Fiber Tracking Pipeline – Command Line Version Documentation

Setting up the Virtual Machine

1. Install Virtual Box
2. Create a new virtual machine with Virtual Box with no attached storage. The machine should be a Red Hat 64 bit machine
3. Attach disks *dtipipeline_root.vdi* and *dtipipeline_processing.vdi*
4. Configure the NAT setup to allow an ssh connection to the machine

Note: The *dtipipeline_processing.vdi* disk is empty but it is setup to dynamically grow to a max capacity of 250G. You must make sure that the disk can grow to this amount of space but placing it in an area that has at least 250G free. If you need more than 250G of space you can enlarge the disk. Optionally, if the disk is too large you can detach this disk and add a new one of your own size.

Running the pipeline

Basic steps

1. ssh to the machine with username *erin.* The password is *epipeline*. The root user password is *epipelinenow*
2. The command to start processing is *fbtpipeline*
3. Processing input **and** output should be placed in */home/erin/processing/.* This is a symlink to the mounted mass storage disk *dtipipeline_processing.vdi.* The root disk has only ~100 mb free.

Detailed Information

Overview

The pipeline proceeds in two steps. In the first step scripts *ftpartial* and *ftprep* are called to prepare the inputs. Once prepared *ft* is launched which does the actual fiber tracking. This is a very time intensive task and by our current benchmarks can take up to anywhere from eight to sixteen hours using sixteen threads on an Intel Xeon 2.4ghz machine.

However, before *ftpartial* and *ftprep* are called, the md5sums of the inputs are recorded for provenance and the arguments given to the pipeline are recorded as well. You will find the record in a file *fbtlog*, see the **Logging** section below. *After ftprep* has finished running *fbtpipeline* does a rudimentary verification that certain outputs have been created successfully. *Fbtpipeline* does this check for *ft as well.*

Input

The input consists of twelve ANALYZE images.

1. A segmentation image containing gray matter (GM) and white matter (WM) voxels
2. Seperate 3D images for the X,Y,Z components of the principle eigenvector
3. An exclude mask for cortical fiber tracking
4. An exclude mask for subcortical fiber tracking
5. An image containing the cocomac cortical regions

6. A mask defining the left side of standard space
7. A mask defining the right side of standard space
8. An image containing the cocomac subcortical regions
9. An FA image
10. An MD image

Output

Where data is output depends on the value of the *–dst* switch. The output directory should preferably not already exist since many files will be placed in that directory by the analysis. The chosen directory also defines the prefix of output files as well. For instance, if *–dst* is */home/processing/xnat_output/es* then all output will be placed in this directory and most files will be prefixed with *es_*. In this folder, after the analysis is complete you will find files with the prefix such as *es_002_003_AvgFlow.txt*, *es_002_003_AvgLength.txt*, etc.

Logging

Inside the output directory there is a file *fbtlog* which contains a record of the arguments given to the pipeline, the checksums of the images provided and the output of the pipeline.

Defining thread usage

This pipeline makes use of a parallel *ft* version. By default 16 threads are used. Define the number you want *ft* to use with *–threads. --threads 4* would allow *ft* to use four threads.

<u>Connection Matrices</u>

The connection matrices exist in the output folder:

*_ConnectionCapacityMatrix_BrainVoxelsNormalized.csv
*_ConnectionCapacityMatrix_CapacityNormalized.csv
*_ConnectionCapacityMatrix.csv
*_ConnectionDistanceMatrix.csv
*_ConnectionDistanceMatrix_DistanceNormalized.csv

They correspond to the following descriptions respectively

Full-brain connectivity matrix for capacity (i.e. flow/strength) -- Normalized by the total number of brain voxels
Full-brain connectivity matrix for capacity (i.e. flow/strength) -- Normalized by the 99th percentile (values between 0 and 1)
Full-brain connectivity matrix for capacity (i.e. flow/strength)
Full-brain connectivity matrix for distance
Full-brain connectivity matrix for distance -- Normalized by the 99th percentile (values between 0 and 1))

Example: Starting the pipeline

```
[erin@pipeline fj]$ pwd
/home/erin/processing/fj
[erin@pipeline fj]$ ll ft\ inputs/
total 29800
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 01toDTI.hdr
-rw-r--r--. 1 erin erin 1124352 Mar 23 14:47 01toDTI.img
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 02toDTI.hdr
-rw-r--r--. 1 erin erin 1124352 Mar 23 14:47 02toDTI.img
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 03toDTI.hdr
-rw-r--r--. 1 erin erin 1124352 Mar 23 14:47 03toDTI.img
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 04toDTI.hdr
-rw-r--r--. 1 erin erin 1124352 Mar 23 14:47 04toDTI.img
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 05toDTI.hdr
-rw-r--r--. 1 erin erin 1124352 Mar 23 14:47 05toDTI.img
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 06toDTI.hdr
-rw-r--r--. 1 erin erin 1124352 Mar 23 14:47 06toDTI.img
-rw-r--r--. 1 erin erin       1 Mar 23 14:47 BERLIN_ES.txt
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 EV1X.hdr
-rw-r--r--. 1 erin erin 4497408 Mar 23 14:47 EV1X.img
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 EV1Y.hdr
-rw-r--r--. 1 erin erin 4497408 Mar 23 14:47 EV1Y.img
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 EV1Z.hdr
-rw-r--r--. 1 erin erin 4497408 Mar 23 14:47 EV1Z.img
-rw-r--r--. 1 erin erin     348 Mar 25 21:53 FA.hdr
-rw-r--r--. 1 erin erin 4497408 Mar 25 21:53 FA.img
-rw-r--r--. 1 erin erin     348 Mar 25 21:53 MD.hdr
-rw-r--r--. 1 erin erin 4497408 Mar 25 21:53 MD.img
-rw-r--r--. 1 erin erin     348 Mar 23 14:47 SegToDTI.hdr
-rw-r--r--. 1 erin erin 1124352 Mar 23 14:47 SegToDTI.img
[erin@pipeline fj]$ fbtpipeline --dst "out put"/es --seg "ft
inputs"/SegToDTI.hdr --pev "ft inputs"/EV1X.hdr "ft inputs"/EV1Y.hdr
"ft inputs"/EV1Z.hdr --img "ft inputs"/01toDT\
I.hdr "ft inputs"/02toDTI.hdr "ft inputs"/03toDTI.hdr "ft
inputs"/04toDTI.hdr "ft inputs"/05toDTI.hdr "ft inputs"/06toDTI.hdr
--fa "ft inputs"/FA.hdr --md "ft inputs"/MD.hdr -\
-wm 3 --gm 4 --threads 15&

[erin@pipeline fj]$ tail -f out\ put/es/fbtlog
Masking values: 51, 53,
Writing masked image: es_seg_to_dti_wmonly_scm_masked
Image : es_seg_to_dti_wmonly_scm_rightonly
Mask  : es_subcortical_to_dti
...
...
```

Caveats

You will see output resembling the following below when *ftprep* is run. This is normal and can be ignored for the following files.

_t1_to_dti.img
_cocomac_cortical_to_t1.img
_mnim_to_t1.img
_mni_to_t1.img
_subcortical_to_t1.img

```
Error reading input!

itk::ImageFileReaderException (0x93c60e0)
Location: "void itk::ImageFileReader<TOutputImage,
ConvertPixelTraits>::GenerateOutputInformation() [with TOutputImage =
itk::Image<double, 3u>, ConvertPixelTraits = itk::Defa\
ultConvertPixelTraits<double>]"
File: /home116/egibson/openSuSE9.0/InsightToolkit-
3.12.0/Code/IO/itkImageFileReader.txx
Line: 144
Description:  Could not create IO object for file es_t1_to_dti.img
The file doesn't exist.
Filename = es_t1_to_dti.img
```