
DASH3R Documentation

Maged Goubran, Edward Ntiri, Melissa Holmes, Hassan Akhavein,

Oct 04, 2019

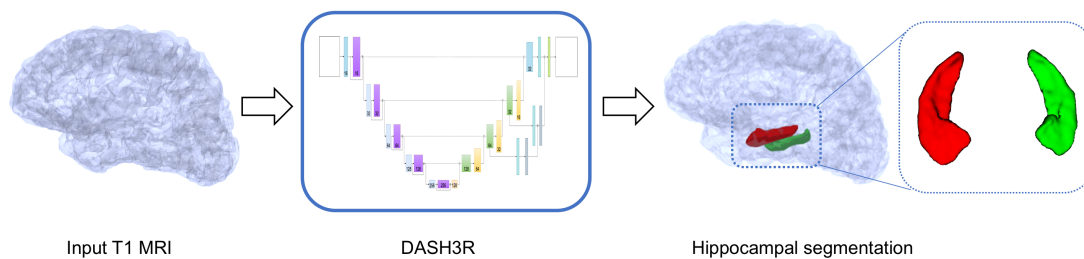
Contents:

1	Before installing DASH3R	3
1.1	Acknowledging this work	3
1.2	Reference	3
1.3	Warranty	3
1.4	License	3
2	Local Install	5
2.1	Python	5
2.2	Installing package and dependencies for DASH3R locally	5
2.3	Download deep models	6
2.4	For tab completion	6
2.5	Updating DASH3R	6
3	Getting started	7
3.1	For GUI	7
3.2	For Command Line	9
3.3	Hippocampal volumes	9
3.4	QC	10
3.5	Logs	10
3.6	File conversion	10
4	Segmentation tutorials	11
4.1	GUI	11
4.2	Command Line	13
5	Issues	15
6	Docker (coming soon)	17
7	Indices and tables	19



Deep Automated Segmentation of the Hippocampus in 3/thRee dimensions (pronounced dasher)

DASH3R is a CNN-based segmentation technique of the whole hippocampus using MRI images from BrainLab. It can deal with brains with extensive atrophy and segments the hippocampi in seconds. It uses a T1-weighted image as the only input and segments both with-skull and skull-stripped images.



Before installing DASH3R

1.1 Acknowledging this work

If you wish to include results generated by DASH3R in a publication, please include a line such as the following:

- Whole hippocampal segmentation was performing using the DASH3R algorithm (dash3r.readthedocs.io) based on a convolutional neural network.

1.2 Reference

- Paper in preparation

1.3 Warranty

The software described in this manual has no warranty, it is provided “as is”. It is your responsibility to validate the behavior of the routines and their accuracy using the source code provided, or to purchase support and warranties from commercial redistributors. Consult the [Mozilla Public License](#) for further details.

1.4 License

DASH3R is free software: you can redistribute it and/or modify it under the terms of the [Mozilla Public License](#) as published by the [Free Software Foundation](#), either version 2 of the License, or (at your option) any later version.

DASH3R is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the [Mozilla Public License](#) for more details. You should have received a copy of [Mozilla Public License](#) along with MRtrix. If not, see <http://mozilla.org/MPL/2.0/>.

2.1 Python

For the main required Python packages (numpy, scipy, etc.) we recommend using [Anaconda for Python 3.6](#)

2.2 Installing package and dependencies for DASH3R locally

1. Clone repository

```
git clone https://github.com/mgoubran/DASH3R.git dasher  
  
(or install zip file and unzip)  
  
cd dasher
```

If you want to create a virtual environment where DASH3R can be run,

```
conda create -n dasher python=3.6 anaconda  
source activate dasher
```

To end the session,

```
source deactivate
```

To remove the environment

```
conda env remove --name dasher
```

2. Install dependencies

```
pip install -e .[option] --process-dependency-links
```

If the computer you are using has a GPU, replace “option” with “dasher_gpu”. Otherwise, replace it with “dasher”

3. Test the installation by running

```
dasher --help
```

To confirm that the command line function works, and

```
dasher
```

To launch the interactive GUI.

2.3 Download deep models

Download the models from [this link](#) and place them in the “models” directory

2.4 For tab completion

```
pip3 install argcomplete  
activate-global-python-argcomplete
```

2.5 Updating DASH3R

To update DASH3R, navigate to the directory where DASH3R was cloned and run

```
git pull  
pip install -e .[{option}] --process-dependency-links
```

where “option” is dependent on whether or not you have a GPU (see package installation steps above)

You can use DASH3R through the graphical user interface (GUI) or command line:

3.1 For GUI

To start the GUI, type

```
dash3r
```

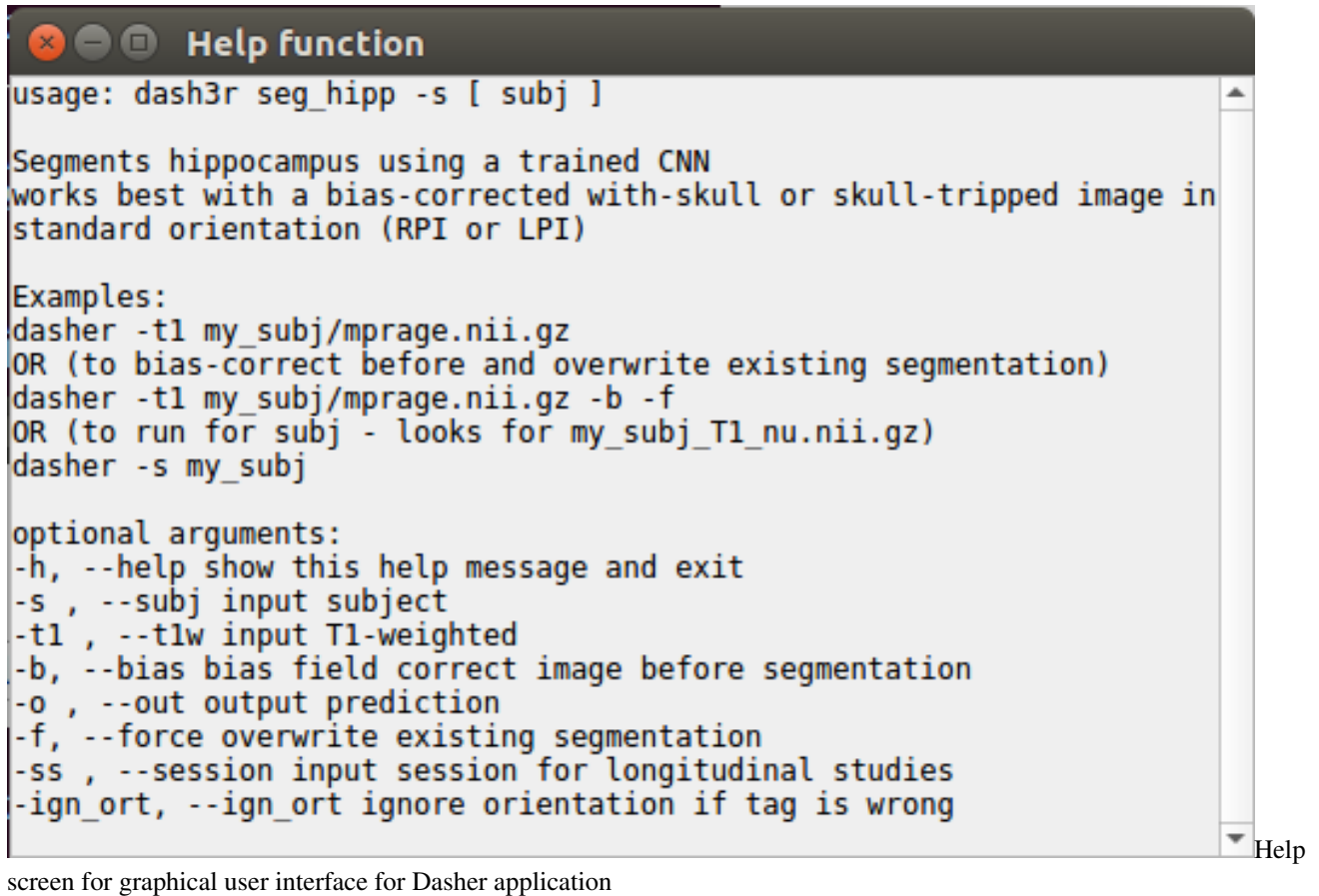
A GUI that looks like the image below should appear. You can hover any of buttons in the GUI to see a brief description of the command.



Graphical user

interface for the Dasher application

You can get the command usage info by click the “Help” box on any of the pop-up windows.



3.2 For Command Line

You can see all the dasher commands by typing either of the following lines:

```
dash3r -h
dash3r --help
```

Once you know the command you want to know from the list, you can see more information about the command. For example, to learn more about `seg_hfb`:

```
dash3r seg_hipp -h
dash3r seg_hipp --help
```

3.3 Hippocampal volumes

To extract hippocampal volumes use the GUI (Stats/Hippocampal Volumes) or command line:

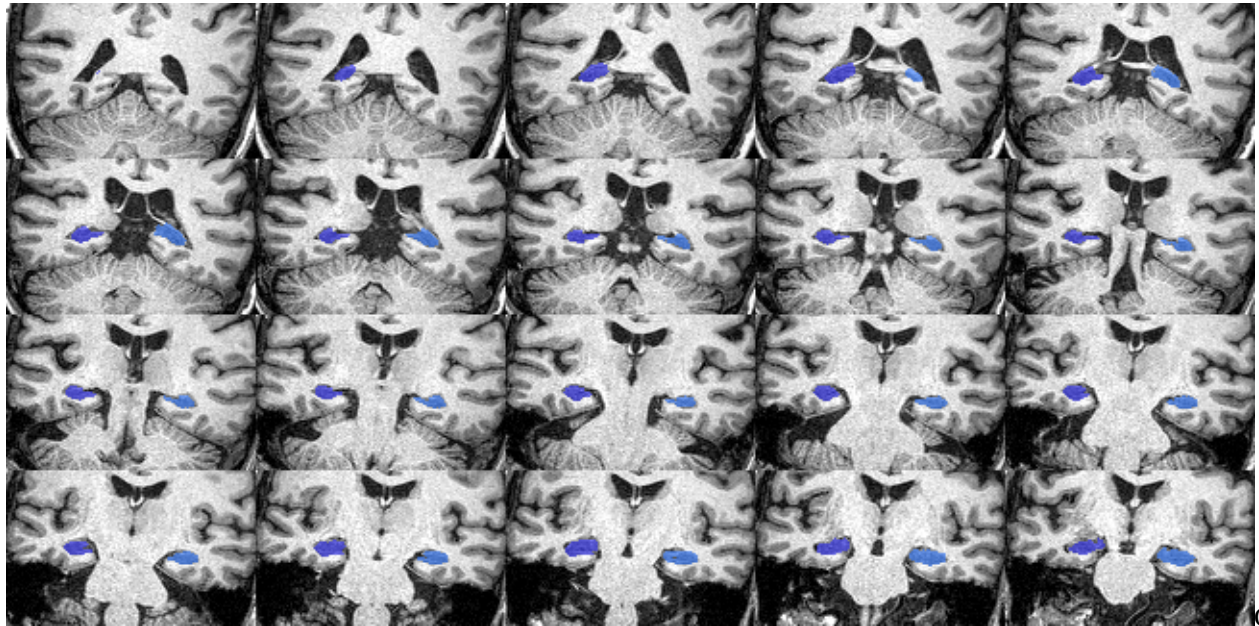
```
dash3r stats_hp -h
```

3.4 QC

QC files are automatically generated in a sub-folder within the subject folder. They are .png images that show a series of slices in the brain to help you quickly evaluate if your command worked successfully, especially if you have run multiple subjects. They can also be created through the GUI or command line:

```
dash3r seg_qc -h
```

The QC image should look like this:



Quality

control image for hippocampus segmentation

3.5 Logs

Log files are automatically generated in a sub-folder within the subject folder. They are .txt files that contain information regarding the command and can be useful if something did not work successfully.

3.6 File conversion

Convert Analyze to Nifti (or vice versa)

```
dash3r filetype
```

Required arguments:

```
-i , --in_img    input image, ex:MM.img
-o , --out_img   output image, ex:MM.nii
```

Example:

```
dash3r filetype --in_img subject_T1.img --out_img subject_T1.nii.gz
```

Segmentation tutorials

4.1 GUI

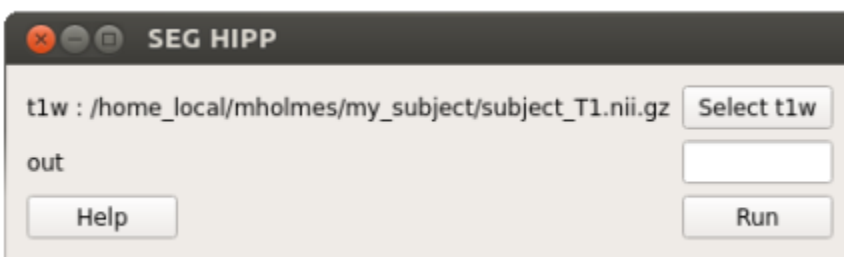
Watch this video tutorial:



ALT TEXT

Or follow the steps below:

After opening the Dasher GUI, click “Hippocampus” under the “Segmentation” tab. Wait for a new pop-up window to appear. The window should look like the image below.



hippocampus pop up window

Click “Select t1w” and chose your T1 image. Click “Run”. Type your desired output name in the “out” box. Your output file will automatically appear in your t1w folder.

4.2 Command Line

```
dash3r seg_hipp
```

Optional arguments:

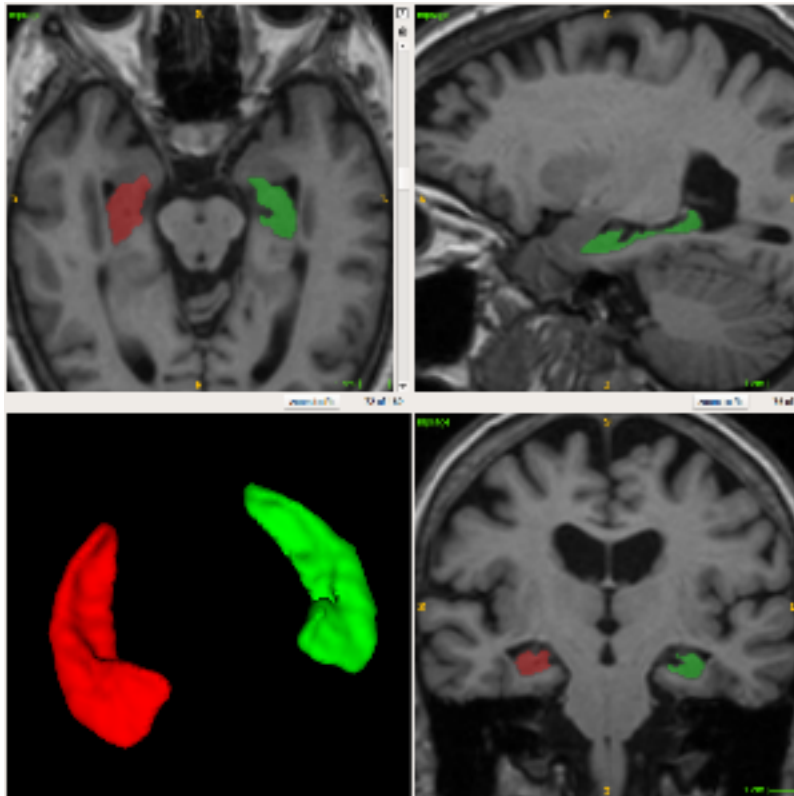
```
-s , --subj      input subject
-t1 , --t1w      input T1-weighted
-b , --bias      bias field correct image before segmentation
-o , --out       output prediction
-f , --force     overwrite existing segmentation
-ss , --session  input session for longitudinal studies
```

Examples:

```
dash3r seg_hipp -s subjectname -b
```

```
dash3r seg_hipp -t1 subject_T1_nu.nii.gz -o subject_hipp.nii.gz
```

The output should look like this.:



hippocampus segmentation

CHAPTER 5

Issues

CHAPTER 6

Docker (coming soon)

CHAPTER 7

Indices and tables

- `genindex`
- `modindex`
- `search`