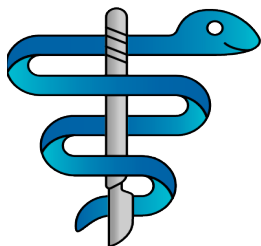

scikit-surgery-evaluation Documentation

Stephen Thompson

Mar 02, 2022

Contents

1	Developing	3
2	Installing	5
3	Licensing and copyright	7
4	Acknowledgements	9
	Python Module Index	15
	Index	17



Author: Stephen Thompson

scikit-surgery-evaluation provides an application to evaluate surgical skills. You can provide a set of unstructured grids representing a set of locations that the user is then expected to target using a tracked pointer, utilising a SciKit-Surgery tracking library (`scikit-surgeryarucotracker`, or `scikit-surgerynditracker`). You can specify paths for the user to follow, or let the system select target meshes automatically.

scikit-surgery-evaluation is part of the [SciKit-Surgery](#) software project, developed at the [Wellcome EPSRC Centre for Interventional and Surgical Sciences](#), part of [University College London \(UCL\)](#).

scikit-surgery-evaluation supports Python 3.X.

```
python sksurgeryeval.py -c configuration.json
```


1.1 Cloning

You can clone the repository using the following command:

```
git clone https://github.com/SciKit-Surgery/scikit-surgery-evaluation
```

1.2 Running tests

Pytest is used for running unit tests:

```
pip install pytest
python -m pytest
```

1.3 Linting

This code conforms to the PEP8 standard. Pylint can be used to analyse the code:

```
pip install pylint
pylint --rcfile=tests/pylintrc sksurgeryeval
```


You can pip install directly from the repository as follows:

```
pip install git+https://github.com/SciKit-Surgery/scikit-surgery-evaluation
```

2.1 Contributing

Please see the [contributing guidelines](#).

2.2 Useful links

- [Source code repository](#)
- [Documentation](#)

CHAPTER 3

Licensing and copyright

Copyright 2019 University College London. scikit-surgery-evaluation is released under the BSD-3 license. Please see the [license file](#) for details.

Acknowledgements

Supported by [Wellcome](#) and [EPSRC](#).

4.1 Requirements for scikit-surgery-evaluation

This is the software requirements file for scikit-surgery-evaluation, part of the SNAPPY project. The requirements listed below should define what scikit-surgery-evaluation does. Each requirement can be matched to a unit test that checks whether the requirement is met.

4.1.1 Requirements

ID	Description	Test
0000	Module has a help page	pylint, see tests/pylint.rc and tox.ini
0001	Functions are documented	pylint, see tests/pylint.rc and tox.ini
0002	Package has a version number	No test yet, handled by git.

4.2 latest

4.2.1 sksurgeryeval package

Subpackages

sksurgeryeval.algorithms package

Submodules

skurgeryeval.algorithms.algorithms module

Algorithms for the surgery evaluation application

`skurgeryeval.algorithms.algorithms.add_map(config)`

Loads vtk models from a directory and returns a list of vtk actors, with mesh visualisation

Param configuration, may contain a “map” key

Param model_to_world: 4x4 matrix, of dtype float32

Returns actors, None if no “map” key

`skurgeryeval.algorithms.algorithms.configure_tracker(config)`

Configures the tracking system. :param: A dictionary containing configuration data :return: The tracker object

:raises: KeyError if no tracker entry in config

`skurgeryeval.algorithms.algorithms.np2vtk(mat)`

Converts a Numpy array to a vtk matrix :param: the number array, should be 4x4 :return: a vtk 4x4 matrix

:raises: ValueError when matrix is not 4x4

`skurgeryeval.algorithms.algorithms.point_in_locator(point, point_locators, radius=1.0)`

Tests whether a point is within a set distance of any of a list of point locators.

Parameters

- **point** – the point to test, in 3D (x,y,z)
- **point_locators** – a list of vtkPointLocators
- **radius** – optional search radius in mm (default=1.0)

Return locator the index of the nearest point locator,

-1 if no locators within radius) :return distance: distance to nearest point_locator

Raises delegates to vtk

`skurgeryeval.algorithms.algorithms.populate_models(config)`

Loads vtk models from a directory and returns a list of vtk actors and associated vtkPointLocators

Param configuration, should contain a target value

Param model_to_world: 4x4 matrix, of dtype float32

Returns locators

Returns actors

Raises KeyError if target not in config

`skurgeryeval.algorithms.algorithms.random_targets(count)`

Create a list of targets

skurgeryeval.algorithms.background_image module

A class to provide the background image

class `skurgeryeval.algorithms.background_image.OverlayBackground(config)`

Bases: object

Provides the background image for the overlay window.

next_image()

Returns a background image. The behaviour is determined by the configuration dictionary used at init.

skurgeryeval.algorithms.locators module

Main loop for surgery evaluation

class skurgeryeval.algorithms.locators.**Locators** (*config*)

Bases: object

stores a list of vtk models and corresponding locators, and handles associated logic

is_hit (*tracking, logger*)

Checks whether a target has been hit :param: the tracking data (3D point) :param: a logger to write notification to

Module contents

skurgeryeval.logging package

Submodules

skurgeryeval.logging.surgery_logger module

Class to handle logging

class skurgeryeval.logging.surgery_logger.**Logger** (*config*)

Bases: object

Implements logging functionality for skurgery-evaluation. Configuration is done by passing a dictionary on construction. Subsequent calls to log("message") will write to log file.

log (*message*)

If logging, passes message to logger

Module contents

skurgeryeval.shapes package

Submodules

skurgeryeval.shapes.cone module

VTK pipeline to represent a surface model via a vtkPolyData.

class skurgeryeval.shapes.cone.**VTKConeModel** (*height, radius, colour, name, visibility=True, opacity=1.0*)

Bases: skurgeryvtk.models.vtk_surface_model.VTKSurfaceModel

Class to create a VTK surface model of a cone.

Module contents

sksurgeryeval.ui package

Submodules

sksurgeryeval.ui.sksurgeryeval_command_line module

Command line processing

```
sksurgeryeval.ui.sksurgeryeval_command_line.main (args=None)  
    Entry point for scikit-surgery-evaluation application
```

sksurgeryeval.ui.sksurgeryeval_demo module

Hello world demo module

```
sksurgeryeval.ui.sksurgeryeval_demo.run_demo (configfile, verbose)  
    Run the application
```

Module contents

scikit-surgery-evaluation

sksurgeryeval.widgets package

Submodules

sksurgeryeval.widgets.overlay module

Main loop for surgery evaluation

```
class sksurgeryeval.widgets.overlay.OverlayApp (config)  
    Bases: sksurgeryutils.common_overlay_apps.OverlayBaseApp  
    Inherits from OverlayBaseApp, adding code to test the proximity of a tracked object to a set of vtk objects  
    update ()  
        Update the background renderer with a new frame, move the model(s) and render
```

Module contents

Module contents

scikit-surgery-evaluation

4.3 First notebook

You can write up experiments in notebooks, and they can be generated into Sphinx docs using `tox -e docs`, and for example set up to run on readthedocs.

See [this](#) and [this](#) examples.

4.3.1 NOTE:

Getting jupyter to run your code in this package relies on 3 things:

- You must ensure you start jupyter within the tox environment.

```
# If not already done.
source .tox/py36/bin/activate

# Then launch jupyter
jupyter notebook
```

- Then when you navigate to and run this notebook, select the right kernel (named after your project) from the kernel menu item, in the web browser.
- Add project folder to system path, as below.

```
[1]: # Jupyter notebook sets the cwd to the folder containing the notebook.
# So, you want to add the root of the project to the sys path, so modules load_
↪ correctly.
import sys
sys.path.append("../..")
```

- modindex
- genindex
- search

a

`skssurgeryeval.algorithms`, 11
`skssurgeryeval.algorithms.algorithms`, 10
`skssurgeryeval.algorithms.background_image`,
10
`skssurgeryeval.algorithms.locators`, 11

l

`skssurgeryeval.logging`, 11
`skssurgeryeval.logging.surgery_logger`,
11

s

`skssurgeryeval`, 12
`skssurgeryeval.shapes`, 12
`skssurgeryeval.shapes.cone`, 11

u

`skssurgeryeval.ui`, 12
`skssurgeryeval.ui.skssurgeryeval_command_line`,
12
`skssurgeryeval.ui.skssurgeryeval_demo`, 12

w

`skssurgeryeval.widgets`, 12
`skssurgeryeval.widgets.overlay`, 12

A

`add_map()` (in module *skurgeryeval.algorithms.algorithms*), 10

C

`configure_tracker()` (in module *skurgeryeval.algorithms.algorithms*), 10

I

`is_hit()` (*skurgeryeval.algorithms.locators.Locators* method), 11

L

Locators (class in *skurgeryeval.algorithms.locators*), 11

`log()` (*skurgeryeval.logging.surgery_logger.Logger* method), 11

Logger (class in *skurgeryeval.logging.surgery_logger*), 11

M

`main()` (in module *skurgeryeval.ui.skurgeryeval_command_line*), 12

N

`next_image()` (*skurgeryeval.algorithms.background_image.OverlayBackground* method), 10

`np2vtk()` (in module *skurgeryeval.algorithms.algorithms*), 10

O

OverlayApp (class in *skurgeryeval.widgets.overlay*), 12

OverlayBackground (class in *skurgeryeval.algorithms.background_image*), 10

P

`point_in_locator()` (in module *skurgeryeval.algorithms.algorithms*), 10

`populate_models()` (in module *skurgeryeval.algorithms.algorithms*), 10

R

`random_targets()` (in module *skurgeryeval.algorithms.algorithms*), 10

`run_demo()` (in module *skurgeryeval.ui.skurgeryeval_demo*), 12

S

skurgeryeval (module), 12

skurgeryeval.algorithms (module), 11

skurgeryeval.algorithms.algorithms (module), 10

skurgeryeval.algorithms.background_image (module), 10

skurgeryeval.algorithms.locators (module), 11

skurgeryeval.logging (module), 11

skurgeryeval.logging.surgery_logger (module), 11

skurgeryeval.shapes (module), 12

skurgeryeval.shapes.cone (module), 11

skurgeryeval.ui (module), 12

skurgeryeval.ui.skurgeryeval_command_line (module), 12

skurgeryeval.ui.skurgeryeval_demo (module), 12

skurgeryeval.widgets (module), 12

skurgeryeval.widgets.overlay (module), 12

U

`update()` (*skurgeryeval.widgets.overlay.OverlayApp* method), 12

V

VTKConeModel (class in *skurgeryeval.shapes.cone*), 11