FreeSurfer Multiprocessing Pipeline

Contents

Module `freesurfer_wrapper`

freesurfer_wrapper

freesurfer_wrapper aims to facilitate the creation of a multiprocessing pipeline using FreeSurfer. It is a Python wrapper to execute parallel runs of recon-all and some pial edits algorithms.

Requirements

- Docker[1]
- FreeSurfer license key[2]

[1] https://www.docker.com/
[2] https://surfer.nmr.mgh.harvard.edu/registration.html

Usage

Example

Sub-modules

- freesurfer_wrapper.run
- freesurfer_wrapper.scripts

Module `freesurfer_wrapper.run`

Command-line wrapper tool to execute parallel runs of FreeSurfer recon-all and some pial edits algorithms.

This file can also be imported as a module and contains the following functions:

```
* argument_parser -  parser for command-line options, arguments and sub-commands.
* run_command -
* handle_workers - creates a pool of parallel worker processes running commands.
* worker - invokes a subprocess running the command.
* recon - formats recon-all command string.
* edit - formats mri_gcut and mri_binarize command string.
* recon_edit - formats a cp and recon-all command string.
* parse_input_file - parses the input tables.
```

Functions

Function `argument_parser`

```
def argument_parser(
    args: list
) -> ArgumentParser.parse_args
```

Parser for command-line options, arguments and sub-commands.

Parameters

`args` : list Command-line arguments list

Returns

Parser

Function `edit`

```
def edit(
    edit_args: list
) -> str
```

Formats mri_gcut and mri_binarize command string. mri_gcut performs skull stripping algorithm based on graph cut. mri_binarize binarizes the edited mask.

Parameters

`edit_args` : list mri_gcut and mri_binarize arguments list

Returns

`mri_gcut [args] && mri_binarize [args]`

Function `handle_workers`

```
def handle_workers(
    p: int,
    command: function,
    input_file: str
)
```

Creates a pool of parallel worker processes running commands. Workers will be called until all lines from the input file are processed.

Parameters

`p` : int The number of parallel processes.
`command` : function Function returning the command-line string to pass the worker.
`input_file` : str Tab-separated .txt file.

Returns

None

Function `parse_input_file`

```
def parse_input_file(
    input_file: str
) -> List[List[str]]
```

Parses the input tables.

Parameters

`input_file` : str Tab-separated .txt file.

Returns

File lines and columns parsed as a list of lists.

Function `recon`

```
def recon(
    recon_args: list
) -> str
```

Formats recon-all command string.

Parameters

`recon_args` : list recon-all arguments list

Returns

`recon-all [args]`

Function `recon_edit`

```
def recon_edit(
    recon_edit_args: list
) -> str
```

Formats a cp and recon-all command string. cp replaces the original brainmask with the edited brainmask.gcutsT{tissue_ratio}.mgz. recon-all re-runs -autorecon2-wm -autorecon3 stream with the new mask.

Parameters

`recon_edit_args` : list cp and recon-all arguments list

Returns

`cp [args] && recon-all [args]`

Function `run_command`

```
def run_command(
    args
)
```

Pass the appropriate command function to the worker handler.

Parameters

`args` **: list** Command-line arguments list

Returns

None

Function `worker`

```
def worker(
    cmd: str
) -> <function run at 0x7fba824b30e0>
```

Invokes a subprocess running the command.

Parameters

`cmd` **: str** Command-line string

Returns

subprocess.run()

Namespace `freesurfer_wrapper.scripts`

Sub-modules

- freesurfer_wrapper.scripts.create_recon_input

Module `freesurfer_wrapper.scripts.create_recon_input`

Script to create recon input table

This script creates an input table based on the directory organization of the image files.

Please edit the **PATH_PATTERN** variable with the appropriate pathname pattern to find each file.

This file can also be imported as a module and contains the following functions:

```
* create_input_file - creates the input table.
```

Functions

Function `create_input_file`

```
def create_input_file(
    path_pattern: str
)
```

Creates a two column text file to be used as input for the main script recon command. First column: unique ID (combines SUBJECT ID and SESSION ID). Second column: path to DICOM file.

Parameters

`path_pattern` : str Glob pathname pattern to find each DICOM file.

Returns

None

Generated by *pdoc* 0.9.2 (https://pdoc3.github.io).