**LIDAR Filters**

This project “LIDAR Filters” is designed to reduce noise in LIDAR scans. There are two types of filters:

1. Range Filter - This filter crops extreme values so that all values lie within an acceptable range of values

2. Median Filter - This filter smoothens the scan values by taking the median of 'D' previous scan values

**PREREQUISITES:**

1. This project requires Python 2.7.x with numpy module.

**SETUP:**

***Step 1:*** To utilize these filters, just copy the source files into your project and import them:

from <copied-location> import RangeFilter

from <copied-location> import MedianFilter

***Step 2:*** The error messages in these classes are externalized into errormessages.py. Also copy and import this class:

from <copied-location> import ErrorMessages

**USING THE RANGE FILTER:**

***Step 1.*** Initialize the RangeFilter object. This object gives you the option to modify the default range minimum (0.03) and range maximum (50) values.

Ex: rangefilter = RangeFilter(range\_min = 1, range\_max = 10)

rangefilter = RangeFilter()

***Step 2.*** Pass valid scan lists one by one to the update method of this object. This method returns a list of range cropped scan values.

Ex. rangefilter.update([1.5, 2.0, 2.5, 3.0])

**USING THE MEDIAN FILTER:**

***Step 1.*** Initialize the MedianFilter object by passing a valid D value (D must be a positive integer). This object gives you the option to add historic scans that you may want to be considered in the smoothening future scans.

This data must be a list of scans of the same length. Based on the given D value, this historic scans will be used to calculate the median of future scans.

Ex: medianfilter = MedianFilter(2, [[1, 2 , 3], [3, 2, 5]]

medianfilter = MedianFilter(2)

***Step 2.*** Pass valid scan lists one by one (of equal lengths) to the update method of this object. This method returns a list of smoothend values

Ex. medianfilter.update([5,2,6])

**RUNNING TEST CASES:**

The unit test files for the source files can be found under “tests” directory. These tests use a utility ExtendedTestCase class to check if the raised error message matches the expected error message.

**CONSTRAINTS:**

1. A valid scan is a list of Integer/Float values.
2. All scans sent to an instance of MedianFilter object must be of equal length.
3. Historic scan data sent to initialize the MedianFilter object must be a list of valid scans of equal length.
4. D value sent to MedianFilter must be a positive integer.
5. Custom range minimum and maximum values must be Integer/Float values.

**REFERENCES:**

1. ExtendedTestCase class has been taken from a blog in StackOverFlow.
2. Referred <https://docs.python-guide.org/writing/structure/> to structure my project.
3. Referred <https://docs.python.org/2/library/unittest.html> to understand the Unit Testing framework in python.