scikit-surgerytorch Documentation

Thomas Dowrick

Mar 22, 2022

Contents

The aim of scikit-surgery torch is to provide a home for various pytorch models/examples/utilities that may be useful for Image Guided Surgery.

CHAPTER 1

Features

Implemented models:

- High Resolution Stereo network Inference only, see author's repo for pre trained weights. As at commit aae0b9b.
- Volume2SurfaceCNN Inferencece only, see author's repo for pre trained weights. As at commit 5a656381.
- Models can run on GPU or CPU.
- Example usage in *tests/*.

Source code is avaialble on GitHub.

1.1 scikit-surgerytorch

Author: Thomas Dowrick

scikit-surgerytorch 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).

The aim of scikit-surgery torch is to provide a home for various pytorch models/examples/utilities that may be useful for Image Guided Surgery.

1.1.1 Features

Implemented models:

- High Resolution Stereo network Inference only, see author's repo for pre trained weights. As at commit aae0b9b.
- Volume2SurfaceCNN Inferencece only, see author's repo for pre trained weights. As at commit 5a656381.
- Models can run on GPU or CPU.
- Example usage in *tests/*.

scikit-surgerytorch is NOT meant to be a layer on-top of pytorch or provide a new kind-of platform. The aim is that researchers can learn from examples, and importantly, learn how to deliver an algorithm that can be used by other people out of the box, with just a `pip install`, rather than a new user having to re-implement stuff, or struggle to get someone else's code running.

Cloning

You can clone the repository using the following command:

```
git clone https://github.com/UCL/scikit-surgerytorch
```

Running tests

Pytest is used for running unit tests:

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

Linting

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

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

1.1.2 Installing

You can pip install directly from the repository as follows:

```
pip install git+https://github.com/UCL/scikit-surgerytorch
```

Contributing

Please see the contributing guidelines.

Useful links

- Source code repository
- Documentation

1.1.3 Licensing and copyright

1.1.4 Acknowledgements

Supported by Wellcome and EPSRC.

1.2 Stereo Reconstruction

1.2.1 High Resolution Stereo

Module to implement Hierarchical Deep Stereo Matching on High Resolution Images network.

class sksurgerytorch.models.high_res_stereo.**HSMNet**(*max_disp: int = 255*, *entropy_threshold: float = -1*, *level: int = 1*, *scale_factor: float = 0.5*, *weights=None*)

Class to encapsulate network form 'Hierarchical Deep Stereo Matching on High Resolution Images'.

Thanks to Gengshang Yang, for their network implementation.

Parameters

- **max_disp** – Maximum number of disparity levels
- **entropy_threshold** – Pixels with entropy above this value will be ignored in the disparity map. Disabled if set to -1.
- **level** – Set to 1, 2 or 3 to trade off quality of depth estimation against runtime. 1 = best depth estimation, longer runtime, 3 = worst depth estimation, fastest runtime.
- **scale_factor** – Images can be resized before passing to the network, for perfomance impromvents. This sets the scale factor.
- **weights** – Path to trained model weights (.tar file)

predict(*left_image: numpy.ndarray*, *right_image: numpy.ndarray*) → numpy.ndarray

Predict disparity from a pair of stereo images.

Parameters

- **left_image** (*np.ndarray*) – Left stereo image, 3 channel RGB
- **right_image** (*np.ndarray*) – Right stero image, 3 channel RGB

Returns Predicted disparity, grayscale

Return type np.ndarray

sksurgerytorch.models.high_res_stereo.**run_hsmnet_model**(*max_disp*, *entropy_threshold*, *level*, *scale_factor*, *weights*, *left_image*, *right_image*, *output_file*)

This is for the command line entry point

class sksurgerytorch.models.high_res_stereo.**toTensorLegacy**
.

1.3 Non Rigid Registration

1.3.1 Volume 2 Surface CNN

V2SNet Model Impementation

class sksurgerytorch.models.volume_to_surface.**Volume2SurfaceCNN**(*mask: bool = True*, *weights: str = None*, *grid_size: int = 64*)

Class to encapsulate network form 'Non-Rigid Volume to Surface Registration using a Data-Driven Biomechanical Model'.

Thanks to Micha Pfieffer, for their network implementation.

Parameters

- **mask** (*bool*) – If true, use masking
- **weights** (*str*) – Path to trained model weights (.tar file)

predict(*preoperative: numpy.ndarray*, *intraoperative: numpy.ndarray*) → numpy.ndarray

Predict the displacement field between model and surface.

Parameters

- **preoperative** (*np.ndarray*) – Preoperative surface/point cloud
- **intraoperative** (*np.ndarray*) – Intraoperative surface/point cloud

Returns Displacement field

Return type np.ndarray

Index