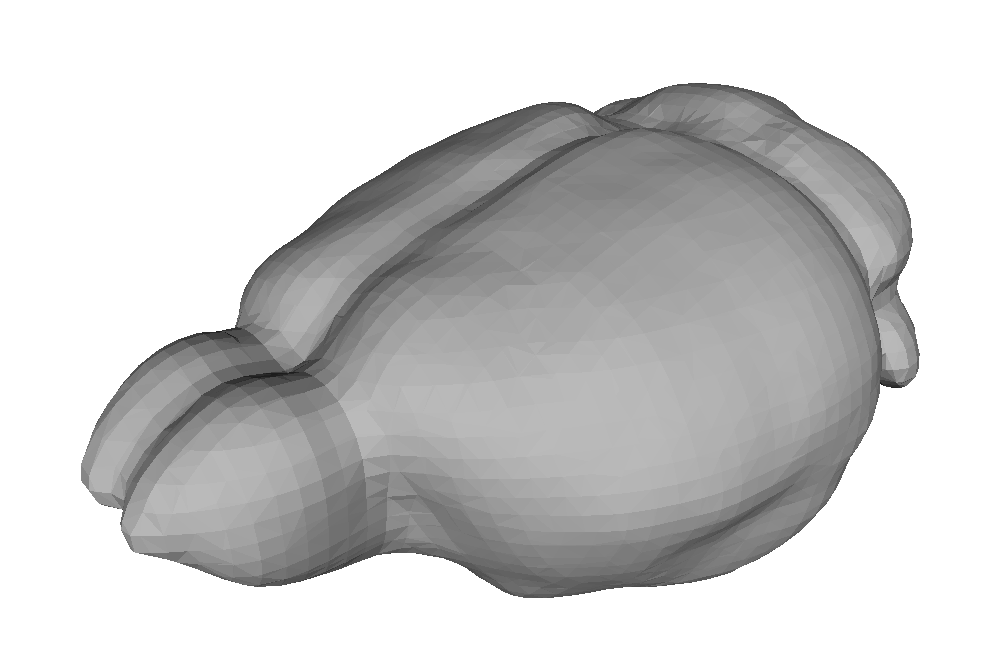
****

**MouseMorph**

An automatic mouse MR image processor

# Tutorial

Nick Powell

nicholas.powell.11@ucl.ac.uk

created Sunday 20 April 2014

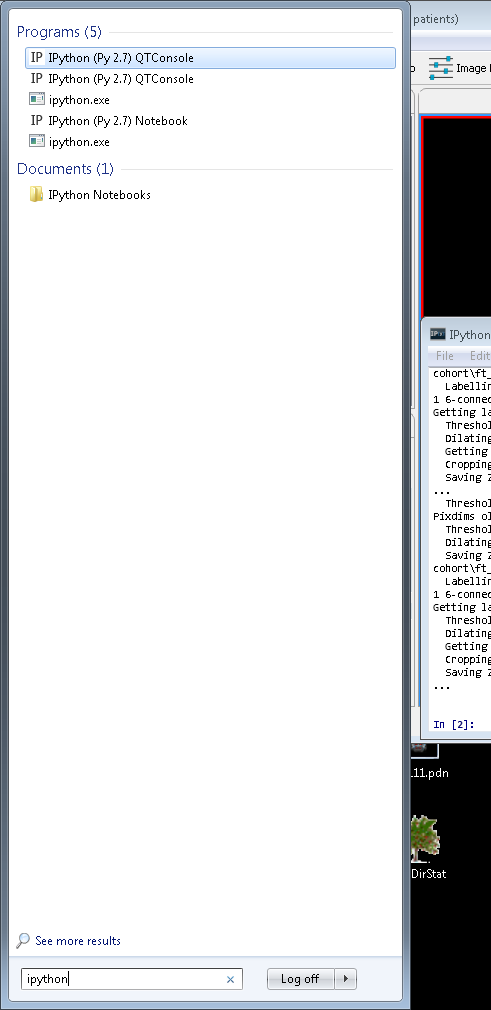
last updated Tuesday 22 April 2014

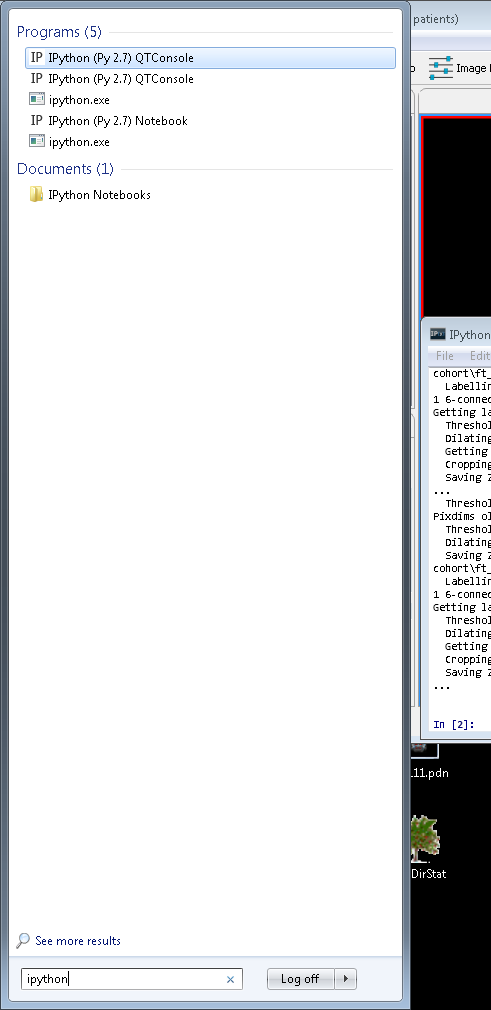
# 1 Python

* *Python* is a programming language which is very easy to understand and read.
* *Anaconda* is a collection of useful Python tools: it is a bit like Matlab.
* *IPython* is just like the Matlab command window. However, instead of there being a code editor included, you edit code in any other text-editing program; IPython is just for running it.

On Megamind:

1. Start > type **IPython** > click on IPython (Py 2.7) QTConsole

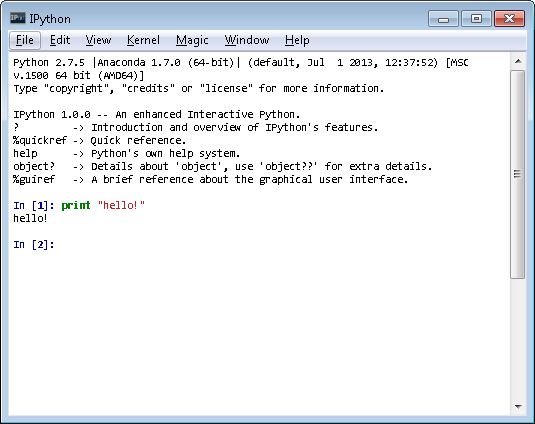




1. Once the window has opened, type:

**print "hello!"**

and hit Enter.



Congratulations, you can program in Python!

# 2 MouseMorph

* *MouseMorph* is for processing large amounts of µMRI mouse brain data.
* It is set up to run on Megamind without too much installation work.

(All of MouseMorph’s required dependencies are already installed on MegaMind.

In Python, a dependency for a program is a collection of functions and scripts (a library) upon which it relies.

For MouseMorph, these include:

* NiBabel for NIfTI-1 file reading and writing
* Numpy for math
* Scipy for N-dimensional image operations
* Matplotlib for plotting graphs
* Pandas for CSV file reading and writing
* Patsy design matrix construction for statistics

MouseMorph also relies upon CMIC’s *NiftyReg* and *NiftySeg* – both are also already installed on Megamind.

If a new installation of NiftyReg or NiftySeg arrives, you can add the installation directory to MouseMorph’s “mm\_config\_user.ini” file.)

* The MouseMorph code is stored in the CABICode folder of CABINet.
* It can be run directly from this folder, or copied to your User area of Megamind (e.g. My Documents) and run from there.

# 3 Extracting ex-vivo brains from multi-subject scans

1. Open IPython
2. Change directory to the MouseMorph directory (as above) by typing:

**cd X:\CABICode\MouseMorph**

and hit Enter. Now you can run the scripts in that directory.

* To run a Python script, use the syntax:

**run script\_name.py –argument1 "parameter1" –arg2 "param2"**

|  |  |  |  |
| --- | --- | --- | --- |
| argument  (short form) | (long form) | example parameter | what it is |
| -i | --input | **"Z:\Projects\TC1xJ20\Ex vivo\ft"** | Input directory of multi-subject images (or a single image) |
| -o | --output | **"Z:\Projects\TC1xJ20\Ex vivo\01\_extract"** | Output directory for saving results |
| -n | --number | **50** | Number of subjects in the study (number of brains in all the images combined) |
| -ds | --downsample | fast: **10**  recommended: **8**  slow: **2** | Factor by which to downsample, to speed up the process. Smaller values will increase accuracy while significantly increasing the time taken to compute the convex hull around each object |
| -md | --maskdir | **"Y:\CABICode\MouseMorph\**  **full\_skull\_extraction\_**  **training\_masks"** | Directory containing existing masks which are *approximately the correct volume* (e.g. previous successful extraction masks). The **50** brains in the **input** data which *best match* these volumes will be extracted. |

1. You set:
   * Input directory (where the multi-subject images are)
   * An output directory to save results, such as in the example above
   * The number of subjects to expect

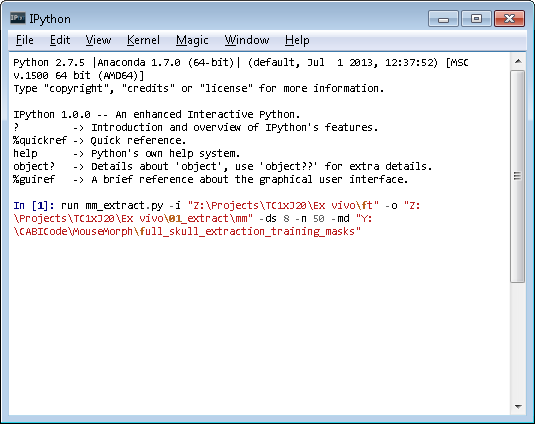
Keep the same as above:

* + the downsampling factor (8, as above)
  + the mask directory (as above)

1. Run MouseMorph **mm\_extract.py** with the correct arguments, which could be (where Z is the drive with data, and Y is the CABINet drive) for example:

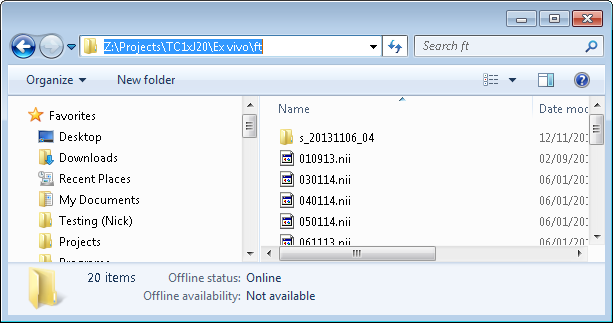
**run mm\_extract.py –i "Z:\Projects\TC1xJ20\Ex vivo\ft** **" –o "Z:\Projects\TC1xJ20\Ex vivo\01\_extract** **" –n 25 –ds 8 –md "Y:\CABICode\MouseMorph\full\_skull\_extraction\_training\_masks"**

and hit Enter.



1. And wait. (This might take a while.) The results will eventually appear in the specified output directory.

* MouseMorph includes some handy pre-existing training data, in the “full\_skull\_extraction\_training\_masks” folder.
* Get directories by copy-pasting from the address bar when in the folder. Add quotes after pasting.

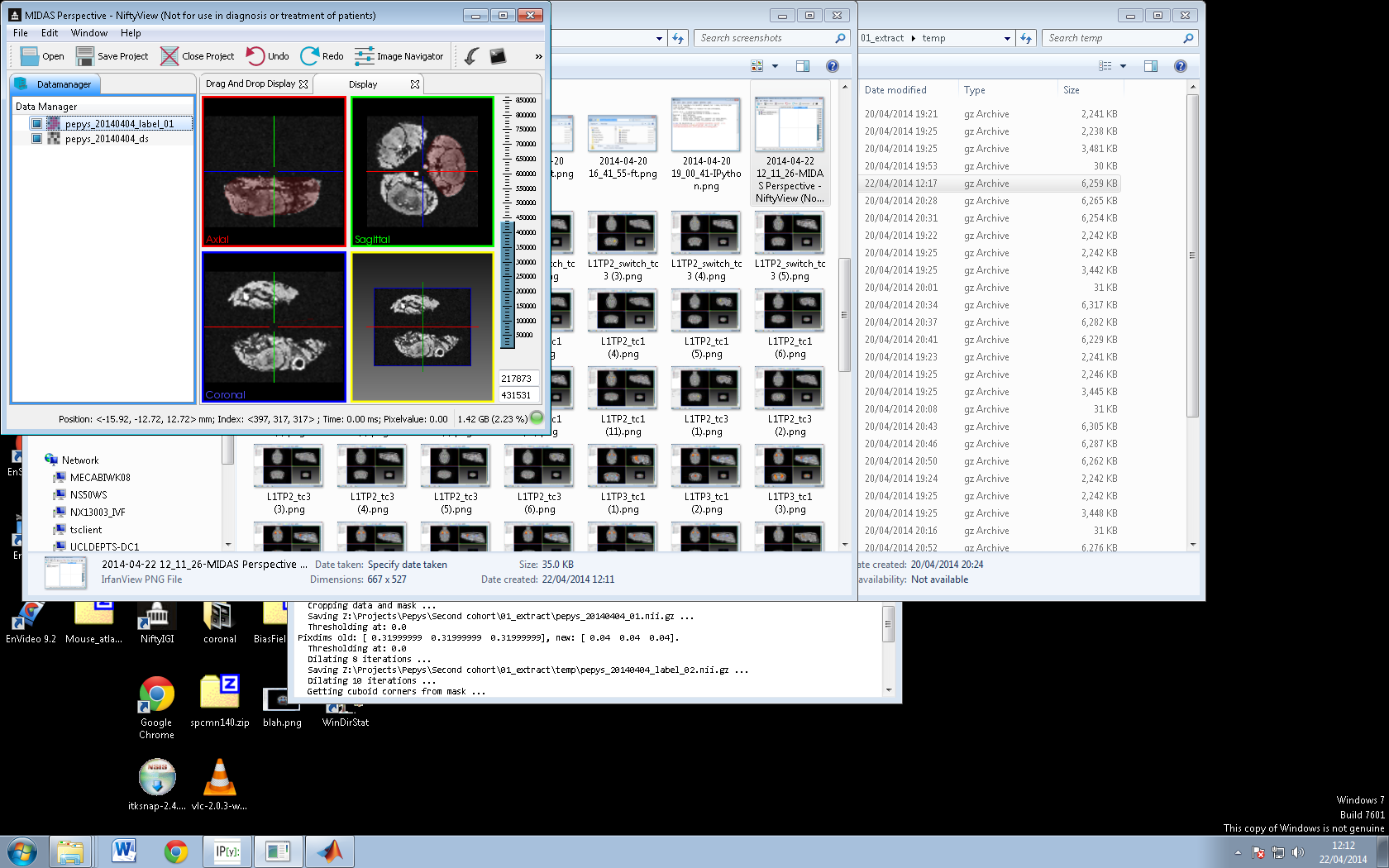


**copy directory text**

## Checking the output

When complete, the output folder should contain:

1. Cropped subjects (check they’re all intact!)
2. a ‘temp’ folder containing:
   1. **name\_ds.nii.gz** (downsampled original image)
   2. **name\_label\_01.nii.gz** (subject 1’s label for overlaying on the downsampled image to identify which subject is which):



# 5 Orienting brains to a standard space

1. On Megamind, open IPython
2. Change directory to the MouseMorph directory (as above) by typing:

**cd X:\CABICode\MouseMorph**

and hit Enter. Now you can run the scripts in that directory.

1. To run the orientation script, use the syntax:

**run mm\_orient.py –i "input directory" –o "output directory" –ds 3 –at "atlas directory"**

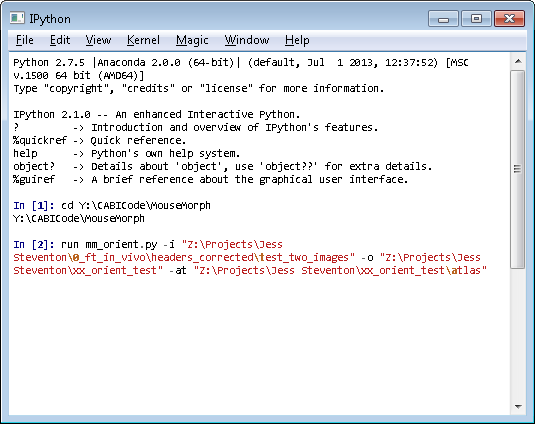
|  |  |  |  |
| --- | --- | --- | --- |
| argument  (short form) | (long form) | example parameter | what it is |
| -i | --input | **"Z:\Projects\TC1xJ20\Ex vivo\ft"** | Input directory of images to be oriented (or a single image) |
| -o | --output | **"Z:\Projects\TC1xJ20\Ex vivo\01\_extract"** | Output directory for saving results |
| -t | --true | N/A | Only include this flag if all the input images are in the same initial orientation. Only one image will be tested; the rest will be oriented in the same way, with some final adjustment per-image. Significantly speeds up the process. |
| -ds | --downsample | fast: **5**  recommended: **3**  slow: **2** (or off) | Optional factor by which to downsample, to speed up the process. Smaller values will slightly increase accuracy while significantly increasing the time taken to register. Larger values will decrease stability. |
| -at | --atlas | **"Z:\Projects\TC1xJ20\Ex vivo\01\_extract\atlas"** | An empty directory in which you will place atlas images, or a directory containing at least one image already in the correct orientation.**\*** |

**\*** The **atlas directory** will be created in the output directory if it doesn’t already exist. If it is empty, you will be asked to manually choose a correctly oriented image during the early stages of the script, and copy it to this directory. If it is not empty, it should contain images in the desired orientation (probably RAS).

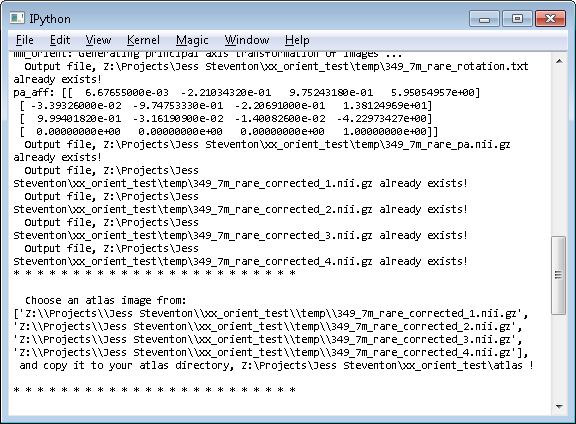
1. Run MouseMorph **mm\_orient.py** with the correct arguments, which could be (where Z is the drive with data) for example:

**run mm\_orient.py –i "** **Z:\Projects\Pepys\Second cohort\extract\_HH" –o "Z:\Projects\Pepys\Second cohort\orient" –ds 3 –at "Z:\Projects\Pepys\Second cohort\orient\atlas"**

and hit Enter.



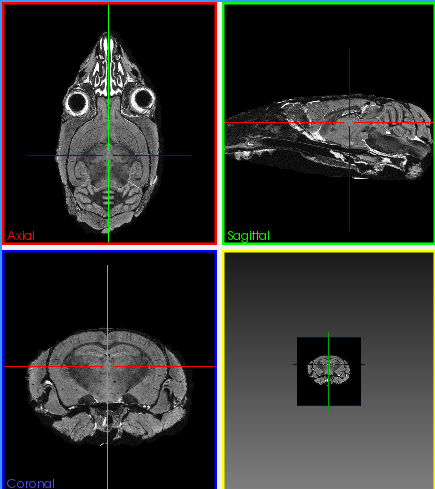
1. If you did not provide atlas images, you’ll need to manually choose one after a few minutes. That will look like this:



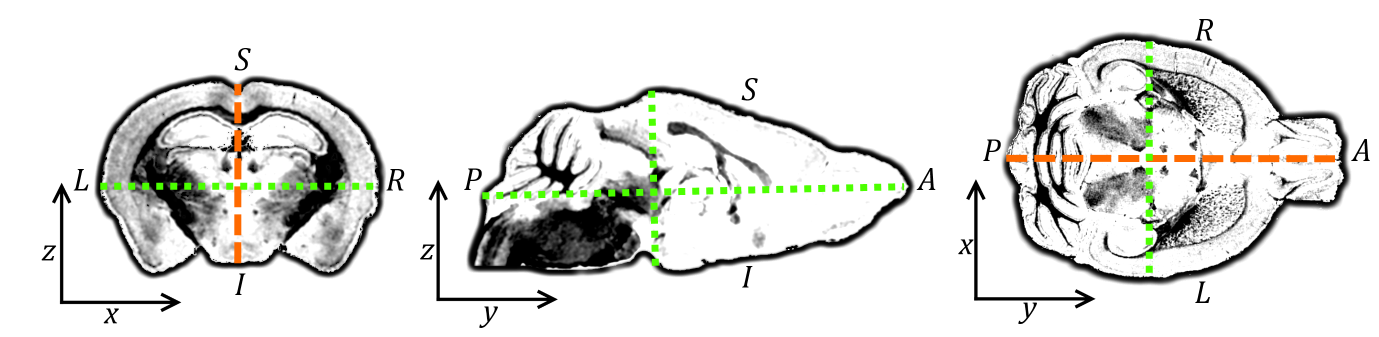
(they are called **\_corrected\_1**, **\_corrected\_2**, etc., and there are only 4 to choose from). Copy the correct one into your atlas folder, and the script will continue.

1. And wait (this might take a while.) The results will eventually appear in the specified output directory.
2. Check all output images are aligned correctly using a viewer which takes the NIfTI header into account.

* In NiftyView, correctly-oriented brains in RAS space look like this:



* Illustration of RAS orientation:



* Slight orientation offsets shouldn’t be a problem as later registration steps should resolve them. Severe orientation offsets shouldn’t occur at all – if they do, there is a problem; please let me know.