fMRI Brain Slice

Zhen Rong (Dennis) Liew
ZhenRong.Liew001@umb.edu
University of Massachusetts Boston

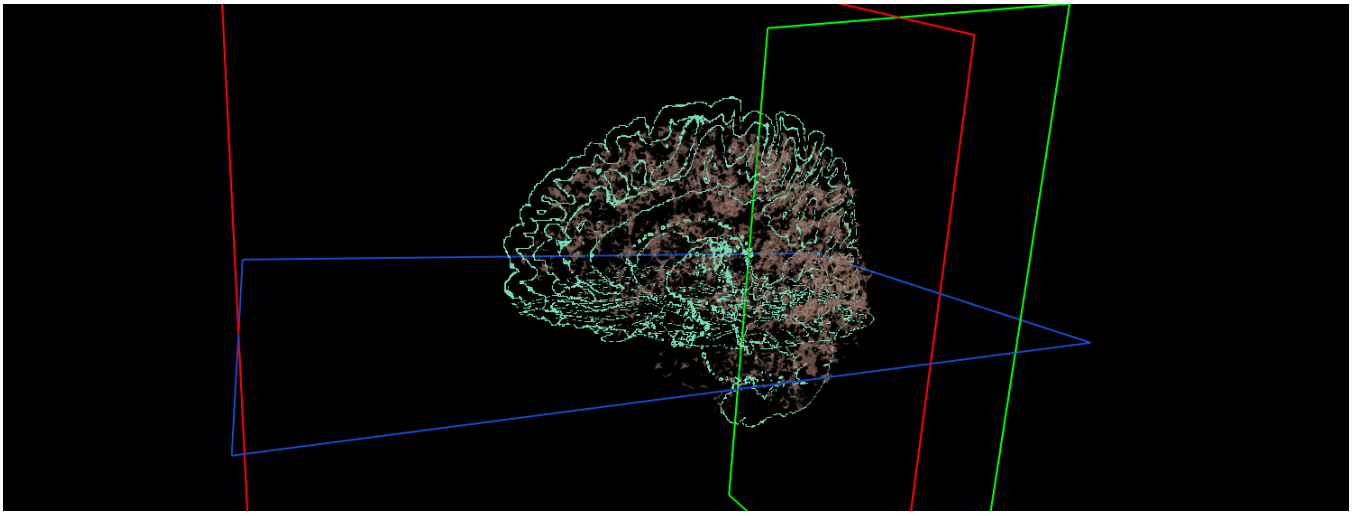

Figure 1: fMRI data in X,Y,Z slices.

ABSTRACT

KEYWORDS

XTK, Visualization, Brain, fMRI

ACM Reference Format:
Zhen Rong (Dennis) Liew. 2020. fMRI Brain Slice. In *CS460: Computer Graphics at UMass Boston, Fall 2020*. Boston, MA, USA, 3 pages. https://CS460.org

1 INTRODUCTION

This project was an exploratory experiment on some personal fMRI data. The main objective was to explore the different ways of visualizing the brain as well as creating 3D meshes from the ground up.

2 RELATED WORK

XTK [2] & XTK Toolkit [1] & Paraview [3].

2.1 Method

The project uses the XTK framework to initially import the volumes. The framework provides several API to move the x,y,z slice indices. With that, I was able to use dat.GUI to control the indices. As for the self generated mesh. I wrote some conversion code to take NifTI files and output vtk files to be imported by XTK. I experimented on several different output file formats and it seems vtk is the best considering that XTK does not support some.

Addition of several diffenet volumes required the creation of a controller to manipulate visibility of certain unselected volumes and meshes.

Animation was created using a sine wave function to created a continuous oscillating movement. From this, I created several variations of frequency and amplitude, then applied it to change the different values of the mesh and volume. The different sine waves were to create some notion of complexity and interesting colour variations.

2.2 Implementation

The following code snippet shows dat.GUI controlling the different index values on the x,y,z plane slices.

```
  var volumegui = gui.addFolder('Volume');
var sliceXController = volumegui.add(controller.index_params, 'i
  volume1.dimensions[2] - 1).onChange(controller.setX);
var sliceYController = volumegui.add(controller.index_params, 'i
  volume1.dimensions[1] - 1).onChange(controller.setY);
var sliceZController = volumegui.add(controller.index_params, 'i
  volume1.dimensions[0] - 1).onChange(controller.setZ);
volumegui.open();
```

The following code shows several sine wave function for the animation of colour and index values.

```
y0 = Math.abs(Math.sin(0.3*(counter)));
y = Math.abs(Math.sin(1*(counter)));
y1 = Math.abs(Math.sin(1.5*(counter)));
```


CS460, Fall 2020, Boston, MA

ACM ISBN 1337.
https://CS460.org

```
y2 = Math.abs(Math.sin(2*(counter)));
y3 = Math.abs(Math.sin(2.5*(counter)));
for (i = 0; i < volumes.length; i++) {
  if (i==3) {
    volumes[i].upperThreshold = y * 2239;
    volumes[i].maxColor = [y3,y1,y2];
    volumes[i].minColor = [y3,y2,y1];
  }
  else {
    volumes[i].upperThreshold = y * 255;
    mesh_grey.color = [y1,y2,y3];
    volumes[i].maxColor = [y3,y2,y1];
    volumes[i].minColor = [y3,y1,y2];
  }
```

The following code shows a small part of the conversion code of NifTI to vtk.

```
writer = vtk.vtkSTLWriter()
writer.SetInputConnection(smoother.GetOutputPort())
writer.SetFileTypeToASCII()
writer.SetFileName(filename_stl)
writer.Write()
```

2.3 Milestones

The project had 2 initial milestones, listed in subsection. However, some challenges arose and had to add new milestones to overcome them.

2.3.1 Milestone 1. Slicing brain data into x, y and z slices. Enables the brain to be viewed in 3 planes.

2.3.2 Milestone 2. Add different view points and brain data type.

2.3.3 Milestone 3. Highlighting certain areas of the brain to showcase activity on certain stimuli input.

2.3.4 Milestone 4. Creation of 3D mesh using NifTI files as input.

2.3.5 Milestone 5. Add animation loop of colour change and automatic slice oscillation.

2.4 Challenges

2.4.1 Challenges 1. Implementation of highlighting function on certain areas of the sliced brain data served particularly challenging. This seemed to involve creationg of a labelmap file and labelmap colortable file - both of which I did not already have, and therefore cannot manipulate. My limited knowledge on the matter has led me to many dead ends.

2.4.2 Challenges 2. Volume rendering using XTK is very computationally intensive. It creates a very noticeable lag on the browser and needs to reload the volume every time the volume needs to be rotate. Furthermore, the volume engulfs the 3 slices and cannot show both of them at the same time.

2.4.3 Challenges 3. In response to the previous 2 challenges, I decided to create 3D meshes from the NifTi file. This came with several challenges along the way. Such as NifTi file parsing, fixing the normals and redundant vertices of many generated meshes, and experimenting with different output file formats.

3 RESULTS

The final version of the project looked like the following figure 2. On the top-right, containing the GUI for controlling the xyz indices, 6 axial views, different brain data, and animation loop.

Figure 3 represents the 3D mesh created in python. The mesh was presents the "folds" and "holes" on the brain itself - an inverse of most brain models in other words. This was to achieve a 3D mesh that did not cover the 3 xyz slices, thus allowing this mesh to be showcased alongside the slices while still holding the general shape of the brain.

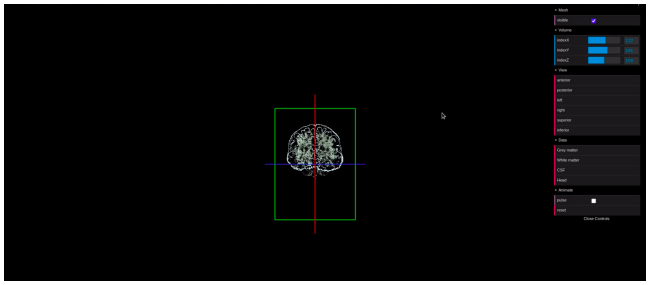

Figure 2: Viewport of the project.

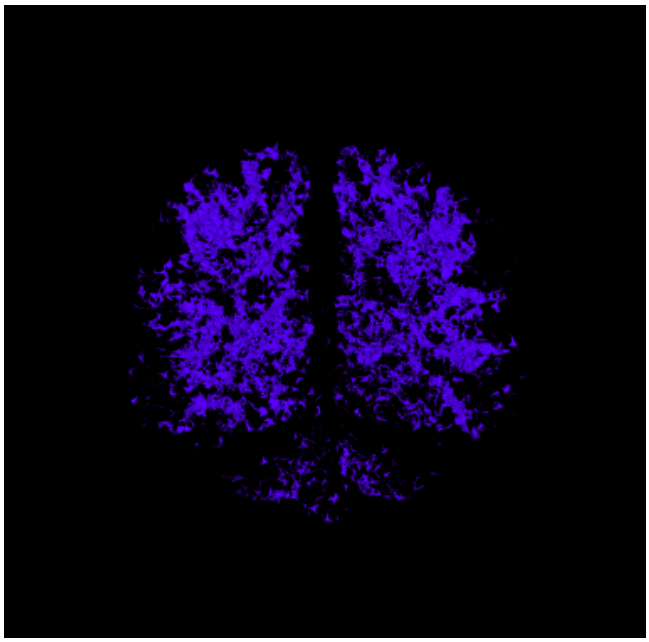

Figure 3: Topdown view of a 3D mesh created.

4 CONCLUSIONS

In conclusion, this project was a great way for me to explore and apply what we have learnt in-class into "real-world" data. Although most of the functionalities implemented in this project are not new and groundbreaking, it was interesting and fun to tackle the challenges and come up with different solutions to them.

REFERENCES

[1] Daniel Haehn et al. [n.d.]. The X Toolkit: WebGL™ for Scientific Visualization. ([n. d.]).

[2] Daniel Haehn, Nicolas Rannou, Banu Ahtam, P. Ellen Grant, and Rudolph Pienaar. 2012. Neuroimaging in the Browser using the X Toolkit. *Frontiers in Neuroinformatics* (2012).

[3] Kitware. [n.d.]. Paraview. ([n. d.]).