A Report on Music Visualization

3D HUMAN BRAIN GENE VISUALIZATION
[Independent Study]

Saranya Gopinath

Supervised by
Dr. Shiaofen Fang

Department of Computer and Information Science
Indiana University Purdue University Indianapolis (IUPUI)

AUGUST 2016

Table of Contents

Abstract

Over the years, advanced computer graphics and visualization have been used increasingly such that it can add value to a wide variety of medical applications. The significant improvements in the stream of computer science has led to the development of powerful 3D image analysis and procedures that provide promising results to the field of medicine. Visualization algorithms support both direct volume rendering (DVR) algorithms such as ray casting and surface fitting algorithms such as marching cubes.

Image data obtained from three dimensional human MRI or CT scans contain very useful information about internal anatomy of human body and they can be effectively rendered with the help of DVR algorithms without losing those information whereas the surface fitting algorithms fail to do so. This is the reason why DVR is preferred over surface fitting algorithms.

In this report, I present the related work of 3D volume rendering of Human Brain generated from the Allen Human Brain Atlas [Brain]. The MRI image of a donor H0351.2001, a 24-year-old African American male is downloaded and rendered as a three dimensional volume with dummy gene expression information which will later be manipulated with the actual information in the future work. The rendering of MRI and gene expression volume were done with the help of VTK and Python.

Chapter 1 Introduction

A three dimensional view of any part of the human body can provide doctors and medical practitioners with a detailed valuable insight into the organ. 3D Volume rendering of human brain along with an overlay of dummy gene expression information is performed in this project. Here, a 3D MRI image data is downloaded from the Allen Brain Atlas (ABA) in order to interactively render a volume to study how the gene expression values can be spread throughout the brain [Brain]. Gene expression is the process by which a gene which is a segment of DNA, is turned into a protein or RNA structure that are responsible for running everything in our bodies [ABADV]. Every cell in our body possess same genes. The active genes are the one that produce the gene expressions whereas the inactive genes are not capable of producing genes. It is well known that 'Brain' has the highest production of genes in our bodies. Changes in these genes in the brain can tell us the risk of a particular disease.

***Keywords** – Allen Brain Atlas, 3D, Gene expression, Human brain, VTK, Python*

1.1 Background

Human Brain visualization can be represented in both 2D and 3D, but which of the two is more useful to doctors has to be considered. A 2D volume view of an organ can be very difficult to interpret. Thus a 3D interpretation of the brain with its gene expression information is captured here. Combining both structural and functional neuroimaging results is very tedious and requires advanced visualization tools and techniques. The NIFTI file format allows a convenient utilization of range of existing 3D visualization packages.

1.2 Dataset

The data used in this project + retrieved from the ABA, a resource that is widely available through the ABA data portal. The MRI image of a donor H0351.2001, a 24-year-old African American male was downloaded from the following link [Brain];

Link: http://human.brain-map.org/mri_viewers/data

1.3 Challenges and Roles

Though the ABA offers their own search engine and tool for researches to view their growing online collection of online public data sets consisting of an extensive gene expression and neuroanatomical data from human and mouse brain, a number of their tools limit the amount of genes and brain structures researchers can view simultaneously. Visualization in three dimension is often critical to developing better insights into the high dimensional structure of data sets. The data processing of the MRI image in a tool like VTK Python was challenging as there were very few libraries supporting the NIFTI image format and there were version compatibility issues between VTK and the functions used. Animation of those volume-visualized data is also critical for data analysis process. My role in this project was to develop such an advanced 3D volume rendering model of a human brain MRI sample along with an overlay of assumed gene expression values using VTK and Python.

Chapter 2 Tools Used

2.1 Python

Python as the core programming language coupled with high level visualization packages are used to handle broad array of neuroimaging data visualization applications. Python can serve as a convenient, powerful and manageable scripting language for generating volume visualizations, optical image reconstruction and hierarchical statistical analysis better than time consuming C++ coding and debugging.

The following third party Python modules were used;

2.1.1. VTK

The Visualization toolkit (VTK) is an open-source software used for imaging, computer graphics and visualization. This can be written in C++ and scripting languages like Python, Tcl/Tk and Java. VTK has a built in capacity to render multiple volumes at once, allowing different color transfer functions for coloring each volume as needed.

2.1.2. NUMPY

It is one of the fundamental packages used for scientific computing in Python. It is a powerful N-dimensional array (ndarray) object providing mathematical functions and has the capability to handle large multi-dimensional arrays.

2.1.3. NIBABEL

It is a pure Python package that is mainly useful for reading, writing and manipulating common neuroimaging file formats.

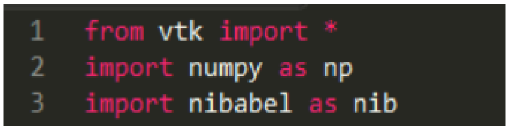
Chapter 3 Visualization Techniques

As per VTK user's guide, "volume rendering is a term used to describe a rendering process applied to 3D data where information exists throughout a 3D space instead of simply on 2D surfaces defined in 3D space". According to [ELVINS], volume visualization is nothing but "the process of projecting a multi-dimensional data set over a two dimensional image plane/surface in order to get an understanding of how the structure (or lack of structure) is contained within the volumetric data".

The three dimensional volume rendering rendered from human brain MRI sample in NIFTI format requires the following imports before coding further.

3.1. Imports

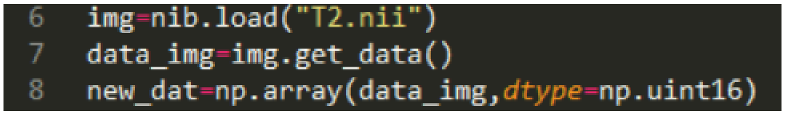
The imports in general is the first step in any coding structure to properly associate the necessary libraries or plugins to work with one's code. Here we have used the following;



```
from vtk import *
import numpy as np
import nibabel as nib
```

3.2 Image -Data -Input

Once the imports are mentioned, the MRI image is downloaded from the Allen Human Brain Atlas website for the donor H0351.2001, a 24-year-old African American male. The image is in form of zipped NIFTI file format '.nii.gz' which has to be unzipped into '.nii' format in order to proceed. The NIBABEL package from Python will be able to read such file formats and manipulate them efficiently.



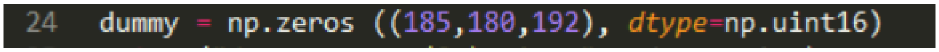
```
img=nib.load("T2.nii")
data_img=img.get_data()
new_dat=np.array(data_img,dtype=np.uint16)
```

The NIFTI image data is being read in the above code and the array information of this 3D image is captured such that the 3d array data can be used further for volume rendering purposes. The key then, is calling the appropriate method to set the desired data type of our output image. The data type of this array is very important to be noted and the key is to call the appropriate method to set the desired data type of the output image. Here, the data type of the image is 'int16' by default and it is then manipulated into 'uint16' for avoiding negative values.

3.3 Gene Expression -Data- Input

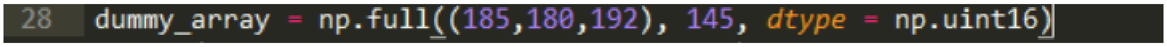
The gene expression data can be extracted from the MRI image of the same donor after applying suitable algorithms. Here, the 3D array values are randomly set to check how the overlaying happens upon the 3D MRI volume of the brain data. The data type and dimensions of the volumes should be the same in order to perform any addition or overlay of multiple volumes before rendering. So, the data type is set to ‘uint16’ for both and the dimension of the new dummy 3D array created is same as that of the MRI 3D array which is (180,185,192).

The below example creates a 3 dimensional array with all values as 0 with the specified dimension and data type using NUMPY.



```
dummy = np.zeros ((185,180,192), dtype=np.uint16)
```

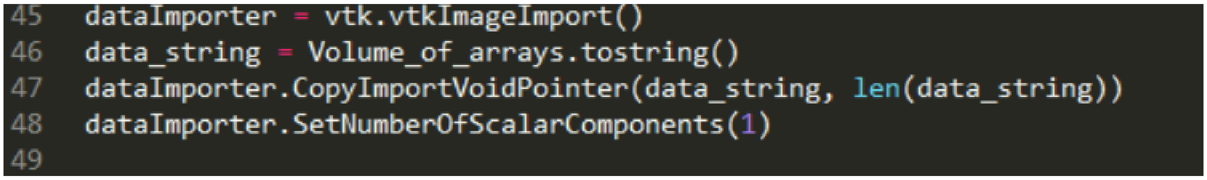
The following code will create a 3 dimensional array filled with values as 145 with the specified dimension values and data type using NUMPY.



```
dummy_array = np.full((185,180,192), 145, dtype = np.uint16)
```

3.4 Data Import

The combined array information has to be imported before proceeding to set the transfer functions. The array from the MRI and the array that was created as dummy are both added with the help of NUMPY addition where two arrays can be added just as simple as two numbers, provided the data type, number of elements and dimensions are same. Once the two arrays are added together as one volume, this new volume is imported using VtkImageImport() function.



```
dataImporter = vtk.vtkImageImport()
data_string = Volume_of_arrays.tostring()
dataImporter.CopyImportVoidPointer(data_string, len(data_string))
dataImporter.SetNumberOfScalarComponents(1)

```

3.5 Setting up connections

The mapper function for the volume has to be set up after the new volume data is created. VtkFixedPointVolumeRayCastMapper() function is used as a replacement for VtkVolumeRayCastMapper() function of VTK due to version compatibility restrictions. The volume data is connected with the mapper function.

3.6 Transfer functions

Before starting the volume rendering process, some prep work has to be done which is very important for how the rendering should eventually look like.

3.6.1. Color function

The color transfer function is what that defines the color values for the 3D volume. This has to be an instance of the VtkColorTransferFunction() class and acts as a map of scalar pixel values to a specified RGBA (Red, Blue, Green, Alpha) color. This function helps in assigning what color has to be given for every pixel value that is read from the volume. Here, a criterion is set such that every pixel value of the new volume is read element by element to check if the pixel values are either less than 256 or greater than 256. So we define the color transfer function based on these values and assign the pixel values to greyscale if it is less than 256 and color if it is greater than 256.

3.6.2. Scalar Opacity function

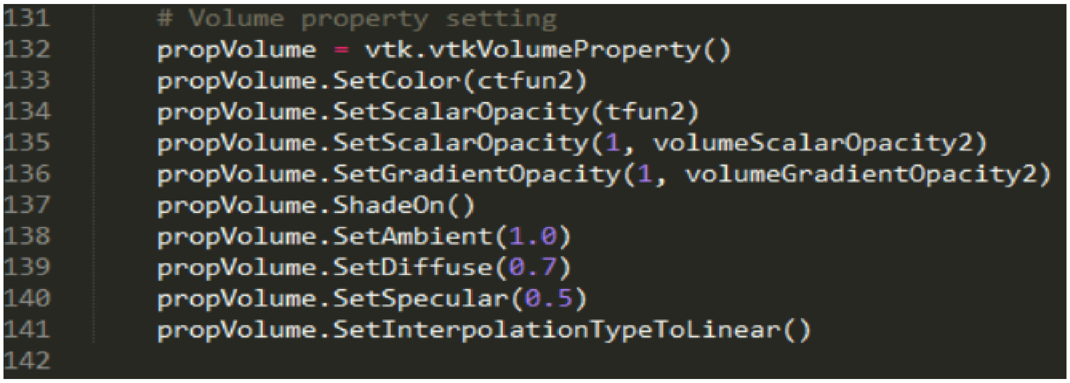
The opacity value is set for every pixel based on the value in the volume. The opacity function is set with the help of VtkPiecewiseFunction() which defines 1-1 mapping. It also consists of AddPoint() method to add new points. According to [VTK], “When the scalar opacity or the gradient opacity of a volume is not set, then the function is defined to be a constant value of 1.0. When a scalar and gradient opacity are both set simultaneously, then the opacity is defined to be the product of the scalar opacity and gradient opacity transfer functions.”

3.6.3. Gradient Opacity function

Now that the color function and scalar opacity function were defined, the gradient opacity function should be defined in order to get the inside of the brain rather transparent and make the boundaries clearer. As the scalar opacity just assigns a value per pixel intensity, gradient opacity is set with the VtkPiecewiseFunction() and AddPoint() functions.

3.7 Volume Properties

[VTKVol] The VtkVolumeProperty() is said to represent the common properties for rendering any volume. The three transfer functions are assigned to the volume properties through the SetColor, SetScalarOpacity, and SetGradientOpacity methods. The shading is set to ShadeOn() for the volume such that the shading calculations are performed on it along with setting Ambience -SetAmbient(), Diffuse – SetDiffuse() and Specular- SetSpecular(). Linear interpolation is usually employed when there is continuous data as it provides smoother transitions and this is set through SetInterpolationTypeToLinear() method.



```
    # Volume property setting
    propVolume = vtk.vtkVolumeProperty()
    propVolume.SetColor(ctfun2)
    propVolume.SetScalarOpacity(tfun2)
    propVolume.SetScalarOpacity(1, volumeScalarOpacity2)
    propVolume.SetGradientOpacity(1, volumeGradientOpacity2)
    propVolume.ShadeOn()
    propVolume.SetAmbient(1.0)
    propVolume.SetDiffuse(0.7)
    propVolume.SetSpecular(0.5)
    propVolume.SetInterpolationTypeToLinear()

```

3.8 Volume Rendering

The VtkFixedPointVolumeRayCastMapper() method is used in place of VtkVolumeRayCastMapper() due to version challenges but performs the same way. The VolumeProperty is set by calling the propVolume as per the above code. Actors are added to the VtkVolume() method and thereby called back. Renderers VtkRenderer() and VtkInteractiveRender() and VtkInteractiveRenderWindow() are initialized and associated with the volume that requires rendering [VTKVol]. Finally, a neat black background is set to view a clear volume rendering upon an interactive window where user can drag the image to rotate and view at desired angles.

Chapter 4 Results

The snapshot of a 3D volume rendering of the human brain with the overlay of dummy gene expression information is shown below;

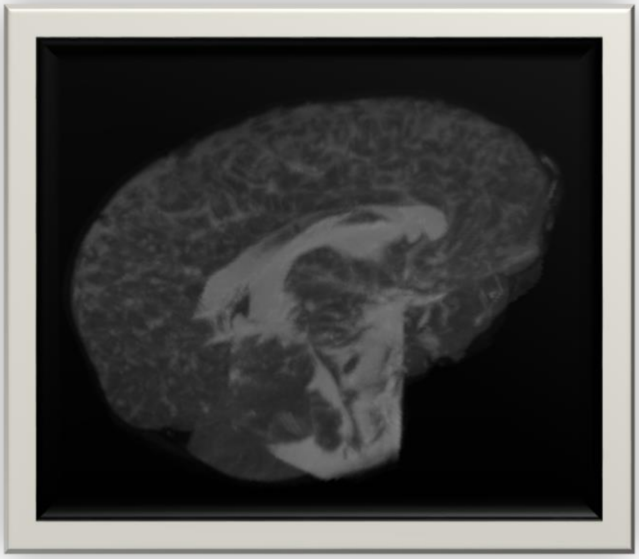


Figure 1. MRI Volume Rendering

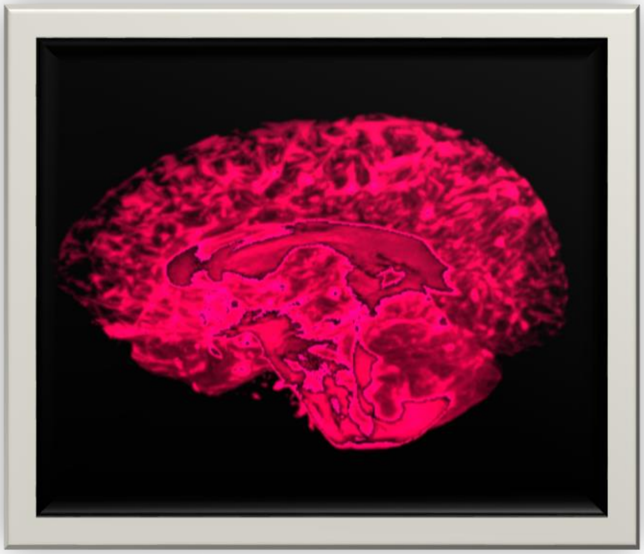


Figure 2. MRI + Gene Expression Volume Rendering (Angle 1)

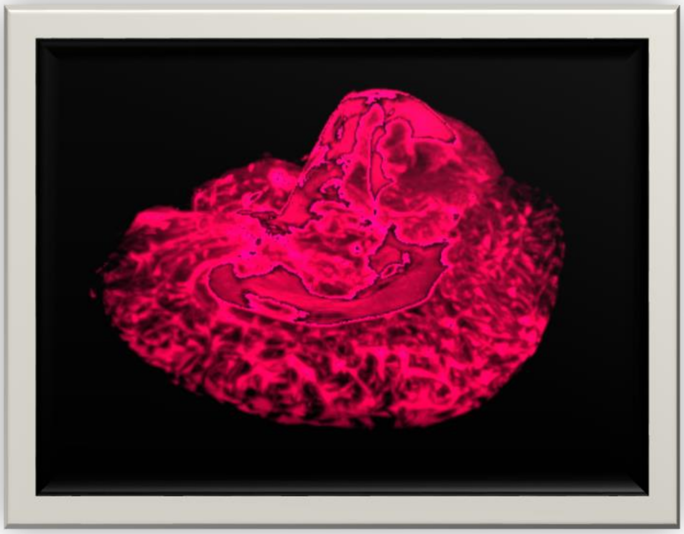


Figure 3. MRI + Gene Expression Volume Rendering (Angle 2)

The 1st figure shows only the MRI image volume rendering. The figures 2nd and 3rd represent the combined volume rendering of both MRI brain image and the dummy gene expression information which will be replaced with actual values in the future work.

Chapter 5 Discussions

There are numerous useful interactive brain atlases created and available online. This tool manipulates the Allen Brain Atlas data and renders through VTK and Python by overlaying the gene expression information. [Dewarrn] is one similar project that deals with volume rendering of the brain using VTK but uses binary masking and complex masking for 2D and 3D volumes respectively. There are other useful alternatives for the same idea such as 3DSlicer and MRIcroGL that offer the same functionalities but in a better way. In spite of quick rendering of the volume, the inputs fed here are just images either 2D or 3D unlike array rendering that is done in brain gene visualization project. [Robert] Along with his team performed a stereoscopic volume rendering of the Brain Atlas that seems to be effective than the other alternative publicly available online datasets as it has the following add-ons -1) The data is volume rendered in real time 2) Stereoscopic projection is directly supported in the atlas, 3) A co-registered MRI is simultaneously rendered, and interactively tunable 4) Text-to-speech translation is incorporated into the software giving verbal auditory descriptions of the anatomical regions. In Brain Gene Visualization model that is proposed in this project, the gene values will be replaced with actual array data in further work.

References

[Brain]http://www.brain-map.org/

[VTK] http://www.vtk.org/

[ABADV]http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4033128/

[VTKVol]http://www8.cs.umu.se/kurser/TDBD13/VT00/vtk-material/manhtml/vtkVolumeProperty.html

[Elvins]https://www.cs.duke.edu/courses/spring03/cps296.8/papers/elvins92surveyVolumeVisualization.pdf

[Robert]http://www.brains-minds-media.org/archive/1512

[Dewarrn]https://bitbucket.org/dewarrn1/vtk_render_demo