Eurographics Workshop on Visual Computing for Biology and Medicine (2020)
K. Nieselt and R. G. Raidou (Editors)

The Virtual Reality Flow Lens for Blood Flow Exploration

B. Behrendt[1], L. Piotrowski[1], S. Saalfeld[1,2], B. Preim[1,2], and P. Saalfeld[1]

[1]Institute for Simulation and Graphics, University of Magdeburg, Germany
[2]Research Campus STIMULATE

Abstract

The exploration of time-dependent measured or simulated blood flow is challenging due to the complex three-dimensional structure of vessels and blood flow patterns. Especially on a 2D screen, understanding their full shape and interacting with them is difficult. Critical regions do not always stand out in the visualization and may easily be missed without proper interaction and filtering techniques. The FlowLens [GNBP11] was introduced as a focus-and-context technique to explore one specific blood flow parameter in the context of other parameters for the purpose of treatment planning. With the recent availability of affordable VR glasses it is possible to adapt the concepts of the FlowLens into immersive VR and make them available to a broader group of users. Translating the concept of the Flow Lens to VR leads to a number of design decisions not only based around what functions to include, but also how they can be made available to the user. In this paper, we present a configurable focus-and-context visualization for the use with virtual reality headsets and controllers that allows users to freely explore blood flow data within a VR environment. The advantage of such a solution is the improved perception of the complex spatial structures that results from being surrounded by them instead of observing through a small screen.

CCS Concepts

• ***Human-centered computing*** → ***Interaction techniques;*** *Scientific visualization; Visualization systems and tools;* • ***Computing methodologies*** → ***Virtual reality;***

1. Introduction

A common way to visualize blood flow data is in the form of pathlines representing the flow, embedded into a 3D vascular model. Hemodynamic or morphological parameters, such flow velocity, surface curvature or wall shear stress, can be mapped to both the pathlines as well as the surface using color scales. However, such a visualization often suffers from visual clutter and occlusion, especially due to mapping complex, three-dimensional structures onto a two-dimensional screen. Filtering techniques can be used to reduce these problems. Removing or de-emphasizing *uninteresting* structures allows the user to focus on those that are important. However, the definition of *importance* is not trivial, especially in an explorative context where the user may not even know what she is looking for.

The use of VR is becoming more and more widespread, being not only used in games, but also in medical applications or anatomy education [Sat95]. VR technology fully exploits the spatial perception and awareness of humans and is well suited for creating audio-visual immersion, especially for training simulations. Additionally, interactions with the virtual environment feel more natural, as they take place in three dimensions instead of just two and they can make use of natural gestures instead of artificial mouse movements.

In this paper, we discuss design choices and implementation details to translate the concepts of a Flow Lens to immersive VR. We employ the *magic lens* approach where filter geometries are used to provide a *steerable* focus-and-context visualization. Combining the excellent spatial perception and 3D interaction capabilities of VR with a lens-based focus-and-context visualization seems like a natural approach. We demonstrate these concepts on the example of simulated cerebral blood flow data, focussing on two pathologies – intracranial aneurysms and arteriovenous malformations. To the best of our knowledge, this is the first attempt to filter 3D meshes with a volumetric object in VR.

2. Related Work

VR in Scientfic Visualization. Since the pioneering work on the Virtual Windtunnel from Steve Bryson [BL91] there have been various attempts to employ immersive VR to explore complex scientific datasets, as they may arise from simulations or measurements. A general project report from van Dam et al. summarized early prototypes and concepts to be explored in the future [vFL*00]. In a follow-up article, van Dam et al. [vLS02] provide an overview of immersive VR for scientific visualization, arguing that VR has great potential to enable the exploration of *growing* datasets. Yu et al. [YSI*10] present a touch-based exploration technique for

3D data aiming at an improved navigation for scientific visualization applications, such as particle simulations. Recently, Johnson et al. [JOR*19] provides a VR environment to explore 4D simulation data.

VR in Medical and Flow Visualization. There have been only few attempts to employ immersive VR for medical flow visualization. Most research has been carried out to employ VR for surgical training, such as the work by Seymour et al. [SGR*02], where it serves to acquire skills. Few papers addressed essential medical visualization tasks, such as the exploration of DT-MRI fiber tracts [ZDK*01], virtual endoscopy [MGO*19], the exploration of brain data along with textual annotations [PHTP*10] and immersive visualization of medical flow through an artery [FLv*00]. Forsberg et al. [FLv*00] present an explorative VR-based solution where the user is placed inside the vessel geometry and has multiple tools at her disposal, such as throwing dust which follows the blood flow, seeding streamlines at a specific location or creating iso-surfaces. The development of a VR exploration for virtual colonoscopy [MGO*19], which also deals with elongated structures similar to blood vessels, was motivated by previous work on the influence of the field of view on the diagnostic performance.

Shi et al. presented *Harvis* [SAR20], a flexible software platform that allows the exploration of flow simulation results. An interesting aspect of this platform is the variety of systems that it can be used on ranging from standard 2D display, over semi-immersive stereoscopic displays to fully immersive VR headsets.

Lens-Based Interaction. Magic lenses were introduced by Bier et al. [BSP*93]. They are not used to magnify information, but instead dynamically change what is displayed within the lens region. This concept was extended to 3D data (*Magic Spheres*) and served to explore surface and volume data [CMS94]. Viega et al. [VCWP96] later extended this to fit a more general approach. Inspired by these developments, Gasteiger et al. [GNBP11] introduced the FlowLens for exploring hemodynamic simulation results, showing focus-and-context pairs of hemodynamic attributes for the assessment of aneurysms. Fuhrmann et al. [FG98] used magic lenses and boxes to provide a focus-and-context visualization of flow streamlines. For VR, Borst et al. [BTB10] developed a method to achieve real-time performance for composable volumetric lenses suitable for VR. A survey on lens-based interaction summarizes these and many other developments [TGK*17].

3. Concept of a VR Flow Lens

The core concept of the Flow Lens by Gasteiger et al. [GNBP11] was to blend between different preset visualizations at user-specified locations in screen space. We translate this concept into VR by using volumetric Lens Objects (*LO*) in scene space that apply customizable filters to geometry they intersect. In the following, we will outline the filtering step (Section 3.1), our design choices for the user interface (Section 3.2) and scene interactions (Section 3.3).

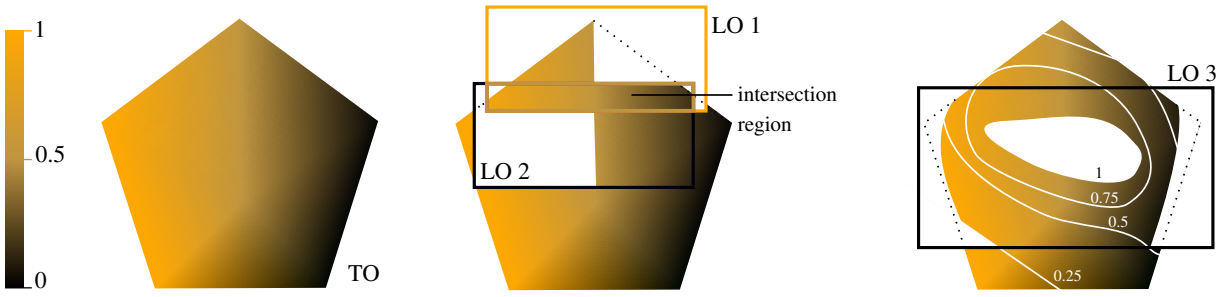
3.1. Filtering

Our filtering algorithm uses 3D objects as Lens Objects. Their associated filter criterion is applied to geometry intersecting their volume. We refer to the geometry that is subject to filtering, e.g. pathlines and vessel surfaces, as target objects (*TO*). These TOs can have a multitude of attributes mapped to their vertices, such as pressure or flow velocity. If regions of the intersecting part of a TO do not fulfil the filter criterion, they are clipped from the visualization. As this is a *cut/keep* calculation, we do not fade any fragments or otherwise alter their appearance.

In the first rendering step, only the LOs are rendered. A fragment shader captures the depth of their front- and backfaces for each screen pixel and stores them in a buffer on the GPU. Thus, the buffer needs to be large enough to capture two depth values for each existing LO and pixel. In comparison to more advanced methods, such as dynamic linked lists, our approach consumes more graphics memory, but offers better performance.

When the TOs are rendered, their fragment shader has access to a list of depth ranges (from the Filter Depth Buffer) and associated filter definitions (from the Filter Entry Buffer). Therefore, the shader can decide for each fragment if it needs to be removed. Fragments that do not fall in the depth range of any LO are always shown. Attributes are mapped to the TOs using the UV-channels of their mesh. They can be easily accessed and are automatically interpolated between vertices by the fragment shader. To simplify interaction and retain an interactive frame rate, we implemented some restrictions on how different filter attributes and geometries can be combined. Each LO is associated to a single TO (such as the vessel surface or pathlines) and will ignore other objects that intersect its volume. This significantly simplifies user interaction required to adjust the filter criteria, as the filter no longer has to handle multiple objects with different parameter sets and parameter value ranges. The filter criterion for a single LO can include both ranges for multiple attributes and multiple ranges for a single attribute. In this case, these atomic conditions are combined with a logical **AND** to determine if a specific fragment will be rendered (Fig. 1c). It is also possible to define multiple LOs for a single TO. In regions where multiple LOs overlap, their individual filter criteria are combined with a logical **OR** to decide if a fragment is to be shown. This allows the user to easily create complex filter criteria by choosing multiple overlapping LOs with different range or parameter configurations.

Performance is crucial in VR applications. A frame rate of 60 is widely considered a minimum requirement for head-mounted displays, however 90 FPS are usually targeted to avoid an uncomfortable feeling and motion sickness. Thus the shader design needs to be optimized to achieve sufficient frame rates. The filter algorithm within the shader is illustrated in Algorithm 1. For each fragment and LO, the per-pixel depth values are read from the depth buffer to determine if the fragment is located within the LOs volume. If this is the case, all filter entries associated to the specific LO are extracted from the filter entry buffer. For each entry, we compare the attribute value of the fragment with the range defined by the filter entry. If any attribute falls out of range, the LO *votes* for the fragment to be discarded. This process repeats for all LOs and models the `AND` relation between the attributes that are on the same LO. The fragment is only actually removed if each of the applicable LOs *votes* for it to be discarded, modeling the `OR` relation between different LOs.



(a) *Before filtering* **(b)** *Two LOs operating on the same attribute* **(c)** *LO operating on 2 attributes*

Figure 1: *Example representations of the restrictions and relations for Lens Objects and Target Objects. (a) TO with an attribute shown as gradient. (b) LO 1 shows range [0.5, 1], LO 2 shows range [0, 0.5]. As multiple LOs are connected with an* ***or****-relation, the intersecting region shows a range of* $[0,1](=[0,0.5]\cup[0.5,1])$*. (c) LO 3 filters for two ranges of different attributes (first shown as gradient, second as isolines). Only regions in range* $[0.1,0.9]$ *of the gradient attribute* ***and*** $[0.25,0.99]$ *of the isoline attribute are shown.*

3.2. User Interface

User interfaces in VR have different requirements and challenges compared to those on desktop applications. For example, the UI needs to exist as an actual object in the virtual world instead of being layered on top of the scene. The user interface should be easily usable in VR and to provide access to settings for LO and TO. In this section, we discuss design challenges and how we approached them as well as interaction with the user interface.

Algorithm 1 Filtering algorithm within the fragment shader.

```
f ← fragment
B_e ← filter entry buffer
B_d ← filter depth pixel buffer

function FILTER(LO ← LensObject)
    for all entry in B_e do
        if ID( entry ) = ID( LO ) then
            a ← ATTRIB( f, entry_attrib )
            if not INBETWEEN ( a, entry_min, entry_max ) then
                return true                    ▷ Vote to discard f
            end if
        end if
    end for
    return false        ▷ Vote to keep f if all attrib ranges match
end function

dsc ← 0
for all LO in B_d do
    if INBETWEEN ( f_depth, LO_front, LO_back ) then
        if not FILTER (entry) then
            return keep          ▷ Keep f if any LO votes to keep
        else
            dsc++
        end if
    end if
end for
if dsc > 0 then
    return discard           ▷ Discard f if no LO votes to keep
else
    return keep              ▷ Keep f if no LO was applicable
end if
```

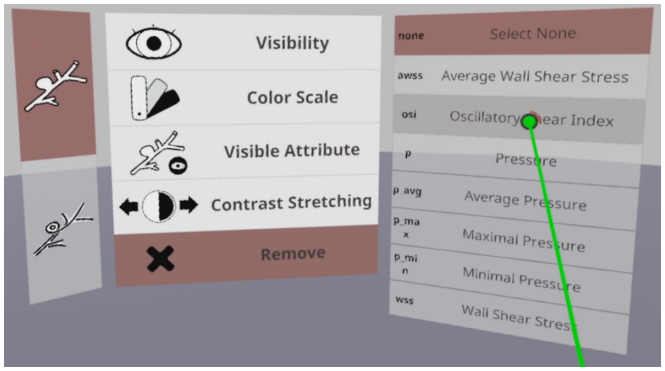


Figure 2: *Implementation of the UI concept in VR*

3.2.1. Design Choices

As mouse and keyboard are not available within the virtual environment, the primary input method are motion controllers that work together with the headset. We decided to use a ray cast-based interaction represented with a laser pointer that allows grabbing near and far objects. Subtle presses to the trigger button preview the pointer, while applying more force performs the actual selection action. The user interface is mounted to the virtual representation of the motion controllers. Thus, it is at most one arm length away and can easily be moved in and out of view by the user or brought closer to the camera.

Despite the comparatively high resolution of common VR headsets, such as the HTC Vive with `1,080 × 1,200` pixels per eye, the low distance between eye and display causes individual pixels to be easily perceivable for the user. Thus, we used a large font

with a clear typeface to guarantee legibility for the labels. Wherever possible, descriptive icons are used either in addition or in place of text to convey information. To prevent erroneous clicks due to the aforementioned lack of pointer precision, we implemented a *hold to click* mechanic for buttons. Section 3.2.2 discusses the layout of the UI in detail.

A special interaction is changing the range for attributes. To adjust contrast in grey-value medical images, such as X-ray images, *windowing* is used. The transfer function to convert these grey-values to actual pixel brightness is a ramp function, defined by its mid-point (the *window center*) and width. Adjusting these two values is carried out in radiological workstations using horizontal and vertical mouse movement, respectively. To allow for an effective way to configure filter value ranges, we decided to use the touchpad of the HTC Vive controller to model the same interaction in VR. Movement on the horizontal axis moves the value range from left to right and movement on the vertical axis scales the range's size. A preview of the currently selected range is shown on the UI.

3.2.2. Structure of the UI

The UI is structured into multiple context-sensitive panels, which are horizontally divided into three parts each. The central part houses the main actions, such as the option to add or remove objects or to apply a color scale (Fig. 2). The right part contains additional actions and settings in relation to the central part, such as the attribute selection or value range control for the color mapping. The left part is a narrow vertical list of icon buttons used for navigating between categories of actions.

3.3. Scene Objects

This section describes how the LOs and different types of TOs are rendered. Within the scope of this section, the term *object* refers to both LOs and TOs, but not other visible elements within the scene, such as the UI or controller representations.

3.3.1. Presentation

The rendering of the TO is carried out differently with and without a color scale. Without a color scale, the front faces are rendered semi-transparently and the back-faces fully opaque. The transparency is modulated using a Fresnel effect, similar to the work of Gasteiger et al. [GNKP10]. If the user chooses an attribute to be color-encoded, the front-faces are drawn fully opaque and a simple Phong lighting is used to convey shape. The user can choose between different color scales for the attributes. Pathlines are rendered as simplified tubes with a triangular cross section (Fig. 3a and 3b). They are always drawn fully opaque, regardless of whether an attribute is mapped to them.

LOs are designed to always be visible but not intrusive, as the user needs to be aware of their position and be able to grab them. Therefore, both their front- and back-faces are rendered transparently, with the transparency once again modulated by the Fresnel effect (Fig. 3a and 3b). Figure 3 shows the application of nested filters. A temperature color scale has been applied to the vessel surface to represent pressure, and a rainbow scale to the pathlines representing velocity. One LO is used to cull the vessel surface to reveal the inner blood flow around an aneurysm (Fig. 3a). To highlight the inflow yet of the blood into the aneurysm, a second LO is created to reveal pathlines with high velocity (Fig. 3b).

3.3.2. Interaction

All TOs and LOs can be grabbed and then moved, rotated and scaled. The user can grab an object by either moving the virtual controller inside the object or directing the laser pointer at the object and pressing the dedicated grab buttons on the HTC Vive controller. Initially, a simple raycasting interaction was implemented, where the object is attached to the hit point of the laser beam. The center of interaction is still the controller's position, which could lead to large rotational changes if the object is far away due to the lever-arm effect. Additionally, the user has to grab and drag an object repeatedly to get it into close range. Instead, we used the *HOMER* technique (hand-centred object manipulation extending raycasting) [BH97]. Here, the rotation of the object follows directly the orientation of the controller. The translation, however, is scaled by a factor that is defined by the distance from the user's hand to her torso. This makes it easy to get objects close to the user quickly and still have control over rotations. This interaction ends once the grab button is released. Uniformly scaling an object requires the user to grab the object with both controllers or laser pointers. Then, moving both controllers closer together scales the object down and moving them apart from each other enlarges the object.

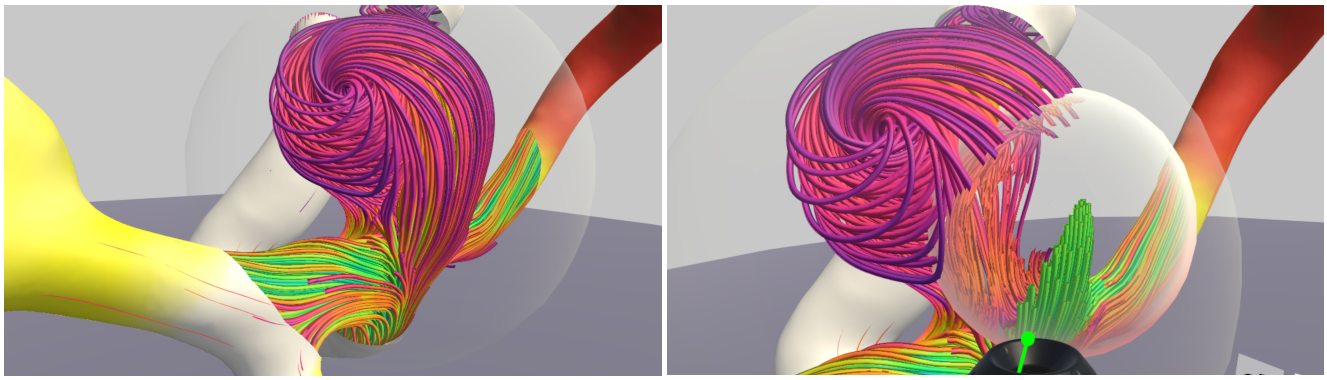
4. Conclusions

Magic lenses are a widely used method to facilitate focus-and-context visualizations. Using two-dimensional lenses positioned in screen space comes with significant draw-backs when being applied to complex, three-dimensional geometry. However the lack of spatial perception makes placing three-dimensional lenses challenging as well. Thus, we proposed a prototype for using volumetric filtering geometry as magic lenses in VR to support the exploration of blood flow data. The complex spatial structure of vasculature along with the time-dependent flow information comprising various attributes particularly motivates the use of VR. We discussed the concept and restrictions of the filtering task and highlighted challenges in designing a VR application around it. This applies especially to the user interface as well as the interaction with scene objects.

Our VR prototype is capable of performing the filtering task in real-time. As common use cases would require many LO simultaneously, the main bottleneck regarding performance is the complexity of the examined objects. VR Flow Lenses are a viable concept and can support the exploration of blood flow data. However, further improvements need to be carried out to use this concept in domains such as medical treatment planning, diagnosis or education.

Acknowledgements

We would like to give thanks to Samuel Voß and Dr.-Ing. Philipp Berg for their valuable feedback. Additionally, we thank Naoki Kaneko, MD, PhD (UCLA Interventional Neuroradiology, Los Angeles, USA) for providing the AVM dataset.



(a) (b)

Figure 3: *Nested filters; First, a surface LO attached to the vessel (a). A second pathline LO used to filter pathlines by velocity to visualize the inflow jet (b).*

References

[BH97] BOWMAN D. A., HODGES L. F.: An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proc. of Interactive 3D Graphics* (1997), p. 35–38. doi:10.1145/253284.253301. 4

[BL91] BRYSON S., LEVIT C.: The virtual windtunnel: An environment for the exploration of three-dimensional unsteady flows. In *Proc. of IEEE Visualization* (1991), pp. 17–24. 1

[BSP*93] BIER E. A., STONE M. C., PIER K., BUXTON W., DEROSE T. D.: Toolglass and magic lenses. In *Proc. of SIGGRAPH* (1993), pp. 73–80. doi:10.1145/166117.166126. 2

[BTB10] BORST C. W., TIESEL J.-P., BEST C. M.: Real-time rendering method and performance evaluation of composable 3d lenses for interactive vr. *IEEE transactions on visualization and computer graphics 16*, 3 (2010), 394–406. doi:10.1109/TVCG.2009.89. 2

[CMS94] CIGNONI P., MONTANI C., SCOPIGNO R.: Magicsphere: an insight tool for 3d data visualization. *Computer Graphics Forum 13*, 3 (1994), 317–328. doi:10.1111/1467-8659.1330317. 2

[FG98] FUHRMANN A., GROLLER E.: Real-time techniques for 3d flow visualization. In *Proc. of IEEE Visualization* (1998), pp. 305–312. doi:10.1109/VISUAL.1998.745317. 2

[FLv*00] FORSBERG A. S., LAIDLAW D. H., VAN DAM A., KIRBY R. M., KAFNIADAKIS G. E., ELION J. L.: Immersive virtual reality for visualizing flow through an artery. In *Proc. of IEEE Visualization* (2000), pp. 457–460. doi:10.1109/VISUAL.2000.885731. 2

[GNBP11] GASTEIGER R., NEUGEBAUER M., BEUING O., PREIM B.: The flowlens: A focus-and-context visualization approach for exploration of blood flow in cerebral aneurysms. *IEEE transactions on visualization and computer graphics 17*, 12 (2011), 2183–2192. doi:10.1109/TVCG.2011.243. 1, 2

[GNKP10] GASTEIGER R., NEUGEBAUER M., KUBISCH C., PREIM B.: Adapted surface visualization of cerebral aneurysms with embedded blood flow information. In *VCBM* (2010), pp. 25–32. 4

[JOR*19] JOHNSON S., ORBAN D., RUNESHA H. B., MENG L., JUHNKE B., ERDMAN A., SAMSEL F., KEEFE D. F.: Bento box: An interactive and zoomable small multiples technique for visualizing 4d simulation ensembles in virtual reality. *Frontiers in Robotics and AI 6* (2019). doi:10.3389/frobt.2019.00061. 2

[MGO*19] MIRHOSSEINI S., GUTENKO I., OJAL S., MARINO J., KAUFMAN A.: Immersive virtual colonoscopy. *IEEE transactions on visualization and computer graphics 25*, 5 (2019), 2011–2021. doi:10.1109/TVCG.2019.2898763. 2

[PHTP*10] PICK S., HENTSCHEL B., TEDJO-PALCZYNSKI I., WOLTER M., KUHLEN T.: Automated positioning of annotations in immersive virtual environments. In *Proc. of Virtual Environments & Second Joint Virtual Reality* (2010), Eurographics Association, p. 1–8. doi:10.2312/EGVE/JVRC10/001-008. 2

[SAR20] SHI H., AMES J., RANDLES A.: Harvis: an interactive virtual reality tool for hemodynamic modification and simulation. *Journal of Computational Science 43* (2020). 2

[Sat95] SATAVA R. M.: Medical applications of virtual reality. *Journal of medical systems 19*, 3 (1995), 275–280. doi:10.1007/BF02257178. 1

[SGR*02] SEYMOUR N. E., GALLAGHER A. G., ROMAN S. A., O'BRIEN M. K., BANSAL V. K., ANDERSEN D. K., SATAVA R. M.: Virtual reality training improves operating room performance: results of a randomized, double-blinded study. *Annals of surgery 236*, 4 (2002), 458–63; discussion 463–4. doi:10.1097/00000658-200210000-00008. 2

[TGK*17] TOMINSKI C., GLADISCH S., KISTER U., DACHSELT R., SCHUMANN H.: Interactive lenses for visualization: An extended survey. *Computer Graphics Forum 36*, 6 (2017), 173–200. doi:10.1111/cgf.12871. 2

[VCWP96] VIEGA J., CONWAY M. J., WILLIAMS G., PAUSCH R.: 3d magic lenses. In *Proc. of the 9th annual ACM symposium on User interface software and technology - UIST '96* (1996), pp. 51–58. doi:10.1145/237091.237098. 2

[vFL*00] VAN DAM A., FORSBERG A. S., LAIDLAW D. H., LAVIOLA J. J., SIMPSON R. M.: Immersive vr for scientific visualization: a progress report. *IEEE Computer Graphics and Applications 20*, 6 (2000), 26–52. doi:10.1109/38.888006. 1

[vLS02] VAN DAM A., LAIDLAW D. H., SIMPSON R. M.: Experiments in immersive virtual reality for scientific visualization. *Computers & Graphics 26*, 4 (2002), 535–555. doi:10.1016/S0097-8493(02)00113-9. 1

[YSI*10] YU L., SVETACHOV P., ISENBERG P., EVERTS M. H., ISENBERG T.: Fi3d: direct-touch interaction for the exploration of 3d scientific visualization spaces. *IEEE transactions on visualization and computer graphics 16*, 6 (2010), 1613–1622. doi:10.1109/TVCG.2010.157. 1

[ZDK*01] ZHANG S., DEMIRALP C., KEEFE D. F., DASILVA M., LAIDLAW D. H., GREENBERG B. D., BASSER P. J., PIERPAOLI C., CHIOCCA E. A., DEISBOECK T. S.: An immersive virtual environment for dt-mri volume visualization applications: a case study. In *Proc. of IEEE Visualization* (2001), pp. 437–584. doi:10.1109/VISUAL.2001.964545. 2