helical$_{t}hreadDocumentation$

Release 0.2.3

Wink Saville

Oct 13, 2020

CONTENTS

CHAPTER

ONE

PACKAGE DOCS 0.2.3

class helical_thread.**HelicalThread**(*radius*, *pitch*, *height*, *taper_out_rpos=0*, *taper_in_rpos=1*, *inset_offset=0*, *first_t=0*, *last_t=1*, *angle_degs=45*, *major_cutoff=0*, *minor_cutoff=0*, *ext_clearance=0.1*, *thread_overlap=0.001*)

Bases: `taperable_helix.helix.Helix`

A set of fields used to represent a helical thread and passed as the parameter to *helical_thread*.

Control of the size and spacing of the thread using the various fields in Helix and those below.

angle_degs: float = 45
angle in degrees

major_cutoff: float = 0
Size of of flat at the major diameter

minor_cutoff: float = 0
Size of flat at the minor diameter

ext_clearance: float = 0.1
External clearance between external and internal threads

thread_overlap: float = 0.001
Amount to overlap threads with the core so the union of core and threads is a manifold

first_t: float = 0

helix(*hl=None*)
This function returns a Function that is used to generates points on a helix.

It takes an optional HelixLocation which refines the location of the final helix when its tapered. If HelixLocation is None then the radius is Helix.radius and horz_offset and vert_offset will be 0. If its not None HelixLocation.radius maybe None, in which case Helix.radius will be used. and HelixLocation.horz_offset will be added to the radius and used to calculate x and y. The HelixLocation.vert_offset will be added to z.

This function returns a function, f. The funciton f that takes one parameter, an inclusive value between first_t and last_t. We then define t_range=last_t-first_t and the rel_height=(last_t-t)/t_range. The rel_height is the relative position along the "z-axis" which is used to calculate function functions returned tuple(x, y, z) for a point on the helix.

Credit: Adam Urbanczyk from cadquery [forum post](https://groups.google.com/g/cadquery/c/5kVRpECcxAU/m/7no7_ja6AAAJ)

Parameters **hl** (`Optional[HelixLocation]`) – Defines a refinded location when the helix is tapered

Return type `Callable[[float], Tuple[float, float, float]]`

Returns A function which is passed "t", an inclusive value between first_t and last_t and returns a 3D point (x, y, z) on the helix as a function of t.

inset_offset: float = 0

last_t: float = 1

taper_in_rpos: float = 1

taper_out_rpos: float = 0

radius: float

pitch: float

height: float

class helical_thread.**ThreadHelixes**(*ht*, *int_helix_radius=0*, *int_helixes=<factory>*, *ext_helix_radius=0*, *ext_helixes=<factory>*)

The helixes returned by helical_thread` that represents the internal thread, prefixed with *int_* and the external thread, prefixed with *ext_*.

ht: helical_thread.helicalthread.HelicalThread
The basic Dimensions of the helixes

int_helix_radius: float = 0
The internal thread radius

int_helixes: List[taperable_helix.helix.HelixLocation]
List of the internal helix locations

ext_helix_radius: float = 0
The external thread radius

ext_helixes: List[taperable_helix.helix.HelixLocation]
List of the external helix locations

helical_thread.**helical_thread**(*ht*)

Given HelicalThread compute the internal and external helixes thread and returning them in ThreadHelixes. int_hexlix_radius, int_helixes, ext_helix_radius and ext_helixes. The helixes are an array of HelixLocations that define the helixes of the thread. If minor_cutoff is 0 then the thread will be triangular and the length of the {int|ext}_helixes 3. if minor_cutoff > 0 then the thread will be a trapezoid with the length of the {int|ext}_helixes will be 4.

Parameters **ht** (HelicalThread) – The basic dimensions of the helicla thread

Return type ThreadHelixes

Returns internal and external helixes necessary to use taperable-helix

CHAPTER

TWO

INSTALLATION

2.1 Stable release

To install helical-thread, run this command in your terminal:

```
$ pip install helical-thread
```

This is the preferred method to install helical-thread, as it will always installs the most recent stable release.

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

2.2 Test release from testpypi

To install helical-thread from testpypi, run this command in your terminal:

```
$ pip install --index-url https://test.pypi.org/simple/ helical-thread
```

2.3 From sources

The sources for helical_thread can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/winksaville/py-helical-thread helical-thread
$ cd helical-thread
```

Or download the tarball

```
$ curl -OJL https://github.com/winksaville/py-helical-thread/releases/v0.2.3.tar.gz
```

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

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

Or if you want to install in editable mode for development:

```
$ make install-dev
```

```
$ pip install -e . -r dev-requirements.txt
```

2.4 Uninstall

```
$ pip uninstall helical-thread
```

CHAPTER

THREE

USAGE

To use helical_thread in a project:

```
import helical_thread
```

CHAPTER

FOUR

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:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at helixal-thread 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.

4.1.2 Fix Bugs

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

4.1.3 Implement Features

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

4.1.4 Write Documentation

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

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at helixal-thread 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 :)

4.2 Get Started!

Ready to contribute? Here's how to set up helixal-thread for local development.

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

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

3. Instantiate an (virtual) enviorment which supports python3.7, isort, black, flake8 and bump2version. Using *make install-dev* will install appropriate development dependencies:

```
$ <instantiate your virtual environment if necessary>
$ cd helical_thread/
$ make install-dev
```

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 are formantted correctly and pass the tests:

```
$ make format
$ make test
```

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.

4.3 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 3.7 and 3.8.

4.4 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed. Then run and validate that test.pypi.org is good:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ make push-tags
$ make release-testpypi
```

Finally, assuming test.pypi.org is good, push to pypi.org:

```
$ make release
```

CHAPTER

FIVE

AUTHORS

5.1 Development Lead

- Wink Saville <wink@saville.com>

5.2 Contributors

None yet. Why not be the first?

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

INDEX