Medical Image Atlas Interaction in Virtual Reality

Lindun He[*] Alejandro Guayaquil-Sosa[†] Tim McGraw[‡]

Purdue University

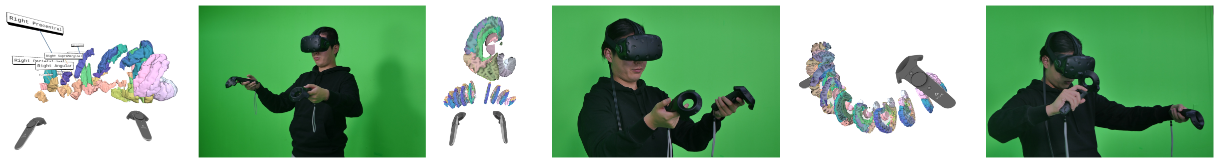


Figure 1: Exploded views demonstrating linear explosion, leafing, fanning and the tracked-controller gestures which generated them.

Abstract

Medical image atlases provide a wealth of information about the anatomy of organisms, but understanding the shapes of regions and the overall hierarchical structure of an atlas can be difficult. Atlases can contain hundreds of regions with complex shapes. The way those shapes fit together is difficult to visualize because they are usually packed tightly together and occlude each other. In this work we describe a technique which enables interactive exploration of medical image atlases and an underlying medical image by creating exploded views similar to those used in technical illustrations to enhance the visualization of mechanical assemblies. A pair of motion tracked controllers allow the user to intuitively control the process of exploded view generation.

Index Terms: J.3 [Computer Applications]: Life and Medical Sciences—Health; I.3.6 [Computing Methodologies]: Computer Graphics—Methodology and Techniques

1 Introduction

A medical image 'atlas' is an image which has had tissue or structures labeled by an expert. Atlases may be constructed by aligning multiple images of the same modality, e.g. MRI, into a common coordinate system and statistically analyzing them to determine the most representative image to obtain a result which is independent of individual anatomical variations. This 'template' image can be manually labeled to complete the digital atlas. Uses include surgical planning, neuroanatomy education, and assisting in medical image analysis tasks such as image segmentation.

A trivial approach to visualizing an atlas is to color-code the labeled regions and overlay them on the template image, as shown in Figure (2). Other approaches involve rendering boundary meshes which separate regions. We propose an alternative approach that permits the user to explore the atlas in VR and gain an improved understanding of the spatial relations between labeled regions and a corresponding medical image. We use interactive exploded views to facilitate exploration of the 3D structure of the atlas at a wide range of scales - global, local and voxel.

An important application of our technique is the development of interactive anatomy teaching tools. Numerous studies support the theory that dynamic and interactive visualizations can improve learning outcomes and spatial reasoning [8].

The main contributions of our work are

- An expressive interaction technique which enables the user to create exploded views of an atlas.
- A demonstration of this technique on the AAL brain atlas.
- Results of a pilot study assessing the impact of our VR-based interactive visualization on anatomy learning outcomes.

In the remainder of this paper we will review the related literature, describe the implementation details of our approach, present the results and discuss future work.

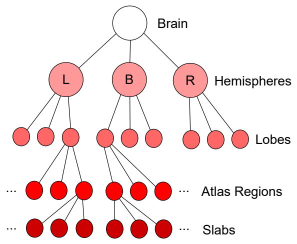
2 Related work

Exploded views change the spatial layout of a scene to improve the visibility of the structural relations among objects. Li et al. [9] generated 3D exploded views of CAD models based on the assembly-subassembly-part hierarchy. They constrain the object displacements using an explosion graph which prevents objects from overlapping. Balabanian et al. [1] presented a system which combined aspects of information visualization and illustrative visualization to browse the spatial and hierarchical relations in a volume dataset. McGuffin et al. [11] described several widgets for deforming the semantic layers of a volume dataset. Deformations included leafing, fanning and popping out a single layer. Although these were considered distinct methods of data manipulation, we will demonstrate that our interaction technique is general enough to create many of these types of deformations. The interaction approaches in the previous works rely on traditional mouse-based input, whereas in this work we develop a novel technique suitable for immersive virtual environments.

The ability to convey depth and spatial cues while enabling 3D data exploration make VR an appealing platform for visualization applications. Popular application domains include flow visualization [2], medicine [16], molecular visualization [12] and climate studies [7]. Although there is a long history of scientific visualization using VR, applications in information visualization and immersive analytics [3] are becoming more common.

Many commercial VR systems feature low-latency, high-precision tracking of both the HMD and handheld controllers. Standalone technologies, such as the MagicLeap system for hand tracking, enable intuitive interaction with virtual environments. Studies on CAVE and fishtank VR systems [4] found that handheld tracked controls led to improved performance on visual search tasks. Theart at al. [14] developed a VR microscopy volume rendering system and found that handtracking outperformed gamepad interaction on most tasks. In this work we explore the use of a pair of controllers in tandem to facilitate hierarchical data exploration and exploded view generation.

[*]e-mail: he370@purdue.edu
[†]e-mail: gguayaqu@purdue.edu
[‡]e-mail: tmcgraw@purdue.edu



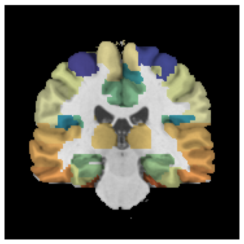


Figure 2: Hierarchical structure of the brain model (left), and a slice of the AAL atlas with colored regions overlaid on T1-weighted MRI (right).

3 Methods

Our application uses a hybrid rasterization/raycasting technique to visualize the atlas meshes and a registered 3D MRI image. But the meshes are not atomic entities in our technique - they can be broken into pieces. When the user explodes individual meshes the interior of the mesh reveals the underlying 3D medical image. The interior rendering is generated by raycasting using the segmented meshes as proxy geometry. In previous work [10] we described the hybrid rendering technique in detail and also described a less intuitive and time-consuming approach to creating exploded views using a mouse and keyboard. In the remainder of this section we describe how we assemble the hierarchy, and the user interactions we support in the VR application.

3.1 Hierarchical model assembly

Surfaces were extracted from the automated anatomical labeling (AAL) brain atlas which consists of 116 gray matter structures which were manually labeled from a human subject registered into a standard coordinate system [15]. A slice of the atlas is shown in Figure (2). We have further grouped the 116 labels into brain hemispheres (left, right) and brain lobes. The regions which do not belong to either hemisphere are considered bilateral. As the atlas is loaded in our application we compute an axis-aligned bounding box for each mesh. A complete bounding volume hierarchy is then built from the bottom-up. Although we focus on the AAL atlas, since it is widely used, the methods we describe here can be applied to any atlas with a template image and a label image.

CAD models describing manufactured products usually have an assembly process that defines an informative exploded view. For example, wheels slide onto hubs, and subassemblies are mounted to assemblies. Biological systems which grow into place don't have an analogy to such an assembly process. So we use the bounding volume hierarchy of the meshes to guide the creation of exploded views. Specifically, we use the parent-child relationships and the geometry of mesh bounding boxes to constrain the transformations.

3.2 Generating exploded views in virtual reality

Recent consumer VR systems consist of a head-mounted display (HMD) and motion-tracked controllers which provide a natural way of interacting with virtual environments. Using a standard mouse and keyboard with a HMD is inconvenient, but hand-held controllers permit the developer of a VR application to respond to user gestures, touches and button presses.

We aim to simplify the process of atlas navigation and reduce cognitive load by exploiting the user's sense of proprioception. The body-centered interaction [13] design principle can lead to more intuitive interaction by tightly coupling the human proprioceptive sense and the visual representation of the virtual environment. By using controller gestures to create explosion paths we reduce dependence on the user interface and reduce visual clutter on the screen.

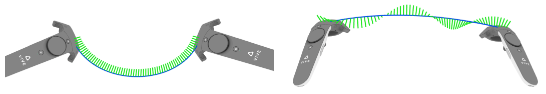


Figure 3: Hermite spline (blue) and a basis vector (green) of the rotation minimizing frame with end condition interpolation (left) and multiple rotations of the twist angle (right).

In this section we describe how we use a parametric curve which is controlled by the two tracked controllers to create an exploded view. Specifically, we formulate the mesh explosion paths as a cubic Hermite spline. Although using Hermite splines for general curve and surface design is complicated by the fact that tangent vectors must be specified at the endpoints, for interaction it is natural to map controller orientation to tangent vectors. Our interaction approach supports linear explosions, leafing, fanning and individual item pop-outs.

3.2.1 Tracked controllers

The tracked controllers generate new transformation matrices M_{left} and M_{right} representing position and orientation as the controllers move. Defining $\mathbf{z} = [0,0,0,1]^T$ and $\mathbf{r} = [1,0,0,0]^T$, we specify the endpoints and tangents of the Hermite curve in world coordinates as

$$\begin{aligned} \mathbf{p}_0 = M_{left}\mathbf{z} \quad &, \quad \mathbf{m}_0 = M_{left}\mathbf{r}, \\ \mathbf{p}_1 = M_{right}\mathbf{z} \quad &, \quad \mathbf{m}_1 = M_{right}\mathbf{r}. \end{aligned} \tag{1}$$

The Vive controller trackpad is a high-resolution touch-sensitive surface that supports motion gestures and clicking interactions. A trackpad is located on each controller where the user's thumbs can comfortably swipe them. Later in this section we describe how we use the trackpad to allow the user to specify a reparameterization of the Hermite curve.

3.2.2 Hermite spline

The Hermite cubic curve [5], $\mathbf{s}(u)$, interpolates between points $\mathbf{p}_0$ and $\mathbf{p}_1$ given the end point tangent vectors $\mathbf{m}_0$ and $\mathbf{m}_1$. The parametric equation is given by

$$\mathbf{s}(u) = \mathbf{p}_0 H_0(u) + \mathbf{m}_0 H_1(u) + \mathbf{m}_1 H_2(u) + \mathbf{p}_1 H_3(u) \tag{2}$$

with the parameter $u \in [0,1]$ and basis functions given by

$$\begin{aligned} H_0(u) = 2u^3 - 3u^2 + 1 \quad &, \quad H_1(u) = u^3 - 2u^2 + u \\ H_2(u) = -2u^3 + 3u^2 \quad &, \quad H_3(u) = u^3 - u^2. \end{aligned} \tag{3}$$

The curve, $\mathbf{s}(u)$, will determine the location of exploded atlas meshes, but in order to specify the mesh orientations we must define a coordinate frame in terms of $\mathbf{s}(u)$. In the following subsection we will describe how we choose the basis vectors b_0, b_1, b_2 which define that frame. The tangent vector to the curve determines b_0, and we can compute it analytically by differentiating Equation (2) to obtain $\mathbf{s}'(u)$. The unit tangent to the cubic Hermite curve is given by $\mathbf{T}(u) = \mathbf{s}'(u)/||\mathbf{s}'(u)||$. However, simply using the Frenet frame (tangent, normal, binormal) to define rotation matrices gives unsatisfactory results.

3.2.3 Rotation minimizing frame

The normal and binormal vectors of a curve can behave in unintuitive ways when the curve becomes flat and the magnitude of $\mathbf{s}''(u)$ becomes small. To resolve this problem we use a rotation minimizing frame [17] along the curve, and then compute a rotation matrix in terms of that frame. Rotation minimizing frames are commonly used for describing the geometry of extruded tube-like structures and defining camera paths.

The "double reflection" method of Wang et al. [17] constructs a rotation minimizing frame incrementally for a discretized curve by computing a rotation minimizing normal vector $\tilde{\mathbf{N}}(u_{i+1})$ from $\mathbf{s}(u_i)$, $\mathbf{s}(u_{i+1})$, $\mathbf{T}(u_i)$ and $\mathbf{T}(u_{i+1})$. In our implementation we recover a continuous frame by analytically computing $\mathbf{T}(u)$, interpolating $\tilde{\mathbf{N}}(u)$, and orthogonalizing the frame using the Gram-Schmidt process.

Interpolating the end conditions. The incremental nature of computing the rotation minimizing frame means that the left controller orientation is interpolated, but the right controller is not. In order to interpolate the right controller normal vector the authors of [17] suggest determining the rotation angle, α, required at the right endpoint, and defining $\alpha(u) = \alpha u$, the rotation angle about the tangent direction to be applied when computing the rotation minimizing $\tilde{N}(u)$. The results of computing the rotation minimizing frame with endpoint interpolation are shown in Figure (3).

To generate views of the atlas meshes exploded along a direction, d, our initial approach was to compute a parameter value, t_i, from each mesh bounding box center, c_i, as $t_i = c_i \cdot d$, then obtain u values by normalizing the t values to be in the range $[0,1]$. However, with this approach some meshes would project to nearby u values and never separate during explosion. So we sort meshes by increasing values of t_i, then separate adjacent sorted values by the projected half-widths, h, of the corresponding bounding boxes $(\tilde{d} \cdot h_a) + (\tilde{d} \cdot h_b)$ where a and b are adjacent meshes in the sorted list, and $\tilde{d} = [|d_x|, |d_y|, |d_z|]$. The result is that when fully exploded the meshes are separated such that their bounding boxes do not overlap.

Finally, the explosion translation matrix $D(u)$ is computed from $\mathbf{s}(u)$, and the rotation matrix is constructed column-wise $R(u) = [b_0(u)|b_1(u)|b_2(u)]$. Exploded meshes are rendered by taking the product $D(u)R(u)$ as the object-to-world transformation matrix.

Tracking twist angle. The rotation minimizing frame will snap back to 0° when rotating the controllers 360°. To handle an extended range of rotation angles we keep track of n - how many times the right controller has rotated through 360° relative to the left controller and incorporate that count in the function $\alpha(u) = (\alpha + 360° n)u$. The result of rotating the right controller multiple times relative to the left controller is shown in Figure (3).

CD gain. The control-display (CD) gain [6] describes how controller inputs are mapped to pointer motion in object selection and pointing tasks when designing a graphical user interface. For example, mouse pointer velocity in screen-space is typically not directly proportional to mouse velocity. Fast motions of the mouse are accelerated (CD gain > 1) so that the pointer can quickly cover a great distance. CD gain reduces the need for "clutching" - lifting the mouse and repositioning to gain an increased range of motion.

In our application, clutching takes the form of regripping the controller when trying to specify a large twist angle. Twist angles greater than 90° are difficult to command with handheld controllers due to the limited range-of-motion of the wrist. But, high angles are desirable in order to create fanning transformations. To enable large twist angles we incorporate a CD gain into the twist angle computation to permit an extended range of angles, $\alpha(u) = CD(\alpha + 360° n)u$. Studies have shown that a CD gain values < 3 are useful for mouse pointers, but we used a value of $CD = 4$ with no problems.

Arc-length parameterization. The length of the tangent vectors $\mathbf{m}_0$ and $\mathbf{m}_1$ are free parameters, and in order to give us better control over object positioning we select vector magnitudes which give us an approximate arc-length parameterization of $\mathbf{s}(u)$.

We use a simple heuristic to adjust the lengths of the end tangents $\mathbf{m}_0$ and $\mathbf{m}_1$ to approximate an arc length parameterization. Each frame we compute the speed of the parameterization at both endpoints, $||s'(0)||$ and $||s'(1)||$, and the middle of the curve, $||s'(\frac{1}{2})||$. Then, if the speed at an endpoint is greater than the speed at the midpoint the length of the tangent at that endpoint is decreased.

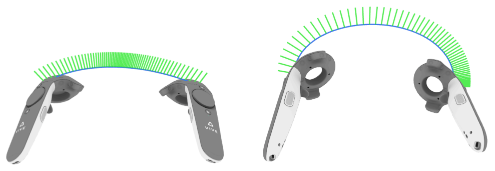


Figure 4: Example reparameterized curves of $\mathbf{s}(v(u))$. The basis vectors in green are plotted at equally spaced intervals in u.

Similarly, if the speed at the endpoint is slower than the speed at the midpoint the corresponding tangent vector is lengthened. The tangent vector lengths c_0 and c_1 are initialized to 1 and then updated each frame by to the rule

$$c_0 = c_0 \frac{||s'(\frac{1}{2})||}{||s'(0)||} \quad , \quad c_1 = c_1 \frac{||s'(\frac{1}{2})||}{||s'(1)||}, \tag{4}$$

.

Reparameterization of $s(u)$**.** The curve $\mathbf{s}(u)$ defines the spacing of meshes along the explosion path. To permit variable spacing we define the reparameterization $\mathbf{s}(v(u))$. Unlike changing the length of the end tangent vectors, reparamaterization does not change the shape of the curve. We define $v(u)$ in terms of the Hermite cubic function

$$v(u) = v_0 H_0(u) + w_0 H_1(u) + w_1 H_2(u) + v_1 H_3(u). \tag{5}$$

Since the range of parameter values should remain $v \in [0,1]$ we must have $v_0 = 0$ and $v_1 = 1$. When $w_0 = w_1 = 1$ we obtain $v(u) = u$ and there is no reparameterization. As the value of w decreases the curve $v(u)$ flattens out at the corresponding endpoint slowing down the parameterization speed and making more u values map to the same v. This causes the exploded meshes to be bunched together (no explosion) at one end of the curve. As parameterization speed increases the meshes spread apart. Example reparameterized curves with variable spacing along the curve are shown in Figure (4).

The end slopes w_0, w_1 are free parameters the user can control with the trackpad. Swiping in an outward direction (to the right on the right controller, and to the left on the left controller) increases the speed of parameterization on the corresponding end of the curve which allows the user to slide the exploded meshes along the curve.

Mesh labeling. The main rationale for reparameterizing the Hermite curve is to change the spacing between meshes along the explosion path and give an unoccluded view of the region shapes which allows the user to see how the shapes fit together. But this explosion also gives us space to place region labels and leader lines. Based on the parameterization speed $||s'(u)||$ we place a label for each mesh radially offset from the center of the mesh bounding box.

4 Results

Our atlas visualization technique was implemented in C++, OpenGL and OpenVR and executed on a PC with a 3.0 GHz Intel CPU, 16 GB RAM, Nvidia GeForce GTX Titan X video card and an HTC Vive VR system. The results were generated by loading the meshes segmented from the AAL atlas label image (a total of 575,460 triangles) and the corresponding T1-weighted MRI image ($181 \times 217 \times 181$).

4.1 Interaction

Figure (1) shows some of the controller gestures made by a user when creating exploded views. Our application is not a room-scale VR experience, meaning that the user can be seated at a desk. This permits the user to rest their elbows on the desk which reduces fatigue. In testing the VR application we attempted to recreate several of the volume interactions described by McGuffin et al. [11].

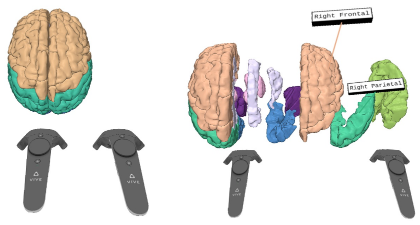


Figure 5: Linear explosion of cerebral lobes. Spreading the controllers apart spreads the meshes further and scales the meshes larger.

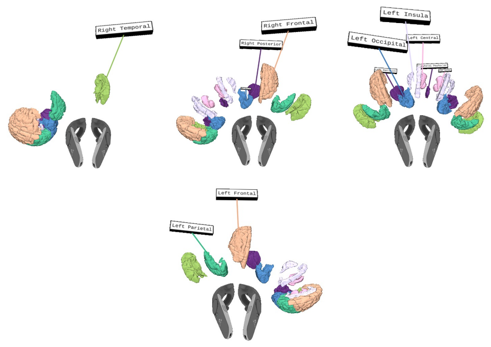


Figure 6: Meshes slide along the explosion path as the reparameterization is changed by swiping the thumb on the trackpad.

Linear explosion. The linear explosion can be generated by simply spreading the controllers apart while keeping them parallel with each other, as shown on the left column of Figure (1). Reparameterizing the curve with the Vive trackpad allows the user to control which parts of the atlas remain unexploded (see Figure (5)).

Leafing interaction. Also referred to as ”riffling”, this interaction transforms a stack of objects as if they were the pages of a book being leafed through. To achieve this, the user rotates the controllers about their long axes to distribute the objects along a curved arc (middle of Figure (1)), then swipes along the trackpad to perform the riffling. Leafing meshes are shown in Figure (6), and leafed slabs of an exploded mesh are shown in Figure (7, left). Leafing open the hemispheres to reveal the bilateral structures is demonstrated in Figure (8).

Fanning interaction. The result of this interaction (see Figure (7, middle)) is like spreading a hand of playing cards so that the suits of the cards are visible. To achieve this the user holds the controllers side-by-side then rotates them about the line joining the two controllers. Then the controllers can be further spread apart to generate the exploded view as shown in Figure (1).

Popping out a single slab. This transformation allows a particular item to be emphasized by pushing it away from other items (see Figure (7, right)). In our application we only permit slabs to

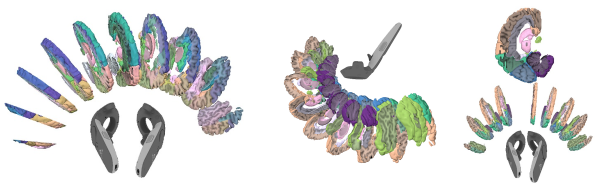


Figure 7: Leafing (left) fanning (middle) and popping out (right) slabs of an exploded mesh.

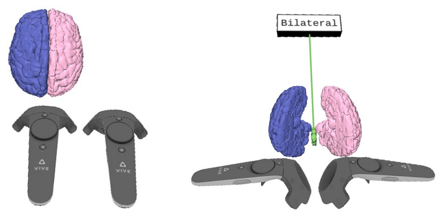


Figure 8: Leafing the hemispheres of the brain by rotating the controllers relative to each other.

be popped out and we billboard them so that the user can get a clear view of the MRI image within. This transformation is implemented by applying an additional displacement in the neighborhood of $u = 0.5$. As the Hermite curve is reparameterized meshes sliding past the middle of the curve will automatically pop out and face the viewer.

4.2 Pilot study

To assess the effectiveness of our visualization and interaction method we performed a user study in which the task was to learn to identify the lobes of the brain. Our hypothesis is that interactive VR experiences will lead to better understanding of 3D shapes and spatial relations than traditional paper learning materials. Twenty participants were recruited from Purdue University and a pre-test and post-test of labeling the lobes of the brain was administered to each participant. Participants used a VR tutorial which asked them to point at labeled lobes, then interact with an exploded view. In the pre-test and post-test we asked participants to also rate their certainty in their answers on a 4-point Likert scale, from ”very unsure” to ”very sure”.

The VR application had a positive influence on test scores. Based on a one-sample t-tests of learning gain score ($\mu = 47\%, \sigma = 57\%$) we reject the null hypothesis that the VR treatment had no effect on gain score. A similar conclusion can be drawn from the t-tests on certainty - VR appears to increase certainty in test responses. In future experiments we intend to compare with a control group using traditional printed materials to assess learning outcomes. We will also include deeper cerebral structures in the tests which will require users to perform more thorough 3D investigations of brain structure.

5 Conclusion and Future Work

In this work we have described an interactive atlas visualization technique which is capable of creating exploded views using a pair of motion-tracked VR controllers. Atlas-defined regions can further be exploded to reveal image intensities in context with segmented surfaces. Our system was demonstrated on the AAL neuronanatomical atlas and was able to maintain the high framerates required for VR applications. Results of a pilot study show that score gain and certainty gain were positive.

In future work we intend to perform more user studies in order to assess the usability of the system and guide further interaction design. We also plan explore the fusion of functional and connectivity information into the atlas visualization.

Acknowledgments

This work was supported in part by a grant from the Purdue Polytechnic Institute. The GTX Titan X videocard used for this research was donated by the NVIDIA Corporation.

References

[1] J.-P. Balabanian, I. Viola, and E. Gröller. Interactive illustrative visualization of hierarchical volume data. In *Proceedings of Graphics*

Interface 2010, pp. 137–144. Canadian Information Processing Society, 2010.

[2] S. Bryson and C. Levit. The virtual windtunnel-an environment for the exploration of three-dimensional unsteady flows. In *Visualization, 1991. Visualization'91, Proceedings., IEEE Conference on*, pp. 17–24. IEEE, 1991.

[3] P. W. Butcher, J. C. Roberts, and P. D. Ritsos. Immersive Analytics with WebVR and Google Cardboard. *IEEE Visualization Posters 2016*, 2016.

[4] C. Demiralp, C. D. Jackson, D. B. Karelitz, S. Zhang, and D. H. Laidlaw. Cave and fishtank virtual-reality displays: A qualitative and quantitative comparison. *IEEE Transactions on Visualization and Computer Graphics*, 12(3):323–330, 2006.

[5] G. Farin. *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide*. Elsevier, 2014.

[6] C. Gibbs. Controller design: Interactions of controlling limbs, time-lags and gains in positional and velocity systems. *Ergonomics*, 5(2):385–402, 1962.

[7] C. Helbig, H.-S. Bauer, K. Rink, V. Wulfmeyer, M. Frank, and O. Kolditz. Concept and workflow for 3d visualization of atmospheric data in a virtual reality environment for analytical approaches. *Environmental earth sciences*, 72(10):3767–3780, 2014.

[8] T. N. Höffler. Spatial ability: Its influence on learning with visualizationsa meta-analytic review. *Educational psychology review*, 22(3):245–269, 2010.

[9] W. Li, M. Agrawala, B. Curless, and D. Salesin. Automated generation of interactive 3d exploded view diagrams. *ACM Transactions on Graphics*, 27(3):101, 2008.

[10] T. McGraw and A. Guayaquil-Sosa. Hybrid rendering of exploded views for medical image atlas visualization. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization, In press*.

[11] M. J. McGuffin, L. Tancau, and R. Balakrishnan. Using deformations for browsing volumetric data. In *IEEE Visualization Conference*, pp. 401–408. IEEE, 2003.

[12] M. Norrby, C. Grebner, J. Eriksson, and J. Bostrom. Molecular rift: virtual reality for drug designers. *Journal of chemical information and modeling*, 55(11):2475–2484, 2015.

[13] M. Slater and M. Usoh. Body centred interaction in immersive virtual environments. *Artificial life and virtual reality*, 1:125–148, 1994.

[14] R. Theart, B. Loos, and T. Niesler. Virtual reality assisted microscopy data visualization and colocalization analysis. *BMC Bioinformatics*, 18(64), 2017.

[15] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. Joliot. Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain. *Neuroimage*, 15(1):273–289, 2002.

[16] A. Van Dam, D. H. Laidlaw, and R. M. Simpson. Experiments in immersive virtual reality for scientific visualization. *Computers & Graphics*, 26(4):535–555, 2002.

[17] W. Wang, B. Jüttler, D. Zheng, and Y. Liu. Computation of rotation minimizing frames. *ACM Transactions on Graphics (TOG)*, 27(1):2, 2008.