. For example, in Fig. 5a the selected line (left) or the highlighted node and attached edge (right) could be deleted by pressing undo.

Navigation around the dataset is accomplished by walking or by translating the volume. Within the focus region, the user is able to walk around the space to navigate. To explore data outside the focus region or pull the regions closer, a panning action is mapped to the navigation wand. By holding the trigger button and moving the wand, the user grabs the focus region and translates it through the volume. Via this interaction, arbitrary-sized volumes can be explored in our system. Furthermore, as volume sizes are often larger than available GPU memory, the data is paged on and off as the user pans, described

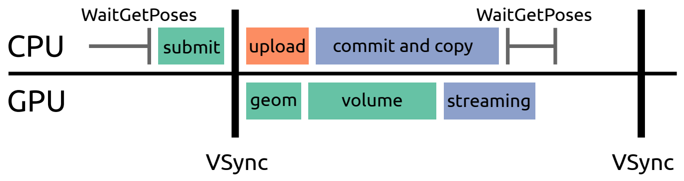


Fig. 6: The anatomy of a single frame. `WaitGetPoses` blocks until ≈ 2ms before VSync and returns the latest head tracking data. This allows the renderer to start submitting work before VSync to fully utilize the GPU. We first submit draw calls for the geometry and volume, and then page in asynchronously uploaded volume data into the sparse texture.

in detail in the following section. To help the user track the location of the focus region relative to the dataset and previously traced neurons, we display a minimap in the corner (Fig. 1). When navigating large datasets, the minimap is useful to keep the user oriented as he or she pans through the space. Traced neurons are also displayed in the minimap to help the user track their progress through the dataset.

4.2 Rendering

The HTC Vive uses a display panel with a resolution of 2160×1200, providing 1080×1200 pixels per eye at a 90Hz refresh rate. Furthermore, due to lens distortion, it is recommended to supersample the image, effectively doubling the number of rendered pixels. Additionally, the VR environment imposes stringent lower bounds on the acceptable frame rate to avoid motion sickness. The combination of high-resolution and frame-rate requirements presents a significant challenge compared to traditional desktop visualization, where low frame rates and intermittent pauses for computations or data loading are more tolerable. To meet these requirements, we take cues from best practices for VR game development [43]. In order to communicate with the HTC Vive HMD, we use the OpenVR SDK, which provides methods for sending images to the eyes and tracking the head and wand positions.

At 90 FPS we have a tight budget of about 11ms to render each frame, from which 1ms is potentially consumed by the operating system. Furthermore, as GPUs are pipelined architectures, submitted work is not executed immediately, but enqueued into a command buffer. To account for this, Vlachos [43] recommends submitting draw calls ≈ 2ms before VSync. Submitting early allows the GPU to start rendering immediately after presenting the previous frame, increasing utilization.

Streamlining rendering performance requires pushing all non-rendering or non-critical work onto background threads and strictly budgeting work on the render thread. A single frame is divided as shown in Fig. 6. First, we wait until ≈ 2ms before VSync by calling `WaitGetPoses` from the OpenVR SDK (left side of CPU in Fig. 6), which obtains the most recent head position. After returning from this function, we submit all rendering work to the GPU. Opaque geometry, e.g., the wands and tracings, is rendered first (1ms). Next, the volume is rendered with raymarching to display a volumetric or implicit isosurface representation (4ms). After submitting the rendering work, we start the asynchronous volume data upload based upon the user's focus region, and once the rendering finishes, we copy it into the sparse texture (2ms). This time budget leaves a buffer of 3ms to prevent unpredictable interferences that could cause dropped frames. Nevertheless, sometimes a system event or an expensive draw call can consume this buffer, causing a frame to be skipped. When this occurs, the OpenVR SDK will automatically render at half frame rate, 45 FPS, while reprojecting the last frame using the latest head tracking information to display at 90 FPS, until the frame rate improves. Unfortunately, the reprojection cannot account for the wands' motion, as the image transform is based only on the head motion. When the system is reprojecting, the wands appear to stutter, making the interaction feel sluggish.

4.2.1 Data Streaming

Typical microscopy volumes exceed the VRAM of current GPUs (4-24GB) and in most cases the RAM of typical workstations (64-128GB); datasets can range from hundreds of gigabytes to terabytes. Exploring such datasets inherently requires a data streaming solution. Moreover, as only 2ms is budgeted for data streaming on the render thread, we must amortize the work of updating the volume data over multiple frames, and perform as much work as possible asynchronously. We use a two-level caching system: the first level loads and caches pages from disk into RAM, and the second level takes these pages and uploads them to the GPU. The caching system lets the tool keep the current focus region and a small neighborhood resident on the GPU, while a substantial history is cached in RAM. The cache drastically reduces disk access frequency and latency to display pages as users navigate.

The first-level cache takes page requests and immediately returns a future [6], which can be used to retrieve the page data. In case the page is not available in the cache, a worker thread will be responsible for loading the data from disk while the requester can asynchronously check for completion and retrieve the page. The second-level cache pushes page queries to a set of worker threads, which request the page from the first-level cache and copy the data into persistently mapped pixel buffer objects (PBOs). By uploading via persistently mapped PBOs, we take advantage of asynchronous data transfers via the GPU's copy engines, thereby overlapping rendering work with data transfers.

We store the volume data on the GPU in a sparse 3D texture, a form of virtual memory where individual pages can be committed or decommitted. This texture allows for transparent handling of volume data larger than VRAM. The rendering work for a frame takes long enough for the asynchronous data upload to complete, after which the page is copied into a newly committed page in the sparse 3D texture (commit and copy, streaming segments of Fig. 6). The system uploads only a limited number of pages per frame to stay within its time budget and decommits pages no longer needed.

To minimize visible popping of pages into view, we load a box slightly larger than the focus region and prioritize pages closer to the user's view. To avoid overwhelming the paging system by requesting many unneeded pages, e.g., in the case of quickly panning through the space, for each frame we enqueue only the four highest priority pages that are not already being uploaded. Additionally, if a page is no longer needed by the time the PBO is filled, we do not commit the page or copy it to the texture, as it would be immediately decommitted. We find this scheme of limiting the enqueueing rate simpler compared to updating priorities for already scheduled pages in a non-blocking thread-safe manner.

4.2.2 Volume Rendering

We use a GPU volume renderer written in GLSL [18]. Although the volume data is stored in a sparse 3D texture, this texture type requires no additional consideration in the GLSL code. Sampling missing pages is defined to return 0. In the raymarching step, we use the depth buffer produced by rendering the opaque geometry to terminate rays early in order to correctly composite with the geometry, wands, and tracings.

In a VR application, it is common to step inside the dataset. When walking through the volume we observed it appeared to vibrate, or the isosurface to move subtly. As the camera moves through the volume, the voxels sampled by rays leaving the eye will be offset differently in the data, causing them to sample slightly different locations. This artifact can be mitigated by increasing the sampling rate, but this is prohibitively expensive for VR. At its core, the issue is similar to ensuring correct ray sampling across subvolumes in distributed volume rendering [28]. To ensure consistent sampling of the data when inside the volume, we begin sampling at the sample point nearest to the clipping plane, based on starting the ray from its entry point into the volume bounds. This approach corrects only for translation, but we found it sufficient in practice.

Even when using gradient shading, depth perception can be challenging in volume rendering, particularly when using transparent transfer functions and subtle lighting cues. This lack of depth cues can make it difficult to tell the exact position of the wands when placed inside neuronal structures, due to the faint occlusion effect provided by the volume. In fact, we had one user report experiencing eye strain while viewing the volume representation, potentially due to the limited depth cues. More advanced rendering techniques such as shadows or global illumination [22] can improve depth perception, and potentially user performance, but come at significant frame-rate or memory cost, making them challenging to apply in a VR setting. To maintain a sufficient frame rate for VR, the volume rendering quality in our tool is relatively simple, providing just gradient shading.

To enhance depth cues, we added the ability to switch to an implicit isosurface mode, with Phong shading and ambient occlusion [19]. In this mode, the user can scroll on either of the two wands' trackpads to change the isovalue, which is necessary to resolve crossings or neurons with low intensity values. The front faces of the isosurface are rendered to be semi-transparent, allowing the user to see when the brush and traces are placed well inside the neuron. The back faces, however, are rendered to be fully opaque, as it is difficult to perceive depth relations when they are semi-transparent. When dealing with noisy data such as microscopy images, isosurfaces are often not ideal, as the noise manifests as small objects in the volume. Especially in a VR environment, these objects result in distracting aliasing artifacts. To counter this effect, we de-noise the data by filtering out objects less than 11 voxels in size before uploading the page to the GPU by finding small connected components.

The performance of volume raycasting is directly tied to the number of rays rendered, i.e., the number of shaded pixels. The rendering resolution recommended by the HMD is very high; fortunately, this resolution is needed only at the center of each eye. In the periphery, due to lens distortion and the properties of the human vision system, it is possible to render at much lower resolutions (e.g., using the `NV_clip_space_w_scaling` extension) with little to no perceived difference.

4.3 Recording and Replaying

Evaluation and iteration of any tool requires understanding how it is used. Moreover, a recording of an expert's session can serve as training material for novices. In desktop applications, user sessions can be recorded using screen recording software. However, in VR recording the "screen" restricts the playback to a single viewpoint, potentially removing relevant context in the space. For example, a mistake made when tracing may depend on the user's viewpoint, but to properly observe the situation, it must be possible to watch the user's session from a viewpoint different from the recorded one. Even more concerning is that viewing the recording in VR typically induces nausea due to the mismatch in the recorded head motion and the viewer's head motion. To this end, we have developed a recording and playback system for tracking user sessions that is based on the actions performed by the users, instead of video recording.

Such action-based recording can be performed at multiple levels. The low-level wand and HMD state and poses could be saved each frame, the tracing could be played back by stepping through snapshots, or the user's logical operations (e.g., *tracing*, *toggled isosurface*, or *panning*) could be saved. The last option provides the most flexibility for both playback and later analysis. This option also supports playback of the recording on different VR systems or later iterations of the tool with different control schemes, without needing to re-map low-level button presses or HMD information. Moreover, session analysis is easier with such a representation, as queries can be made at a higher level, e.g., "how far does the user trace in a single motion?" or "how long did they use each rendering mode?".

Replaying a saved session recreates the entire tracing by moving the wands and HMD as they were during the session. By viewing the hand and head motions during the replay, we can observe differences in how users work. During this time, the user viewing the replay can walk around the space independently. To better demonstrate the replay capability and the tool itself, we include a video taken while replaying an expert's session in the supplementary material.

5 Evaluation

To evaluate our design choices and compare our tool to the state of the art, we conducted a pilot study with neuroanatomists who are familiar with neuron tracing and the existing software (NeuroLucida). Specifically, we present two case studies with seven users tracing neurons in two different datasets. In practice, the two primary metrics of concern are accuracy and speed. In terms of accuracy, the goal is to determine the connectivity of neurons as well as possible, including geometric location and tree topology. Misinterpreting a crossing as a branch point or missing branches entirely will cause substantial errors in the subsequent analysis. Nevertheless, as usual in expert-driven systems, the final result is subjective, and experts sometimes disagree on specific choices. Furthermore, some mistakes are more critical than others. Slightly elongating a trace by crossing a small gap may be acceptable, but erroneously attaching a branching structure is not. In terms of speed, the field of connectomics is moving to acquisition of ever increasing amounts of data; therefore, the time necessary to trace neurons is of significant concern.

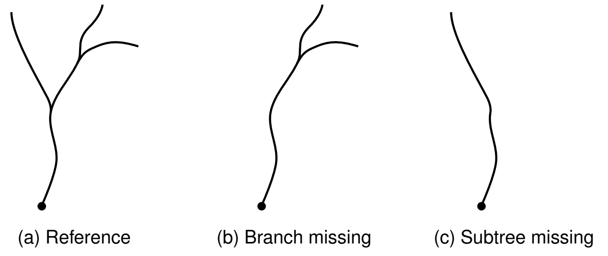


Fig. 7: Examples of different mistakes and their effect on the DIADEM score. Trees (b) and (c) are compared against the reference (a) with scores 0.875 and 0.5, respectively. The error in (c) misses a large subtree, impacting later analysis more significantly than that in (b).

DIADEM Scores. In order to automatically compare traces, we used the DIADEM scoring method [15], which takes into account both the length and the connectivity of a trace. The computed score correlates well with expert judgment, and informal comparisons suggest it is a reasonable proxy for accuracy. The score measures the similarity between traces with values ranging from 0 (dissimilar) to 1 (identical). For example, missing a small branch in a large tree (Fig. 7b) has a smaller impact on the score than missing a large subtree (Fig. 7c).

Case Studies. The first dataset consists of six aligned subvolumes containing 34 mostly planar axons, each with a reference tracing (Section 5.1). The second dataset is a single large volume containing several noisy cell bodies (Section 5.2). In both cases, we provided a predetermined set of points from which users start tracing a neuron in each tool. To avoid bias, we split all starting points into sets traced on alternate days in different tools, such that no set was traced on consecutive days. In most cases, there were multiple days between sessions, due to users' work schedules. For the first case study, we report scores with respect to the reference traces, whereas for the second case study, we compute scores by comparison with traces performed by domain experts. We also collected qualitative feedback from the users during the sessions, and had each participant complete a questionnaire at the end of every tracing session.

We report results for seven users, including two senior neuroanatomists (users 4 and 6), two expert undergraduate students with 2-3 years of experience reconstructing neurons with NeuroLucida in A.A.'s laboratory (users 5 and 7), and three undergraduate students with no background in neuroanatomy (users 1-3). In a typical lab, the bulk of tracing work is performed by trained undergraduates (e.g., users 5 and 7), who start with little background (e.g., users 1-3) and are trained by senior members of the lab (e.g., users 4 and 6). By evaluating with a cross-section of the experience levels found in a typical lab, we can determine how well our tool fits into existing workflows. Specifically, the tool must be usable by senior members and expert students for tracing, and be easy to learn for new hires such as users 1-3. Based on the range of experience with tracing in NeuroLucida, we binned the users into two groups, an *experts* group, consisting of the neuroanatomists and the expert undergraduates, and a *novices* group, with the three inexperienced undergraduates. During our evaluation, the VR tool ran on a workstation with a dual socket Intel Xeon E5-2680 CPU, 64 GB RAM, an NVIDIA GTX 1080 GPU, and an SSD.

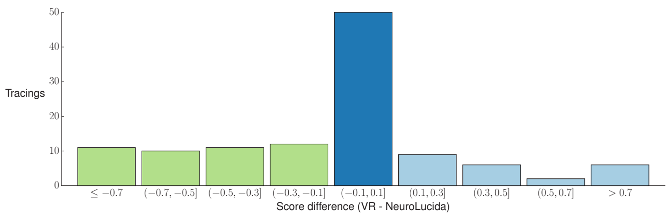
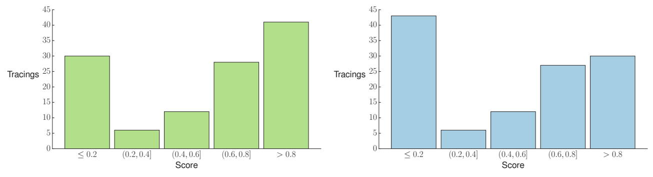


Fig. 8: Differences between scores of expert traces in VR vs. NeuroLucida. For each neuron traced, we compute the difference in score achieved compared to the reference between the two tools. We find that overall experts performed within the acceptable error range (± 0.1, dark blue) and sometimes better in VR (light blue) when compared to their work in NeuroLucida.



(a) NeuroLucida (b) VR

Fig. 9: Distribution of scores (higher is better) for experts. In (a) median score: 0.7, mean score: 0.57 ± 0.38. In (b) median score: 0.6, mean score 0.49 ± 0.39. A score of ≥ 0.8 is a tracing acceptably similar to the reference.

5.1 Planar Axons Reference Dataset

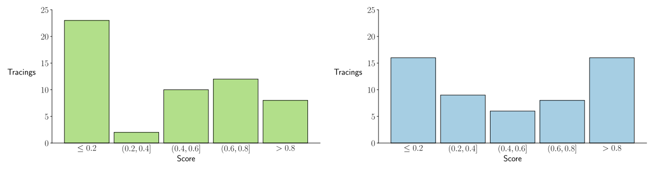
Although all tracings can contain some subjectivity, it is important to establish a baseline of performance with respect to a given reference. Here we use the *Neocortical Layer 1 Axons* dataset [1] from the DIADEM challenge [15] and the corresponding reference traces. The dataset consists of six volumes of neurons in a mouse brain that can be stitched to form a $1464 \times 1033 \times 76$ volume with the provided alignment information. The resolution of the data is $\approx 0.08 \mu m/\text{pixel}$ in X and Y and $1 \mu m/\text{pixel}$ along Z.

The dataset includes 34 reference tracings, of which we used the first two for training and the rest for evaluation. For each neuron, users started from the first point of the reference tracing and traced the corresponding neuron to its perceived termination points. Once all sessions had been completed, we compared the results from each tool with the provided reference tracings. In general, our experts rated the reference tracings as acceptable reconstructions, with the exception of a few neurons where branchings were judged to be crossings, or a crossed gap was considered too wide. Table 1 shows, for each user, the mean score, reconstruction time, and speed-up across all 32 evalution traces. Speed-up is defined as the average time per tracing in NeuroLucida divided by the average time per tracing in VR.

When comparing the scores for traces done by the experts in NeuroLucida vs. those done in VR (Fig. 8), we found that in most cases the traces performed were acceptably equivalent in both tools (dark blue bar, Fig. 8), with some neurons traced better in each tool (green and light blue bars, Fig. 8). Overall, there was no statistically significant difference between the scores achieved in VR vs. NeuroLucida (Mann-Whitney $U = 6426.5$, $n_1 = 122$, $n_2 = 120$, $p = 0.097$). The distributions of scores for experts (Fig. 9) indicate that experts can achieve similar, and sometimes better, tracing quality in our VR tool when compared to their current workflow. In cases where experts produced equivalent quality traces in both tools, we find a statistically significant speed-up, with experts being on average $1.7\times$ faster tracing in VR (Mann-Whitney $U = 1004.5$, $n_1 = 54$, $n_2 = 54$, $p = 0.005$). Moreover, expert users were similarly consistent in VR and NeuroLucida, as indicated by the mean of standard deviations on each trace, 0.23 and 0.24, respectively.

In fewer cases (37%, green bars in Fig. 8), the experts performed better in NeuroLucida than in VR, beyond the acceptable error bounds. When investigating these cases, we found that they involved the same neuron for all experts, with each expert making the same mistake. One such neuron is the eighth neuron from the dataset, where a stitching issue was misinterpreted as two neurons passing each other in VR (Fig. 11). The VR tool performed better in other cases (19%, light blue bars Fig. 8) where neurons traveled along the Z axis down through image slices, as this is much harder to follow in NeuroLucida, requiring scrolling through the image stack (Figs. 12 and 13).

Novice users performed similarly, with 72% of their traces falling within acceptable error or being better than those performed in NeuroLucida (Fig. 10). On average, novices were $2.1\times$ faster in VR on traces where they achieved similar scores in VR and NeuroLucida. We do not report more significant results for novices due to the limited data collected. We discarded user 1's results from the summary statistics entirely, as the user made a mistake in the NeuroLucida sessions and miscalibrated the Z level of the traces.

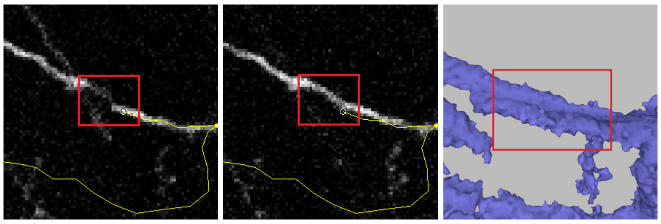


(a) NeuroLucida (b) VR

Fig. 10: Distribution of scores (higher is better) for novices, excluding user 1. In (a) median score: 0.5, mean score: 0.42 ± 0.37. In (b) median score: 0.49, mean score 0.5 ± 0.37. A score of ≥ 0.8 is a tracing acceptably similar to the reference.

User	NeuroLucida		VR		Speedup
	Score	Time	Score	Time	
1*	0.27 ± 0.35	324	0.49 ± 0.35	172	1.9
2	0.34 ± 0.34	188	0.54 ± 0.37	161	1.2
3	0.48 ± 0.38	277	0.45 ± 0.36	149	1.9
4	0.57 ± 0.38	412	0.57 ± 0.36	271	1.5
5	0.56 ± 0.37	237	0.41 ± 0.38	151	1.6
6	0.65 ± 0.35	464	0.50 ± 0.40	229	2.0
7	0.51 ± 0.38	262	0.47 ± 0.41	302	0.9

Table 1: Average scores (with standard deviation), and times in seconds for each tool across the 32 DIADEM traces used for evaluation. We also computed average speed-up over all traces for each user. Users 1-3 are novices and 4-7 are experts. Both novices and experts performed similarly in VR, and tended to be faster on average. *User 1 miscalibrated the Z level in their NeuroLucida sessions, resulting in much lower scores for the majority of traces.



(a) (b) (c)

Fig. 11: A stitching issue clearly visible in NeuroLucida (a-b), but difficult to perceive with volume rendering or isosurfacing (c). What appears as two neurons (c) is in fact a single neuron, slightly misaligned due to stitching issues at the border of two acquisitions. When scrolling through the image slices in NeuroLucida, the stitching issue can be seen by flipping between the slices (a-b) and those above and below. In NeuroLucida, all experts traced the neuron correctly, whereas in VR only one expert traced it correctly. We note that this issue is not specific to VR, but to the volume visualization method chosen.

5.2 Cell Bodies Dataset

To compare usability on a dataset with neuronal structures of interest, we also evaluated our experts' performance on a dataset acquired in

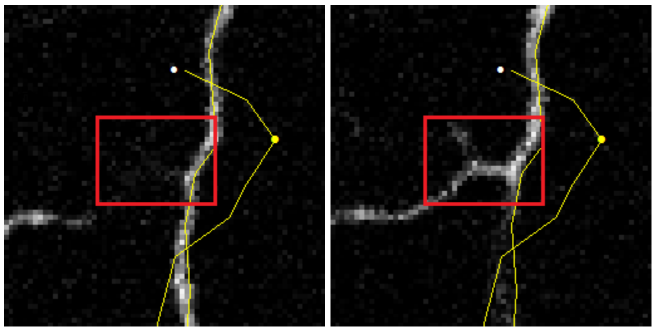


Fig. 12: A neuron branching along the Z plane is not visible on the image plane used to trace the main structure (left). The branch can be seen only after scrolling down the stack (right). Only two experts traced this branch correctly in NeuroLucida, but in VR all users traced it correctly.

User	VR Score	Time (s)
1	0.54 ± 0.23	537
2	0.58 ± 0.17	252
3	0.70 ± 0.23	207
5	0.66 ± 0.19	360
6	–	469
7	0.59 ± 0.34	542

Table 2: Average scores (with standard deviation) and times for traces on the Cell Bodies dataset. User 6 is used as the reference.

A.A.'s laboratory. The dataset, shown in Fig. 1, consists of neurons in the visual cortex of a marmoset monkey labeled with green fluorescent protein, and was acquired in 2012 using a 2-photon microscope. The volume is $1024 \times 1024 \times 314$ with a resolution of $0.331 \mu m/$pixel in X and Y and $1.5 \mu m/$pixel in Z. The neuronal structures in this dataset branch significantly more often than those in the dataset described in Section 5.1. Moreover, this dataset has a higher level of noise and frequency of ambiguous cases, and is therefore more challenging to trace. In this evaluation, we were concerned with scaling in the sense of cognitive load of the user, not necessarily data size. We selected five starting points in the data, to be traced by the experts in VR. Furthermore, as there is no reference available, we measured performance by selecting user 6's tracings as the reference (Table 2).

We compared the proportion of time spent on tracing or panning during each trace. Since this dataset is more complex, we were interested in whether users would use the tool differently with respect to the more planar dataset in Section 5.1. On average, users spent 15% of their time tracing and 48% panning, which is only slightly different from the 21% and 58%, respectively, in the *Neocortical Layer 1 Axons* dataset. Users toggled between the volume and isosurface rendering modes more frequently in this dataset. Although this result is interesting, it requires further evaluation on multiple datasets and a more rigorous measure of data complexity to provide meaningful evaluation.

We replayed several of user 6's sessions and noted that he was changing viewpoints frequently to check potential branchings and obtain a better understanding of the data. Such frequent viewpoint manipulation in NeuroLucida requires moving back and forth through hundreds of images. Furthermore, during these sessions several experts remarked that they would prefer to trace this data in VR, as scrolling through image stacks in NeuroLucida becomes more difficult as the dataset thickness increases.

5.3 Discussion

Our design study consisted of open-ended feedback sessions with neuroanatomists and quality evaluation of the tracings produced by users of our VR system compared to state-of-the-art desktop software, NeuroLucida. Additional feedback was collected during the evaluation through a survey filled out at the end of each tracing session, and by soliciting feedback with regard to the usability and comfort of each tool. This section describes our users' qualitative responses to our

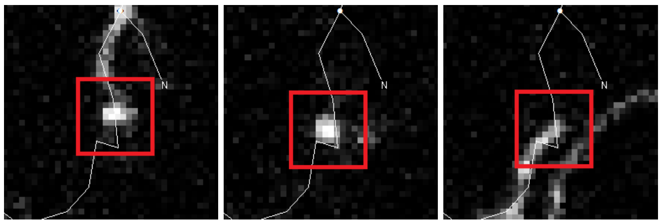


Fig. 13: From left to right: a neuron travels vertically through consecutive slices, appearing as a dot (middle) in these images. In NeuroLucida, only two experts traced this correctly, but in VR all users except one expert traced it correctly.

system regarding neuron tracing, navigation, and rendering. Moreover, it discusses the overall strengths and limitations of our current tool.

Tracing. The experts reported that tracing neurons, creating branch points, and correcting mistakes were more intuitive in the VR tool than in NeuroLucida. The combination of tracing in free space with a single button press and the flexible graph editing system allowed users to focus on the data, instead of having to continuously flip back to a toolbar to mark branches and termination points, or frequently scroll through the image stack. We recorded similar responses in the survey, with users rating the VR tool easier to use for these tasks. However, experts expressed the need for fine manipulation of lines and nodes to edit previous tracings, without going through a delete and re-trace interaction. When analyzing user sessions, we found that users employed the quick undo command more often than explicitly deleting lines or nodes. We hypothesize that the delete and re-trace feature is less intuitive, or that it may be more applicable to post-trace editing sessions. Designing an intuitive system for editing previous tracings in VR poses an interesting challenge. Introducing additional button commands or menus could make the system non-intuitive, and manually switching between tracing and editing modes could break the "flow" of a user during the task.

When replaying the tracing sessions, we noticed that some errors in traces produced in VR were due to users forgetting to return to a branch point. In NeuroLucida, branches and termination points are explicitly marked; when a branch is ended at a termination point, the system scrolls the user back to the branch point to trace the rest of the branches. With our graph-based editing system, creating branches and termination points is implicit, and the user must remember to return to the branch point after completing the current branch. In the VR sessions, some users placed markers at complex crossings or branch points, as reminders to revisit the location later. Users also requested the ability to hide the dataset entirely, allowing them to observe the traced tree structure independently. Our prototype included the ability to hide the rendering by panning the focus region outside the volume and observing the tree from a distance, or viewing the minimap. However, providing an explicit option would be desirable in future iterations.

Navigation. The users easily adapted to navigating by grabbing the focus region and moving it. During the tracing sessions, some users chose to sit, and the panning system allowed them to easily bring the data closer. One user commented that he felt as productive sitting as he did standing. The current version of our tool does not support rotating the volume, due to the inherent ability in VR to simply walk around the dataset instead. However, this feature would be useful when tracing while seated. Instead, when seated, users moved the volume behind them and spun the chair around to view the data from the opposite side. In one case, an expert did not perform this less intuitive action, and misinterpreted a crossing as a branch point. When asked to re-trace this neuron while standing, the expert correctly resolved the crossing by observing it from a different angle. We include a video of these two sessions in the supplementary material.

Users found the minimap to be somewhat useful, especially for displaying the tracings; however, users reported rarely looking at it during active tracing, as it is small and tucked away in a corner. Users did report finding it useful for navigating to the starting point and reviewing the trace. We suspect the minimap might be more useful for larger datasets, where orienting oneself becomes more challenging. Additionally, in some cases users misinterpreted a neuron as terminating, when it was in fact just at the focus region boundary. Based on this feedback, we now display the volume bounds in the world space to clearly convey the dataset bounds.

Rendering. Neuroanatomists are often not familiar with typical scientific visualization representations such as volume rendering and isosurfaces. For example, when viewing the volume representation, users often misinterpreted stitching artifacts between acquisitions (Fig. 11). After a second VR session, one of the experts mentioned preferring the volume representation after gaining more understanding of what is shown. In his first session, he primarily used isosurfaces, but found them to be potentially deceiving, as neurons may manifest at some isovalues but not at others, and can appear to change thickness as the isovalue is adjusted.

Users also raised concerns that the volume and isosurface representations could hide or filter out faint or fine-detail features in the data, such as spines or boutons (small important structures present on dendrites and axons, respectively). Expert users also suggested introducing the option of viewing the original microscope image slices within the volume data, in order to supplement the new representation with something more familiar to the neuroanatomist. It would also be valuable to add support for clipping planes to cull out noisy or dense regions of the data. During the evaluation sessions, we observed users employing the focus region bounds as a form of clipping plane, by panning the data in and out of view, indicating a need for this feature.

When reviewing traces in the reference datasets where users performed poorly in VR, we found that some of these cases involved crossing a gap in the data, where the labeling of the neuron was faint or incomplete. In NeuroLucida, users correctly perceived this gap as caused by non-uniformity in the signal. However, no scale bar was displayed in our VR tool, which could lead users to misinterpret the size of gaps or structures they are seeing in physical units.

Novice users found the 3D representation helpful in understanding the 3D nature of the neuronal structures. One novice mentioned that after using the VR tool, she was better able to construct the 3D structure mentally when working on 2D image slices in NeuroLucida.

6 Future Work

The results of our evaluation are promising, but several additions to our tool could improve users' performance. One useful modification would be to provide additional guidance during the tracing process, by highlighting potential errors and reminding users to return to branch points. For example, trifurcating branch points occur rarely in neurons and, if created by the user, could be automatically highlighted as potential errors. Unsupervised machine learning techniques, such as clustering, could be used to automatically compute the likelihood of sections of a trace by comparing cluster size, and utilize more complex features of the data.

Editing and reviewing traces could be improved by supporting moving nodes and lines. However, developing a natural interaction for editing in VR presents a challenge. As discussed in Section 5.3, adding more complex button combinations or system menus can increase the cognitive load for users and reduce productivity. It is also unclear how to best manipulate the graph. In NeuroLucida, one works on a coarse set of points with straight lines between them, but the VR tool provides smooth lines. To this end, a spline-based manipulation system could work well, but may be unintuitive for novices.

Supporting multiple users in the same virtual environment, either locally or over a network, would be useful for facilitating collaborative work and training sessions. For example, it was difficult for two users to discuss complex crossings or stitching issues, with one wearing the HMD and the other looking at a mirrored view on a desktop monitor. The separation of the users hampered discussions between the two, as the user viewing the desktop could not point to features viewed by the other in VR.

A combined rendering mode [23], potentially with shadows and ambient occlusion built in, could help users by presenting both modes simultaneously, along with stronger depth cues. Providing more familiar representations to the experts could also encourage neuroanatomists to adopt the tool, such as adding the option to view the original images of individual sections. However, in principle, a single well-interpretable rendering modality would be preferable to repeatedly bringing up 2D images.

Although our paging system can handle terabyte-sized data produced by high-resolution microscopes, the small, highly zoomed-in focus region hinders a global view of the data, thereby potentially hampering understanding of the data and productivity. Improving rendering performance would enable us to increase the resolution of the focus region, and adding a zoom option or coarse resolution view would allow users to obtain an overview of the data. Finally, the addition of semi-automated methods for extracting neuron structure would greatly accelerate reconstructions, especially in large datasets, by allowing users to quickly resolve easier cases. We are actively working on integrating a semi-automatic guided method, e.g., using Voxel Scooping [41] and Rayburst sampling [40], to extract neuronal structures and their radii.

The small scope of our pilot study was appropriate due to the domain-specific nature of our system and the familiarity of our collaborators with the problem of manual neuron tracing. However, it would be useful to broaden our study and examine the impact of VR on other problems in neural imaging, such as multi-channel data [44] or data employing automatic or semi-automatic registration and segmentation techniques [21]. To this end, we have released our software open-source and are working to expand deployment to other labs.

7 Conclusion

We have presented a design study to develop a virtual reality tool for neuron tracing, conducted through a close collaboration between computer scientists and neuroscientists. We have established that the resulting tool is effective for neuron tracing. On average, users are as accurate and faster at neuron tracing using our VR tool as they are using the current industry standard tool, and can trace orders of magnitude larger datasets via the integrated paging system. Moreover, users find the VR tool easier to interact with, and less fatiguing. Although we did not rigorously explore this aspect, use of our tool does not require tiring actions such as standing or keeping one's arms raised. In fact, users reported feeling equally productive while seated as when standing. Overall, recent consumer-grade VR systems like the HTC Vive are affordable and provide a sufficiently high-quality VR experience to be used as a standard tool in scientific data analysis and visualization. Although not all analysis tasks may be well suited to VR, those involving understanding complex 3D structures or interacting directly with 3D data, like neuron tracing, can be aided by VR-based tools.

The features we added to the tool proved useful to experts. For example, one such added feature, the replaying system, specifically aided users in joint discussions, as well as in identifying and understanding the causes of tracing mistakes. Through our evaluation and discussions with users, we have identified several potential improvements to the tool that could facilitate identification of common mistakes and aid in understanding of the data, as well as further reducing tracing time.

Acknowledgments

We would like to thank Anders Ynnerman, Yarden Livnat, Amy Gooch, the anonymous evaluation users, and the reviewers. This work is supported in part by NSF:CGV Award: 1314896, NSF:IIP Award: 1602127, NSF:ACI Award: 1649923, DoE/SciDAC DESC0007446, CCMSC DE-NA0002375, PIPER: ER26142 DE-SC0010498, the NIH (NEI R01 EY019743 and R01 EY026812, NINDS BRAIN grant U01 NS099702), the NSF (IOS 1355075 and EAGER 1649923), the U. of Utah Research Foundation (seed grant 10040877), the U. of Utah Neuroscience Initiative, and a grant from Research to Prevent Blindness to the Dept. of Ophthalmology. This material is based upon work supported by the DoE, NNSA, under Award Number DE-NA0002375.

REFERENCES

[1] Cell type-specific structural plasticity of axonal branches and boutons in the adult neocortex. *Neuron*, 49(6):861–875.

[2] Oculus Rift Development Kit 2. www.oculus.com/en-us/dk2/, July 2014.

[3] HTC Vive. www.vive.com/us/product/, April 2016.

[4] L. Acciai, P. Soda, and G. Iannello. Automated Neuron Tracing Methods: An Updated Account. *Neuroinformatics*, 14(4):353–367, Oct. 2016. doi: 10.1007/s12021-016-9310-0

[5] G. A. Ascoli. Neuroinformatics Grand Challenges. *Neuroinformatics*, 6(1):1–3, Mar. 2008. doi: 10.1007/s12021-008-9010-5

[6] H. C. Baker, Jr. and C. Hewitt. The incremental garbage collection of processes. In *Proceedings of the 1977 Symposium on Artificial Intelligence and Programming Languages*, pp. 55–59. ACM, New York, NY, USA, 1977. doi: 10.1145/800228.806932

[7] J. Beyer, M. Hadwiger, A. Al-Awami, W.-K. Jeong, N. Kasthuri, J. W. Lichtman, and H. Pfister. Exploring the connectome: Petascale volume visualization of microscopy data streams. *IEEE Computer Graphics and Applications*, 33(4):50–61, 2013.

[8] D. D. Bock, W.-C. A. Lee, A. M. Kerlin, M. L. Andermann, G. Hood, A. W. Wetzel, S. Yurgenson, E. R. Soucy, H. S. Kim, and R. C. Reid. Network anatomy and in vivo physiology of visual cortical neurons. *Nature*, 471(7337):177–182, 2011.

[9] A. Bria, G. Iannello, and H. Peng. An open-source Vaa3D plugin for real-time 3D visualization of terabyte-sized volumetric images. In *Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on*, pp. 520–523. IEEE, 2015.

[10] K. L. Briggman, M. Helmstaedter, and W. Denk. Wiring specificity in the direction-selectivity circuit of the retina. *Nature*, 471(7337):183–188, 2011.

[11] K. Chung, J. Wallace, S.-Y. Kim, S. Kalyanasundaram, A. S. Andalman, T. J. Davidson, J. J. Mirzabekov, K. A. Zalocusky, J. Mattis, A. K. Denisin, et al. Structural and molecular interrogation of intact biological systems. *Nature*, 497(7449):332–337, 2013.

[12] C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon, and J. C. Hart. The CAVE: audio visual experience automatic virtual environment. *Communications of the ACM*, 35(6):64–73, 1992.

[13] W. Denk, J. Strickler, and W. Webb. Two-photon laser scanning fluorescence microscopy. *Science*, 248(4951):73–76, 1990. doi: 10.1126/science.2321027

[14] A. Febretti, A. Nishimoto, T. Thigpen, J. Talandis, L. Long, J. D. Pirtle, T. Peterka, A. Verlo, M. Brown, D. Plepys, D. Sandin, L. Renambot, A. Johnson, and J. Leigh. CAVE2: a hybrid reality environment for immersive simulation and information analysis. vol. 8649, pp. 864903–864903–12, 2013. doi: 10.1117/12.2005484

[15] T. A. Gillette, K. M. Brown, and G. A. Ascoli. The DIADEM Metric: Comparing Multiple Reconstructions of the Same Neuron.

[16] E. M. Glaser and H. Van der Loos. A semi-automatic computer-microscope for the analysis of neuronal morphology. *IEEE Transactions on Biomedical Engineering*, (1):22–31, 1965.

[17] Google. Tilt Brush, 2016.

[18] M. Hadwiger, P. Ljung, C. R. Salama, and T. Ropinski. Advanced illumination techniques for GPU volume raycasting. In *ACM Siggraph Asia 2008 Courses*, p. 1. ACM, 2008.

[19] F. Hernell, P. Ljung, and A. Ynnerman. Local ambient occlusion in direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 16:548–559, 2009. doi: 10.1109/TVCG.2009.45

[20] M. Ikits and D. Brederson. The Visual Haptic Workbench. *The Visualization Handbook*, pp. 431–449, 2005.

[21] W.-K. Jeong, J. Beyer, M. Hadwiger, R. Blue, C. Law, A. Vázquez-Reina, R. C. Reid, J. Lichtman, and H. Pfister. SSECRETT and NeuroTrace: Interactive visualization and analysis tools for large-scale neuroscience data sets. *IEEE Computer Graphics and Applications*, 30(3):58–70, 2010.

[22] D. Jönsson, E. Sundén, A. Ynnerman, and T. Ropinski. A Survey of Volumetric Illumination Techniques for Interactive Volume Rendering. *Computer Graphics Forum*, 33(1):27–51, 2014. doi: 10.1111/cgf.12252

[23] A. Knoll, Y. Hijazi, R. Westerteiger, M. Schott, C. Hansen, and H. Hagen. Volume ray casting with peak finding and differential sampling. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1571–1578, Nov 2009. doi: 10.1109/TVCG.2009.204

[24] B. Laha, D. A. Bowman, and J. J. Socha. Effects of VR system fidelity on analyzing isosurface visualization of volume datasets. *IEEE Transactions on Visualization and Computer Graphics*, 20(4):513–522, 2014.

[25] B. Laha, K. Sensharma, J. D. Schiffbauer, and D. A. Bowman. Effects of immersion on visual analysis of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):597–606, 2012.

[26] Y. Liu. The DIADEM and Beyond. *Neuroinformatics*, 9(2-3):99–102, Sept. 2011. doi: 10.1007/s12021-011-9102-5

[27] L. Luo, E. M. Callaway, and K. Svoboda. Genetic dissection of neural circuits. *Neuron*, 57(5):634 – 660, 2008. doi: 10.1016/j.neuron.2008.01.002

[28] K.-L. Ma, J. S. Painter, C. D. Hansen, and M. F. Krogh. Parallel volume rendering using binary-swap compositing. *IEEE Comput. Graph. Appl.*, 14(4):59–68, July 1994. doi: 10.1109/38.291532

[29] MBF Bioscience. NeuroLucida 11.08.

[30] MBF Bioscience. NeuroLucida 360.

[31] E. Meijering. Neuron tracing in perspective. *Cytometry Part A*, 77A(7):693–704, Mar. 2010. doi: 10.1002/cyto.a.20895

[32] E. Murray, J. Cho, D. Goodwin, T. Ku, J. Swaney, S.-Y. Kim, H. Choi, Y.-G. Park, J.-Y. Park, A. Hubbert, M. McCue, S. Vassallo, N. Bakh, M. Frosch, V. Wedeen, H. Seung, and K. Chung. Simple, scalable proteomic imaging for high-dimensional profiling of intact systems. *Cell*, 163(6):1500 – 1514, 2015. doi: 10.1016/j.cell.2015.11.025

[33] K. Palmerius, M. Cooper, and A. Ynnerman. Haptic Rendering of Dynamic Volumetric Data. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):263–276, Mar. 2008. doi: 10.1109/TVCG.2007.70409

[34] V. Pascucci and R. J. Frank. Global static indexing for real-time exploration of very large regular grids. In *Proceedings of the 2001 ACM/IEEE Conference on Supercomputing*, SC '01. ACM, 2001. doi: 10.1145/582034.582036

[35] H. Peng, F. Long, T. Zhao, and E. Myers. Proof-editing is the Bottleneck of 3D Neuron Reconstruction: The Problem and Solutions. *Neuroinformatics*, 9(2-3):103–105, Sept. 2011. doi: 10.1007/s12021-010-9090-x

[36] H. Peng, Z. Ruan, F. Long, J. H. Simpson, and E. W. Myers. V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. *Nature Biotechnology*, 28(4):348–353, 2010.

[37] H. Peng, J. Tang, H. Xiao, A. Bria, J. Zhou, V. Butler, Z. Zhou, P. T. Gonzalez-Bellido, S. W. Oh, J. Chen, A. Mitra, R. W. Tsien, H. Zeng, G. A. Ascoli, G. Iannello, M. Hawrylycz, E. Myers, and F. Long. Virtual finger boosts three-dimensional imaging and microsurgery as well as terabyte volume image visualization and analysis. *Nature Communications*, 5, July 2014. doi: 10.1038/ncomms5342

[38] Prabhat, A. Forsberg, M. Katzourin, K. Wharton, and M. Slater. A Comparative Study of Desktop, Fishtank, and Cave Systems for the Exploration of Volume Rendered Confocal Data Sets. *IEEE Transactions on Visualization and Computer Graphics*, 14(3):551–563, May 2008. doi: 10.1109/TVCG.2007.70433

[39] K. Reda, A. Knoll, K.-i. Nomura, M. E. Papka, A. E. Johnson, and J. Leigh. Visualizing large-scale atomistic simulations in ultra-resolution immersive environments. In *LDAV*, pp. 59–65, 2013.

[40] A. Rodriguez, D. Ehlenberger, P. Hof, and S. Wearne. Rayburst sampling, an algorithm for automated three-dimensional shape analysis from laser scanning microscopy images. *Nature Protocols*, 2006.

[41] A. Rodriguez, D. Ehlenberger, P. Hof, and S. Wearne. Three-Dimensional Neuron Tracing by Voxel Scooping. *Journal of Neuroscience Methods*, 2009.

[42] M. Sedlmair, M. Meyer, and T. Munzner. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440, 2012.

[43] A. Vlachos. Advanced VR Rendering. GDC, 2015.

[44] Y. Wan, H. Otsuna, C.-B. Chien, and C. Hansen. FluoRender: an application of 2D image space methods for 3D and 4D confocal microscopy data visualization in neurobiology research. In *Pacific Visualization Symposium*, pp. 201–208. IEEE, 2012.

[45] J. G. White, E. Southgate, J. N. Thomson, and S. Brenner. The Structure of the Nervous System of the Nematode Caenorhabditis elegans. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 314(1165):1–340, 1986. doi: 10.1098/rstb.1986.0056

[46] B. Yang, J. B. Treweek, R. P. Kulkarni, B. E. Deverman, C.-K. Chen, E. Lubeck, S. Shah, L. Cai, and V. Gradinaru. Single-cell phenotyping within transparent intact tissue through whole-body clearing. *Cell*, 158(4):945–958, 2014.

[47] A. Ynnerman, T. Rydell, A. Persson, A. Ernvik, C. Forsell, P. Ljung, and C. Lundstrm. Multi-Touch Table System for Medical Visualization. In H.-C. Hege and T. Ropinski, eds., *EG 2015 - Dirk Bartz Prize*. The Eurographics Association, 2015. doi: 10.2312/egm.20151030