Venus Imaging Analysis Documentation

Release 0.10.0

K.-Michael Aye

Jul 02, 2020

Contents:

CHAPTER 1

Venus Imaging Analysis

Python tools for Venus Image Analysis

- Free software: MIT license
- Documentation: https://venim.readthedocs.io.

1.1 Features

If the feature has been implemented in an importable way, it's indicated by a filled checkmark [x]

- [x] Read FITS arrays (done via astropy.io.fits)
- [x] FITS image stats scanner (creates CSV overview file)
- **[ ] For un-calibrated ground-based observations:**
  - [ ] Standard image reduction pipeline: bias subtraction, flat field normalization, etc.
  - [ ] Other image clean-up: bad pixels, cosmic rays, detector artifacts
  - [ ] Generate estimated errors for each pixel
  - [ ] Transform detector x,y coordinates into local Lat, Lon
- **[ ] Access PDS data automatically via mission dependent interfaces**
  - [x] Automatic downloads per volume orbit
  - [x] Local easy data access per data ID
  - [x] Akatsuki version implemented

    - [ ] VEX
- [x] Annotate images and implement a point-based circle-fit
- [ ] Take various gradients of images
- [ ] Filter images in the spatial frequency domain
- [ ] Sub-pixel disk registration
- [ ] Robust stacking of co-registered images
- [ ] Cloud tracking

1.2 Credits

The content of this package is based on summer science work by Nicolas Ardavin and Kenyon Prater, under the lead of Eliot Young and Mark Bullock. Later improvements have been implemented by the package maintainer Michael Aye.

This package was created with Cookiecutter and the forked michaelaye/cookiecutter-pypackage-conda project template.

CHAPTER 2

Installation

2.1 Stable release

To install Venus Imaging Analysis, run this command in your terminal:

```
$ pip install venim
```

This is the preferred method to install Venus Imaging Analysis, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for Venus Imaging Analysis can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/michaelaye/venim
```

Or download the tarball:

```
$ curl  -OL https://github.com/michaelaye/venim/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

To use Venus Imaging Analysis in a project:

```
import venim
```

CHAPTER 4

venim

4.1 venim package

4.1.1 Subpackages

venim.akatsuki package

Submodules

venim.akatsuki.ir2 module

Module contents

4.1.2 Submodules

venim.config module

venim.fits_cli module

venim.image module

Tools to work with images.

class venim.image.**Image**(*path*)

Bases: object

annotate()

circle_fit

equalized

exposure

full_frame

get_stored_points()

imagetime

name

plot(*sqrt=False*, *pmin=1*, *pmax=99*, *with_fit=False*)

plot_equalized()

plot_title

points_data

points_path

rescaled

store_points()

wavelength

venim.mask_functions module

venim.mask_functions.**cloudMask**(*center*, *area*, *radius*, *angle*)

venim.mask_functions.**crescentMask**(*center*, *area*, *outer_radius*, *pixel_angle*, *border*)

venim.mask_functions.**loadMask**(*path*)

venim.mask_functions.**outerMask**(*center*, *outer_radius*, *border*)

venim.pathmanager module

venim.stats module

venim.stats.**scan_image_directory**(*path*)

Scan directory of FITS files to create basic stats.

Creates CSV file ready to be read by pandas and print-out of the stats if less than 100 entries.

Parameters **path** (*str, pathlib.Path*) –

Returns DataFrame containing the collected stats

Return type pd.DataFrame

venim.utils module

venim.venim module

Main module.

4.1.3 Module contents

Top-level package for Venus Imaging Analysis.

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/michaelaye/venim/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

5.1.4 Write Documentation

Venus Imaging Analysis could always use more documentation, whether as part of the official Venus Imaging Analysis docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/michaelaye/venim/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up *venim* for local development.

1. Fork the *venim* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/venim.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv venim
$ cd venim/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests:

```
$ flake8 venim tests
$ python setup.py test or py.test
```

   To get flake8, just pip install them into your conda env.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/michaelaye/venim/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ py.test tests.test_venim
```

CHAPTER 6

Credits

6.1 Development Lead

- K.-Michael Aye <kmichael.aye@gmail.com>

6.2 Contributors

- Eliot Young
- Mark Bullock

6.3 Initial development

Mostly in the form of Jupyter notebooks

- Nicolas Ardavin
- Kenyon Prater

CHAPTER 7

Changelog

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

V

Index