
pyransac

Release 1.1.3

Adam Morrisett

Sep 06, 2022

CONTENTS

1	Installation	3
2	Getting Started	5
3	Example Program	7
4	Custom Data Models	9
5	Table of Contents	11
5.1	Reference	11
6	Indices and Tables	15
	Index	17

`pyransac` is a general-purpose random sample consensus (RANSAC) framework written in Python. You can use it to remove outliers from your data sets given a data model to which you expect your data to fit. For convenience, some data models (such as a 2D straight line) are already provided. However, you are free to define your own data models.

INSTALLATION

You can install pyransac using pip with the following command:

```
$ python3 -m pip install pyransac
```


GETTING STARTED

After installing `pyransac`, all you need to do is create a data model definition (or use one of the built in models) and specify the RANSAC algorithm parameters. From there you can call the `find_inliers` function and pass in your data, model, and parameters. The function will return a list of your inliers.

EXAMPLE PROGRAM

The following is a simple example of filtering data against a 2D line model.

```
1 import pyransac
2 from pyransac import line2d
3
4 # Create data
5 inliers = [line2d.Point2D(x, x) for x in range(0, 10)]
6 outliers = [line2d.Point2D(x ** 2, x + 10) for x in range(0, 5)]
7 data = inliers + outliers
8
9 # Specify our RANSAC parameters
10 params = pyransac.RansacParams(samples=2,
11                                iterations=10,
12                                confidence=0.999,
13                                threshold=1)
14
15 # Create our model object
16 model = line2d.Line2D()
17
18 # Get the inliers
19 inliers = pyransac.find_inliers(points=data,
20                                model=model,
21                                params=params)
22
23 # Compare the data sets
24 print(data)
25 print(inliers)
```


CUSTOM DATA MODELS

All data models are derived from the `Model` base class. This class defines an interface consisting of two functions: `make_model` and `calc_error`. The `make_model` function generates a data model from a set of data points. The `calc_error` function calculates the error between a data point and the generated model. See the `Model` [reference](#) for more information.

You can define custom data models by extending the `Model` class. `pyransac` provides the following built-in data models:

- *Line2D* — a 2-dimensional line model

TABLE OF CONTENTS

5.1 Reference

5.1.1 RANSAC

class `pyransac.ransac.RansacParams`(*samples: int, iterations: int, confidence: float, threshold: float, replacement: bool*)

Random sample consensus (RANSAC) function parameters.

This class contains the parameters for the RANSAC algorithm.

confidence: float

The RANSAC confidence value ($0 \leq \text{confidence} \leq 1$).

iterations: int

Maximum number iterations to complete.

replacement: bool

The samples should be drawn with replacement (i.e. repeats are OK)

samples: int

The number of random samples to take per iteration.

threshold: float

The error threshold to consider a point an inlier

`pyransac.ransac.find_inliers`(*points: List, model: Model, params: RansacParams*)

Find the inliers from a data set.

Finds the inliers from a given data set given a model and an error function.

Parameters

- **points** – data points to evaluate
- **model** – type of model to which the data should adhere
- **params** – parameters for the RANSAC algorithm

Returns

inliers

5.1.2 Data Models

class pyransac.base.**Model**

ABC class for data models.

Derivative classes should extend this class and implement its interface.

abstract **calc_error**(*point*) → float

Calculates error between data point and model.

Parameters

point – data point to test against

abstract **make_model**(*points: List*) → None

Makes a model from given data points.

Parameters

points – list of data points with which to make model

class pyransac.line2d.**Line2D**(*slope=None, y_int=None, x_int=None*)

Model for a 2-dimensional line.

calc_error(*point: Point2D*) → float

Calculate error between data point and 2D model.

Parameters

point – data point to calculate error with

Returns

calculated error

make_model(*points: List[Point2D]*) → None

Makes equation for 2D line given two data points.

Model parameters are stored internally.

Parameters

points – list of data points to make model (length must be 2)

Returns

None

property **slope**

Gets the slope of the model.

Returns

slope of line (None if model not made).

property **x_int**

Gets the x intercept of the model.

Returns

x intercept of line (None if model not made).

property **y_int**

Gets the y intercept of the model.

Returns

y intercept of line (None if model not made).

5.1.3 Helpers

class pyransac.line2d.**Point2D**(*x: float, y: float*)

2-dimensional point class.

This is a simple class to contain Cartesian coordinates of 2D point.

x: float

x coordinate of point.

y: float

y coordinate of point

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

C

`calc_error()` (*pyransac.base.Model* method), 12
`calc_error()` (*pyransac.line2d.Line2D* method), 12
`confidence` (*pyransac.ransac.RansacParams* attribute),
 11

F

`find_inliers()` (*in module pyransac.ransac*), 11

I

`iterations` (*pyransac.ransac.RansacParams* attribute),
 11

L

`Line2D` (*class in pyransac.line2d*), 12

M

`make_model()` (*pyransac.base.Model* method), 12
`make_model()` (*pyransac.line2d.Line2D* method), 12
`Model` (*class in pyransac.base*), 12

P

`Point2D` (*class in pyransac.line2d*), 13

R

`RansacParams` (*class in pyransac.ransac*), 11
`replacement` (*pyransac.ransac.RansacParams* attribute), 11

S

`samples` (*pyransac.ransac.RansacParams* attribute), 11
`slope` (*pyransac.line2d.Line2D* property), 12

T

`threshold` (*pyransac.ransac.RansacParams* attribute),
 11

X

`x` (*pyransac.line2d.Point2D* attribute), 13
`x_int` (*pyransac.line2d.Line2D* property), 12

Y

`y` (*pyransac.line2d.Point2D* attribute), 13
`y_int` (*pyransac.line2d.Line2D* property), 12