VTK Homework Report

Jintong Han

jintong.han@vanderbilt.edu

1 Approach

In this assignment, we will continue to work on the DTI project, this time we will learn how to retrieve package and use different class in the VTK API to solve some actual problem that we will met in the DTI project. This time we will follow four steps in the instruction to complete the VTK part assignment.

Firstly, we have to visualize the vector image in VTK. Since in the part one we have already made the FA image that we have computed in the ITK assignment. I would like to continue analyze some scalar summary metric we have compute the brain's white matter tracts form the image. In VTK, the Filter for calculating the normal vector in VTK is vtkPolyDataNormals(). This type of calculation is performed on vtkPolyData data whose units are triangles or polygons. Since the normal vector is divided into a point normal vector and a unit normal vector, the normal vector type that needs to be calculated can be set through the functions SetComputeCellNormals() and SetComputePointNormals(). Before I use this VTK package to deal with the image, I followed the methods that we have already learned in course to ITK image to VTK image filter to get a VTK image. Then, like what we have done in course to get a renderer and put it in window and interactor to visualize it.

2

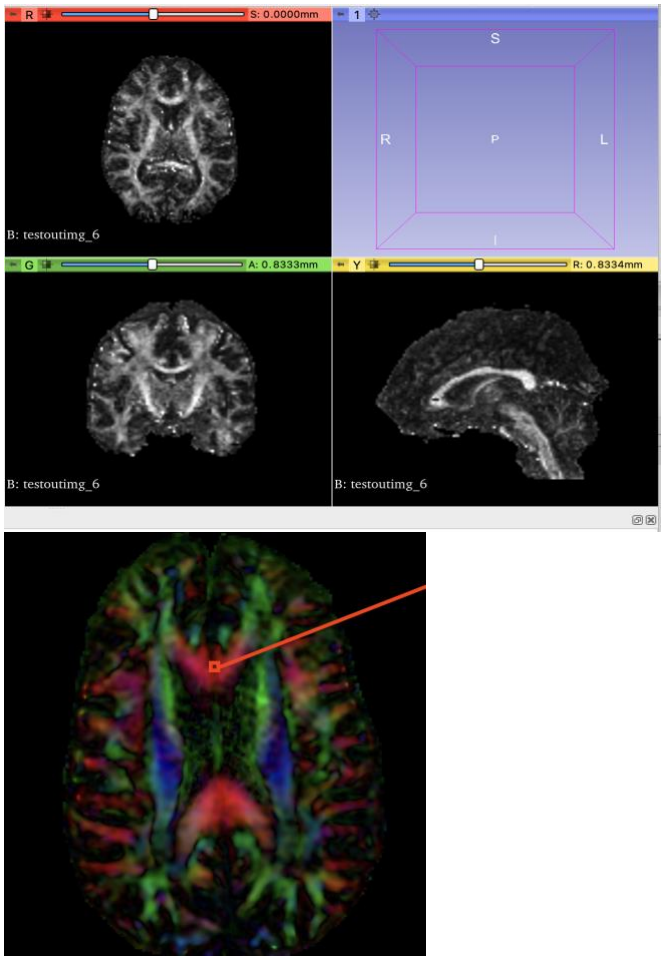

Fig. 1. DTI original picture

Secondly, we implement a timer such that we gradually grow a tract out of our initial seed voxel v1. At the first iteration, we have just the seed voxel. Next iteration, we add the two next points that get added to the visit list and then followed the step 6 of the ITK part. Then, using an iterator that traverses the entire image, the image region iterator, I computed the principal eigenvector image from the diffusion tensor image. Because the output should be an image with a value of 0 for all unvisited voxels and a value of 1 for all visited voxels. I devised a method for determining whether or not each visiting voxel met the stop condition. Nothing should be done if the voxel met any stop conditions. Otherwise, identify the principal eigenvector d(x) at voxel x and add

x+d(X)*delta and x-d(X)*delta to the list of voxels to visit. Since I can choose a smaller step size delta as my points, it can be in continuous 3D space rather than discrete voxel coordinates. Use a single poly data object to represent the entire tract. I created a second renderer to visualize both the track and the FA Image in the same window. Finally, I added an observer to the interactor and created a VTK Callback Command with a timer so that a new point location would be changed every 100 milliseconds. As a result, a window appeared in which a track from the hard-coded seed appeared to grow iteratively to the right of the FA Image. When the visit list is empty, come to a halt.

Thirdly, extend to allow the use of the label input instead of single voxel input. Maintain the previous instruction's value for the other parameter. To verify the technique, I open the FA image in ITKSnap and label the corpus callosum in the midsagittal slice. The following diagram illustrates the results in the 3D Slicer.

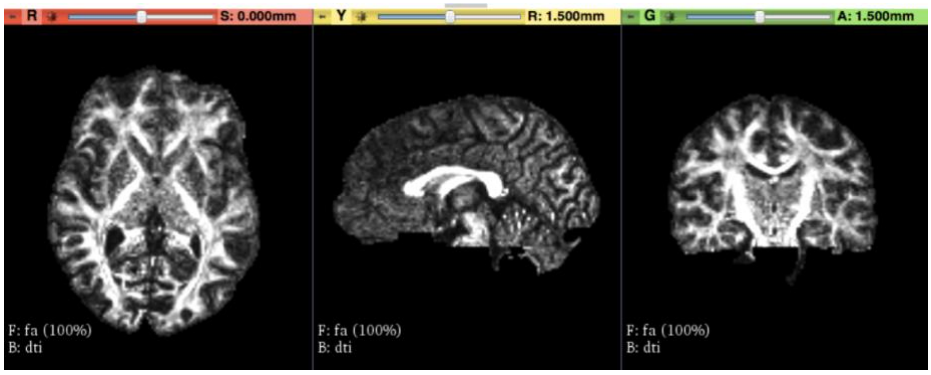

Fig. 2. Experiment with a variety of different min fa values to observe the difference in the resulting FA image, when min fa equals 0.25, max step equals 1000, and delta equals 1.5.

Finally, implement a mouse callback such that each time you click somewhere on the screen, a new tract is created and grown along with the timer, as well as a Sphere Source representing the seed location. Make each tract a different color to be able to distinguish them. To accomplish this, create a click list that will store a vector of index lists when the mouse is clicked on the FA image. In response to VTK mouse and keyboard events, the observer mode is used, which requires first registering the observer. When an event occurs, the member function of

the observer is automatically invoked. Create a callback function and an observer class, then implement the callback function. Register the callback object when the window object is initialized. Additionally, a new VTK PolyPointSource object was created for the new track via a mouse click event. This then activated the timer callback for the new track.

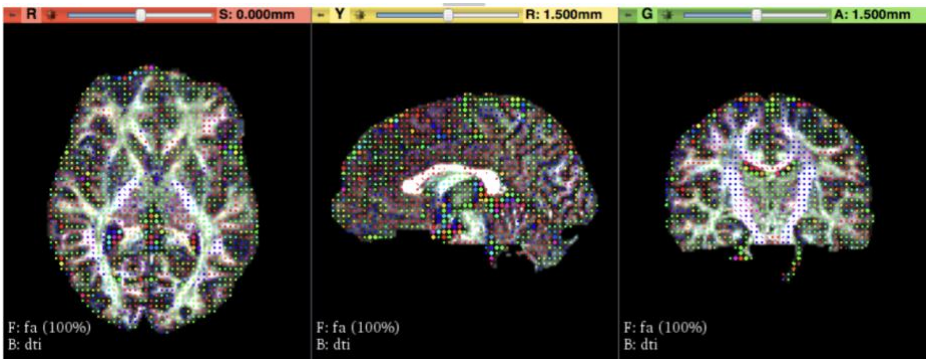

Fig. 3. Experiment with a variety of different min fa values to observe the difference in the resulting FA image. Slicer display the colorful slice.

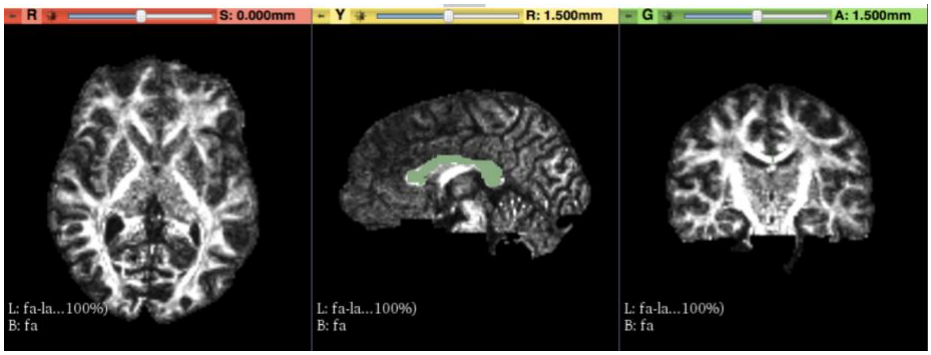

Fig. 4. Experiment with a variety of different min fa values to observe the difference in the resulting FA image. Sign the C shape area in graph.

2 Usage Instructions.

To run the code, you can type the command line like below in the command line.

./VTKasn ../Data/DTIBranin.nrrd

- VTKasn: is the VTKasn.cxx file which is the coding that I made.
- DTIBranin.nrrd: it is the input diffusion tensor image that we download from the 3d slicer.

References

1. https://itk.org/Doxygen/html/search.php?query=computeeigen+analysis
2. https://radiopaedia.org/articles/diffusion-tensor-imaging-and-fibre-tractography?lang=us
3. http://dmri.slicer.org/tutorials/Slicer-4.8/DiffusionMRIAnalysis.pdf