Realtime Visualization of the Connectome in the Browser using WebGL

Dan Ginsburg[1,2], Stephan Gerhard[3], John Congote[4,5], Rudolph Pienaar[1,2]

[1]Children's Hospital Boston, [2]Harvard Medical School, [3]École Polytechnique Fédérale de Lausanne, [4]Vicomtech-IK4, [5]EAFIT University

Introduction

The introduction of WebGL[1] has made exploration of complex 3D datasets possible in the browser, enabling visualizations across a wide range of devices without any client-side software requirements other than a WebGL-enabled browser. This poster introduces several WebGL-based modules that were developed for exploration of connectome data all of which are available as open source software. The viewer we developed includes modules for visualization of FreeSurfer[2] surfaces, cortical surface curvature measures, tractography generated with the Diffusion Toolkit[3], MRI volume data, and connectivity data generated with the Connectome Mapping Toolkit[4]. The viewer allows interactive exploration of the datasets all within the browser (Fig. 1).

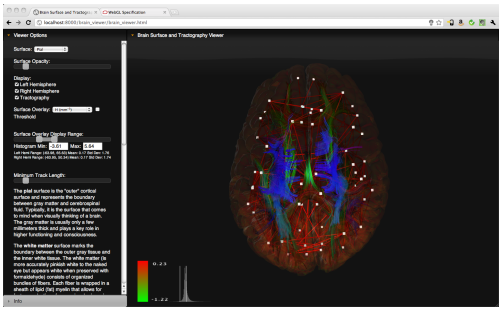

Fig. 1. Screenshot of the WebGL-based visualization. The visualization is composed of the pial surface colored with the per-vertex mean curvature, fiber tracts colored by direction, regions-of-interest rendered with cubes, and lines between regions denoting connectivity. The connectivity lines are colored based on the mean fiber tract length connecting the two regions-of-interest.

Methods

The publicly available Connectome Mapper provides a pipeline to automatically generate structural networks from raw diffusion MRI data for the human brain. In the segmentation stage, T1 MPRAGE MRI is processed by Freesurfer producing gray/white matter segmentations. The Diffusion Toolkit is used for reconstruction, and a deterministic streamline algorithm is used for tractography, generating fiber tracts of the same subject. A parcellation is generated for cortical and subcortical regions-of-interest. These datasets are then coregistered, and a network is generated weighting the connectivity between regions based on the fiber tracts.

In our project, the results of the Connectome Mapper are directly loaded in the browser using WebGL and JavaScript. The FreeSurfer cortical surface reconstruction binary files are loaded and processed in JavaScript and converted to WebGL vertex buffer objects for rendering. The surfaces are overlaid with per-vertex curvature values computed during the FreeSurfer processing stream. The tractography data is likewise parsed in the JavaScript code and rendered as line primitives colored based on direction. Finally, the structural network itself is converted to JSON (JavaScript Object Notation) as an offline preprocess and loaded into the browser using JavaScript. The networks are visualized in 3D along with the fiber tracts and surfaces enabling exploration of connectivity information in realtime.

Rendering

The FreeSurfer surfaces are rendered with curvature overlays that can be dynamically colored based on a histogram of curvature values (Fig. 2). The histogram widget is rendered using WebGL triangle primitives that are generated in the JavaScript code. All of the rendering is performed using GPU shaders written in the OpenGL ES Shading Language[5]. The surface vertex shader computes per-vertex colors based on the min/max histogram range and the per-vertex curvature values. The surface fragment shader computes per-pixel diffuse and specular lighting that attenuate the curvature-based color value.

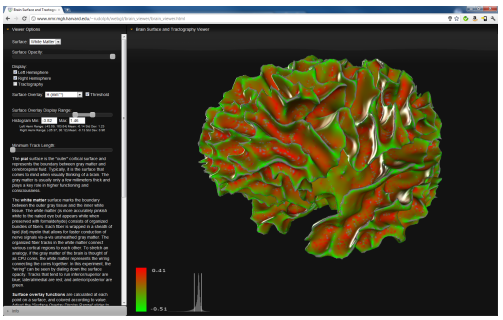

Fig. 2. Gray/white surface rendered with mean (H) curvature colored and thresholded based on curvature value histogram.

The tractography is rendered using WebGL line primitives (Fig. 3). Each fiber tract is defined by a set of points. After loading the tractography file, each fiber tract is assigned a color based on the absolute value of the unit vector pointing in the direction from the start point to the end point of the tract. The length of the tract (in *mm*) is stored in a per-vertex attribute along with the position and color. The minimum tract length is dynamically tuned at runtime (adjustable by a slider) and this minimum value is placed in a uniform variable in the vertex shader. The vertex shader determines whether the length of the tract is greater than the minimum length to render, and if not it throws the tract away by transforming it to a degenerate primitive. The entire tractrography set for the brain is rendered using a single draw call with one vertex buffer object and since the length threshold is taken care of in the vertex shader there is no dynamic geometry generation done in the JavaScript. This was done for efficiency, otherwise the tracts would have to be broken up into multiple draw calls which would negatively impact performance.

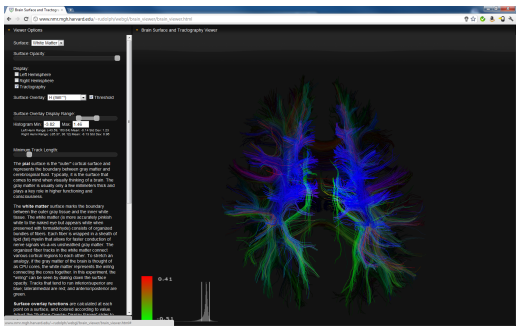

Fig.3. Tractography rendered using WebGL line primitives.

An additional rendering technique that was developed is the direct volume rendering of MRI data simultaneously with the tractography (Fig. 4). The volume renderer loads the MRI data from the server into a tiled 2D texture and then renders it by performing ray-tracing in the shader. [6]

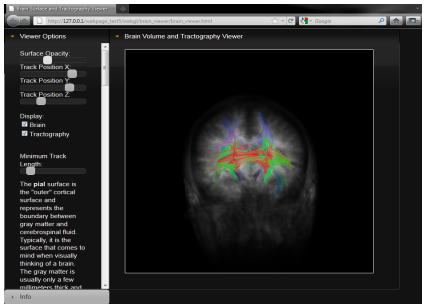

Fig.4. Direct volume rendering of T1 MPRAGE MRI drawn simultaneously with tractography visualization.

Conclusions

Our viewer has demonstrated the potential of using WebGL as a basis for interactive visualization of neuroimaging data including FreeSurfer surfaces, curvature measures, tractography, and connectome data. We have made our implementation available as open source BSD-licensed software so that others can extend and refine our work. Due to the increasing ubiquity of WebGL, we expect to see more visualization tools developed with it. The strength of WebGL is in its ability to provide efficient access to GPU rendering hardware whilst requiring no special client-side software. This makes it a platform with great potential for neuroimaging tools, particularly those providing web-based interfaces for automatic pipelining of neuroimage data processing.

References

[1]WebGL Specification, http://www.khronos.org/webgl/

[2]Ruopeng Wang, Van J. Wedeen, TrackVis.org, Martinos Center for Biomedical Imaging, Massachusetts General Hospital

[3]Dale, A.M., Fischl, B., Sereno, M.I., 1999. Cortical surface-based analysis. I. Segmentation and surface reconstruction. *Neuroimage* 9, 179-19

[4]Connectome Mapping Toolkit, http://www.cmtk.org/

[5]OpenGL ES Shading Language Version 1.00, http://www.khronos.org/registry/gles/specs/2.0/GLSL_ES_Specification_1.0.17.pdf

[6]Congote, J., Alvaro, S., Kabongo, L., Aitor, M., Posada, J., Ruiz, O. Interactive visualization of volumetric data with WebGL in real-time. *In Proceedings of the 16th International Conference on 3D Web Technology (Web 3D '11)*. ACM, New York, NY, USA, 137-146.

For further information

The full source code to the viewer can be downloaded from github at **https://github.com/danginsburg/webgl-brain-viewer**