pyorbital Documentation

Release 1.5.0+8.g7efe253.dirty

The Pytroll crew

Oct 22, 2019

Contents

Pyorbital is a python package to compute orbital parameters for satellites from TLE files as well as astronomical parameters of interest for satellite remote sensing. Currently pyorbital only supports low earth orbit satellites.

CHAPTER 1

Installation

Pyorbital comes with a file platforms.txt that maps satellite name to NORAD identifier. This file needs to be copied to the appropriate satpy etc directory ($PPP_CONFIG_DIR). It is wise to check it contains your satellites of interest. The NORAD identifier can be found as the first number of each line in the Two-Line Elements (eg. from celestrak).

CHAPTER 2

TLE files

Pyorbital has a module for parsing NORAD TLE-files

```
>>> from pyorbital import tlefile
>>> tle = tlefile.read('noaa 18', '/path/to/my/tle_file.txt')
>>> tle.inclination
99.043499999999995
```

If no path is given pyorbital tries to read the earth observation TLE-files from celestrak.com

CHAPTER 3

Computing satellite position

The orbital module enables computation of satellite position and velocity at a specific time:

```
>>> from pyorbital.orbital import Orbital
>>> from datetime import datetime
>>> # Use current TLEs from the internet:
>>> orb = Orbital("Suomi NPP")
>>> now = datetime.utcnow()
>>> # Get normalized position and velocity of the satellite:
>>> orb.get_position(now)
(array([-0.20015267,  0.09001458,  1.10686756]),
 array([ 0.06148495,  0.03234914,  0.00846805]))
>>> # Get longitude, latitude and altitude of the satellite:
>>> orb.get_lonlatalt(now)
(40.374855865574951, 78.849923885700363, 839.62504115338368)
```

CHAPTER 4

Use actual TLEs to increase accuracy

```
>>> from pyorbital.orbital import Orbital
>>> from datetime import datetime
>>> orb = Orbital("Suomi NPP")
>>> dtobj = datetime(2015,2,7,3,0)
>>> orb.get_lonlatalt(dtobj)
(152.11564698762811, 20.475251739329622, 829.37355785502211)
```

But since we are interested in knowing the position of the Suomi-NPP more than two and half years from now (September 26, 2017) we can not rely on the current TLEs, but rather need a TLE closer to the time of interest:

```
>>> snpp = Orbital('Suomi NPP', tle_file='/data/lang/satellit/polar/orbital_elements/
↪TLE/201502/tle-20150207.txt')
>>> snpp.get_lonlatalt(dtobj)
(105.37373804512762, 79.160752404540133, 838.94605490133154)
```

If we take a TLE from one week earlier we get a slightly different result:

```
>>> snpp = Orbital('Suomi NPP', tle_file='/data/lang/satellit/polar/orbital_elements/
↪TLE/201501/tle-20150131.txt')
>>> snpp.get_lonlatalt(dtobj)
(104.1539184988462, 79.328272480878141, 838.81555967963391)
```

CHAPTER 5

Computing astronomical parameters

The astronomy module enables computation of certain parameters of interest for satellite remote sensing for instance the Sun-zenith angle:

```
>>> from pyorbital import astronomy
>>> from datetime import datetime
>>> utc_time = datetime(2012, 5, 15, 15, 45)
>>> lon, lat = 12, 56
>>> astronomy.sun_zenith_angle(utc_time, lon, lat)
62.685986438071602
```

CHAPTER 6

API

6.1 Orbital computations

Module for computing the orbital parameters of satellites.

class pyorbital.orbital.**OrbitElements**(*tle*)
Class holding the orbital elements.

class pyorbital.orbital.**Orbital**(*satellite*, *tle_file=None*, *line1=None*, *line2=None*)
Class for orbital computations.

The *satellite* parameter is the name of the satellite to work on and is used to retrieve the right TLE data for internet or from *tle_file* in case it is provided.

find_aol(*utc_time*, *lon*, *lat*)

find_aos(*utc_time*, *lon*, *lat*)

get_last_an_time(*utc_time*)
Calculate time of last ascending node relative to the specified time

get_lonlatalt(*utc_time*)
Calculate sublon, sublat and altitude of satellite. http://celestrak.com/columns/v02n03/

get_next_passes(*utc_time*, *length*, *lon*, *lat*, *alt*, *tol=0.001*, *horizon=0*)
Calculate passes for the next hours for a given start time and a given observer.

Original by Martin.

utc_time: Observation time (datetime object) length: Number of hours to find passes (int) lon: Longitude of observer position on ground (float) lat: Latitude of observer position on ground (float) alt: Altitude above sea-level (geoid) of observer position on ground (float) tol: precision of the result in seconds horizon: the elevation of horizon to compute risetime and falltime.

Return: [(rise-time, fall-time, max-elevation-time), ...]

get_observer_look(*utc_time*, *lon*, *lat*, *alt*)
Calculate observers look angle to a satellite. http://celestrak.com/columns/v02n02/

utc_time: Observation time (datetime object) lon: Longitude of observer position on ground in degrees east lat: Latitude of observer position on ground in degrees north alt: Altitude above sea-level (geoid) of observer position on ground in km

Return: (Azimuth, Elevation)

get_orbit_number(*utc_time*, *tbus_style=False*)
Calculate orbit number at specified time. Optionally use TBUS-style orbit numbering (TLE orbit number + 1)

get_position(*utc_time*, *normalize=True*)
Get the cartesian position and velocity from the satellite.

exception pyorbital.orbital.**OrbitalError**

pyorbital.orbital.**get_observer_look**(*sat_lon*, *sat_lat*, *sat_alt*, *utc_time*, *lon*, *lat*, *alt*)
Calculate observers look angle to a satellite. http://celestrak.com/columns/v02n02/

utc_time: Observation time (datetime object) lon: Longitude of observer position on ground in degrees east lat: Latitude of observer position on ground in degrees north alt: Altitude above sea-level (geoid) of observer position on ground in km

Return: (Azimuth, Elevation)

pyorbital.orbital.**kep2xyz**(*kep*)

6.2 TLE handling

exception pyorbital.tlefile.**ChecksumError**
ChecksumError.

pyorbital.tlefile.**SATELLITES = {'ALOS-2': '39766', 'CLOUDSAT': '29107', 'CRYOSAT-2': '365**
The platform numbers are given in a file $PPP_CONFIG/platforms.txt in the following format:

class pyorbital.tlefile.**Tle**(*platform*, *tle_file=None*, *line1=None*, *line2=None*)
Class holding TLE objects.

line1
Return first TLE line.

line2
Return second TLE line.

platform
Return satellite platform name.

pyorbital.tlefile.**fetch**(*destination*)
Fetch TLE from internet and save it to *destination*.

pyorbital.tlefile.**main**()
Main for testing TLE reading.

pyorbital.tlefile.**read**(*platform*, *tle_file=None*, *line1=None*, *line2=None*)
Read TLE for *platform* from *tle_file*

File is read from *line1* to *line2*, from the newest file provided in the TLES pattern, or from internet if none is provided.

pyorbital.tlefile.**read_platform_numbers**(*in_upper=False*, *num_as_int=False*)
Read platform numbers from $PPP_CONFIG_DIR/platforms.txt if available.

6.3 Astronomical computations

Astronomy module. Parts taken from http://www.geoastro.de/elevaz/basics/index.htm

pyorbital.astronomy.**cos_zen**(*utc_time*, *lon*, *lat*)
Cosine of the sun-zenith angle for *lon*, *lat* at *utc_time*. utc_time: datetime.datetime instance of the UTC time lon and lat in degrees.

pyorbital.astronomy.**get_alt_az**(*utc_time*, *lon*, *lat*)
Return sun altitude and azimuth from *utc_time*, *lon*, and *lat*. lon,lat in degrees The returned angles are given in radians.

pyorbital.astronomy.**gmst**(*utc_time*)
Greenwich mean sidereal utc_time, in radians.

As defined in the AIAA 2006 implementation: http://www.celestrak.com/publications/AIAA/2006-6753/

pyorbital.astronomy.**jdays**(*utc_time*)
Get the julian day of *utc_time*.

pyorbital.astronomy.**jdays2000**(*utc_time*)
Get the days since year 2000.

pyorbital.astronomy.**observer_position**(*time*, *lon*, *lat*, *alt*)
Calculate observer ECI position.

http://celestrak.com/columns/v02n03/

pyorbital.astronomy.**sun_earth_distance_correction**(*utc_time*)
Calculate the sun earth distance correction, relative to 1 AU.

pyorbital.astronomy.**sun_ecliptic_longitude**(*utc_time*)
Ecliptic longitude of the sun at *utc_time*.

pyorbital.astronomy.**sun_ra_dec**(*utc_time*)
Right ascension and declination of the sun at *utc_time*.

pyorbital.astronomy.**sun_zenith_angle**(*utc_time*, *lon*, *lat*)
Sun-zenith angle for *lon*, *lat* at *utc_time*. lon,lat in degrees. The angle returned is given in degrees

Python Module Index

p

Index