

Baby Brain Toolkit

Fbrain ERC project: Computational Anatomy of Fetal Brain

February 17, 2011

Contents

1	Introduction	1
1.1	Copyright	1
1.2	Installation	2
1.2.1	Dependencies	2
1.2.2	Download and compile the BTK sources	2
2	Applications	3
2.1	Denoising	3
2.2	Anatomical reconstruction	3
2.3	Tractography	3
3	Utilities	6

1 Introduction

BTK stands for Baby Brain Toolkit. This toolkit is developed in the context of the Fbrain ERC project: “Computational Anatomy of Fetal Brain”¹. Studies about brain maturation aim at providing a better understanding of brain development and links between brain changes and cognitive development. Such studies are of great interest for diagnosis help and clinical course of development and treatment of illnesses. Several teams have begun to make 3D maps of developing brain structures from children to young adults. However, working out the development of fetal and neonatal brain remains an open issue. This project aims at jumping over several theoretical and practical barriers and at going beyond the formal description of the brain maturation thanks to the development of a realistic numerical model of brain aging.

1.1 Copyright

This software is governed by the CeCILL-B license under French law and abiding by the rules of distribution of free software. You can use, modify and/ or redistribute the software under the terms of the CeCILL-B license as circulated by CEA, CNRS and INRIA at the following URL ”<http://www.cecill.info>”.

As a counterpart to the access to the source code and rights to copy, modify and redistribute granted by the license, users are provided only with a limited warranty and the software’s author, the holder of the economic rights, and the successive licensors have only limited liability.

In this respect, the user’s attention is drawn to the risks associated with loading, using, modifying and/or developing or reproducing the software by the user in light of its specific status of free software, that may mean that it is complicated to manipulate, and that also therefore means that it is reserved for developers and experienced professionals having in-depth computer knowledge. Users are therefore encouraged to load and test the software’s suitability as regards their requirements in conditions enabling the security of their systems and/or data to be ensured and, more generally, to use and operate it in the same conditions as regards security.

¹<http://lsiit-miv.u-strasbg.fr/miv/index.php?contenu=erc>

1.2 Installation

1.2.1 Dependencies

Baby Brain Toolkit (BTK) depends on:

- CMake (www.cmake.org): this library can be installed for debian-based distribution using the following command line : `apt-get install cmake cmake-curses-gui`
- Insight Toolkit (ITK) version 3.20 (it can be download at www.itk.org). Extract this file (the directory InsightToolkit-3.20 is created). Make another directory where all the build files will be placed, e.g. InsightToolkit-3.20-build:

```
mkdir InsightToolkit-3.20-build
cd InsightToolkit-3.20-build
cmake ../InsightToolkit-3.20/
```

This will bring up the CMake configuration screen. Press [c] for configure and then use [t] to toggle the advanced mode. Make the following changes:

```
BUILD_TYPE = Release
ITK_USE_OPTIMIZED_REGISTRATION = ON
ITK_USE_REVIEW = ON
```

Then press [c] to configure and [g] to generate the make file. Finally, type `make` at the prompt to obtain the final build of ITK.

- Tclap library: this library can be installed for debian-based distribution using the following command line : `apt-get install libtclap-dev`
- OpenMP library: this library can be installed for debian-based distribution using the following command line : `apt-get install libgomp1`
- VTK library: this library can be installed for debian-based distribution using the following command line : `apt-get install libvtk5-dev`

1.2.2 Download and compile the BTK sources

- Install Git: this library can be installed for debian-based distribution using the following command line : `apt-get install git-core`
- Get the BTK sources: `git clone https://github.com/rousseau/fbrain.git`
- Then:

```
mkdir fbrain-build
cd fbrain-build
cmake ../fbrain
make
```

Most of the programs of the BTK suite use the OpenMP library for multi-threading purpose. The number of cores used can be tuned using the following command line (in this example, 4 cores will be used): `export OMP_NUM_THREADS=4`

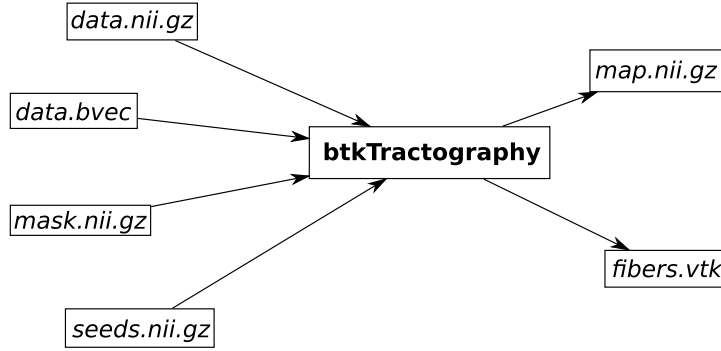


Figure 1: Standard pipeline of the btkTractography program.

2 Applications

2.1 Denoising

btkNLMDenoising This program applies a non-local mean filter to a 3D image for denoising purpose. Usage: `-i input_image_filename -o output_image_filename`. The best results are usually obtained by using a mask (or a padding value).

btkNLMDenoising4DImage This program applies a non-local mean filter to each 3D image of a 4D image, for denoising purpose. Usage: `-i input_image_filename -o output_image_filename`. The best results are usually obtained by using a mask (or a padding value).

2.2 Anatomical reconstruction

btkImageReconstruction This program allows to obtain a high-resolution image from a set of low-resolution images, typically axial, coronal, and sagittal acquisitions [2].

Minimal usage: `btkImageReconstruction -i image1 ... -i imageN -o output --box`.

Recommended usage: `btkImageReconstruction -i image1 ... -i imageN -m mask1 ... -m maskN -o output --mask`. The use of a mask provide better results since it allows an accurately estimation of the initial transform, and constrains the registration to the region of interest.

The full list of optional parameters of the method can be obtained by `btkImageReconstruction --help`

2.3 Tractography

Standard usage

Suppose you want to perform a tractography on a diffusion weighted MRI dataset. You should have a dwi image, the corresponding gradient vectors' coordinates, a mask of the brain white matter and a label image of the seeds. Assume this data is stored in files named respectively for instance `data.nii.gz`, `data.bvec`, `mask.nii.gz` and `seeds.nii.gz`. The tractography is accomplished by the command below.

```
btkTractography -d data.nii.gz -g data.bvec -m mask.nii.gz -l seeds.nii.gz
```

When the program terminates its task, the probability connection map and the fibers estimation are saved in files respectively named `map.nii.gz` and `fibers.vtk`. The connection map is a volume image of probability intensities (i.e. intensities between 0 and 1) with the same origin, orientation and spacing as the diffusion weighted image. The fibers are polygonal data of VTK library in world coordinates. The standard pipeline of the program is shown in Fig. 1.

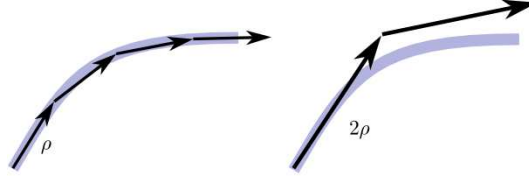


Figure 2: Effect of the step size option on a particle's trajectory. With a large step size (right), the particle may overshoot the trajectory of the ground truth.

Advanced usage

In addition to standard arguments of `btkTractography` program, there are some other parameters that let you to alter algorithm's behaviour. These options can be classified into three groups : model's options, constraints on trajectory and filter's options. The first group options allow you to tweak the model (for more details about it, please refer to [1]). The second group options let you to control the particle's trajectory. These options provide prior informations to the algorithm. The last group options are dedicated to the particle filter control.

Since the default parameters values may work in the most of cases, they are optional. A list is of optional features is available by using the command

```
btkTractography --help
```

and program's arguments are much more described below.

Model's order

The model's order (i.e. the spherical harmonics' order) can be specified by the option

```
--model_order <order> ,
```

for $\text{order} \in \{2, 4, 6, 8\}$. The default value is 4. For more details, please refer to [1].

Model's regularization

A Laplace-Beltrami regularization coefficient is used to assume a better estimation of the model. This coefficient can be manually modified by the option

```
--model_regularization <coefficient> ,
```

for $\text{coefficient} \in \mathbb{R}$. The default value is set as 0.006. For more details, please refer to [1].

Displacement step size

The displacement step size of a moving particle can be adjusted as you want by using the option

```
--step_size <length> ,
```

where $\text{length} \in \mathbb{R}_+^*$. Note that this option is expressed in the voxel unit. The default value is fixed at 0.5 voxel. By setting a big step size, the particles will move quickly. So the bigger is the step, the faster the algorithm will finish, but as shown by Fig. 2, some informations may be missed and the particle's trajectories may overshoot the ground truth, resulting in a bad estimation.

Angular threshold

An angular threshold prevent a particle to return back. This option has to be expressed in radian and can be set by

```
--angular_threshold <angle> ,
```

where $\text{angle} \in]0, 2\pi[$. The default value is set as a $\frac{\pi}{3}$ angle. As illustrated in two dimensions in Fig. 3, an angle threshold is used to define an allowed area for successive sampled directions. This can be seen as a global curvature parameters on trajectories. A small angle defines trajectories with a small curvature. This is a prior information on ground truth trajectory.

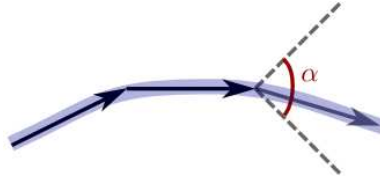


Figure 3: An angle threshold allows the algorithm to sample successive direction only in the cone defined by this angle. This illustration show the principle in two dimensions.

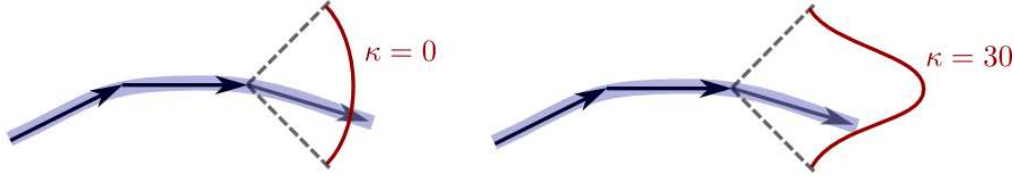


Figure 4: Local effect of rigidity parameter on a particle's trajectory. This parameter helps to “attract” the current displacement vector in the direction of the previous displacement vector of a particle. It correspond to a concentration paramter of a von Mise-Fisher density probability used in the prior density of the system. For instance, a rigidity of 0 leads to an equiprobable distribution, whereas a rigidity tending to infinity leads to a distribution focused on a point.

Rigidity

The rigidity option controls how much you want the particles to have straight trajectory. You can adjust it by

```
--curve_constraint <rigidity> ,
```

where $\text{rigidity} \in \mathbb{R}_+^*$. The default value is fixed at 30. This value correspond to a concentration parameter of a von Mises-Fisher density probability used in the prior density of the system. As Fig. 4 illustrates locally in two dimensions, a high value leads to a straight trajectory.

Number of particles

The number of particles in the system is set by the option

```
--number_of_particles <number> ,
```

where $\text{number} \in \mathbb{N}^*$. By default, the algorithm will use 1000 particles. A poor number of particles leads to a short computation time and a poor estimation. A large number of particles leads to a long computation time and a good estimation. In general, the default number of particles is a good compromise between computation time and estimation.

Resampling threshold

This option modify the resampling threshold of the system. When the number of effective particles in the system falls below this resampling threshold, the particles are resampled according a multinomial resampling. It can be adjust by

```
--resampling_threshold <percent> ,
```

where $\text{percent} \in [0,1]$ is the percent of minimal effective particles in the system. A low threshold value will result in an inefficient algorithm because the particles with low weight are not are not often eliminated. Conversely, a high threshold value leads to a bad estimation because the search space will not be explored enough.

3 Utilities

btkModifyImageUsingLookUpTable This program modifies one image using a look up table defined in a ascii file (2 columns, one for the original values, one for the final values). Usage: `-i input_image_filename -t input_table_filename -o output_image_filename`

btkNrrdToNifti This program convert an image from Nrrd file (*.nhdr *.nrrd) to a Nifti file (*.nii). The conversion of a DWI image is possible by using the option `-d`. Usage: `-i input.nhdr -o output.nii.gz`. Usage for DWI sequence: `-i input.nhdr -o output.nii.gz -g gradients.bvec -d`.

btkNiftiToNrrd This program convert a diffusion sequence in nifti format² (*.nii, *.nii.gz) to the nrrd format (*.nhdr).

Usage: `-i input.nii -b bvalues.bval -g gradients.bvec -o output.nhdr`

The list of optional parameters can be obtained by `btkNiftiToNrrd --help`

btkReorientImageToStandard Sometimes it is useful to reorient the image to the standard orientation. This is necessary with fetal images since in general the fetus is in a random orientation with respect to the scanner.

Usage: `btkReorientImageToStandard -i image -o output -l landmarks`. **landmarks** is a text file containing the spatial coordinates (in RAS world coordinates) of points defining the left-right and the posterior-anterior directions. This information must be organized as follows:

l_x	l_y	l_z
r_x	r_y	r_z
p_x	p_y	p_z
a_x	a_y	a_z

where the points $l = (l_x, l_y, l_z)$ and $r = (r_x, r_y, r_z)$ define the left \rightarrow right direction, and the points $p = (p_x, p_y, p_z)$ and $a = (a_x, a_y, a_z)$ define the posterior \rightarrow anterior direction. Such file can be easily generated by using Slicer³.

In order to create the landmarks by using Slicer you want to follow the following procedure:

1. Open the high-resolution image by using the *Volume* module.
2. Toggle on the visibility of all slices in the 3D view. This allows to identify the left and right sides of the brain in the 2D views.
3. Place the landmarks l , r , p , and a in this order by using `[p]`.
4. Save the file (*.fcsv) by using the menu File \rightarrow Save.
5. Edit the generated file by using any text editor to remove the additional data about landmarks (i.e. the lines beginning with `#` and text 'around' the coordinates).

btkReorientDiffusionSequenceToStandard Reorients a DW sequence to the standard orientation. This is necessary with fetal images since the fetus is in a random orientation with respect to the scanner. This is particularly important in DWI because colormaps lack of significance, which makes difficult the identification of specific bundles

Usage: `btkReorientDiffusionSequenceToStandard -i image -g gradients.bvec -l landmarks -o output -c gradients_corr.bvec`.

landmarks is a landmarks file obtained as explained above. **gradients_corr.bvec** is a text file containing the corrected gradient table.

²Currently there is no nifti standard for DWI, so DW images are saved as a standard nifti sequence and two text files containing the b-values (.bval) and the gradient directions (.bvec).

³<http://www.slicer.org>



Figure 5: Example of an anatomical reconstruction of a fetal brain by using `btkImageReconstruction`. (a) axial, (b) coronal, and (c) sagittal view.

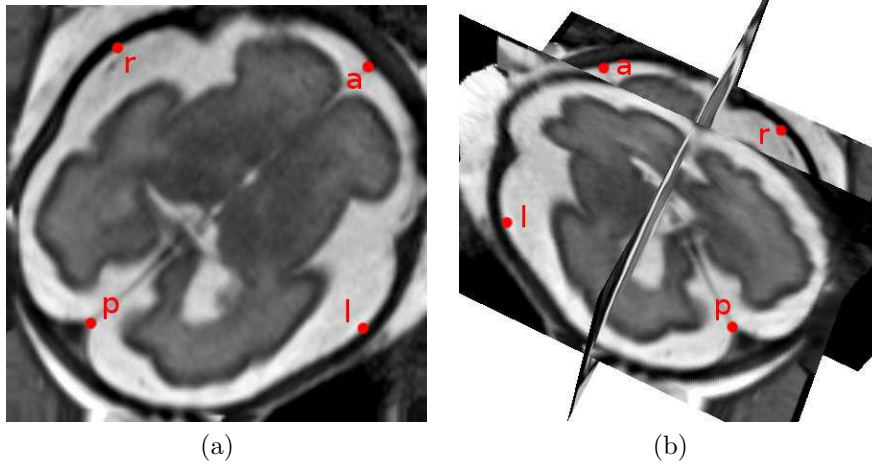


Figure 6: Placement of landmarks by using Slicer. (a) axial slice, (b) 3D view.

Acknowledgment

The research leading to these results has received funding from the European Research Council under the European Communitys Seventh Framework Programme (FP7/2007-2013 Grant Agreement no. 207667).

References

- [1] M. Descoteaux, E. Angelino, S. Fitzgibbons, and R. Deriche. Regularized, fast, and robust analytical q-ball imaging. *Magnetic Resonance in Medicine*, 58(3):497–510, 2007.
- [2] Francois Rousseau, Orit A Glenn, Bistra Iordanova, Claudia Rodriguez-Carranza, Daniel B Vigneron, James A Barkovich, and Colin Studholme. Registration-based approach for reconstruction of high-resolution in utero fetal MR brain images. *Acad Radiol*, 13(9):1072–1081, Sep 2006.