Value Crossing Detector User Guide

Version: 1.0

Document Date: 18/11/2018

Author: Tansel Arif

1. Introduction

The Value Crossing Detector is a library which can be embedded into larger projects to return the number of times a signal crossing a particular value.

Function List:

```
def zero_crossing_detector(signal):
    """
    A function to count the number of times a list of floats crosses the
    value x = 0
    :param signal: A list of floats representing movement in 1-D space
    :return: An integer representing the number of times the trajectory
    has crossed x = 0
    """

def value_crossing_detector(signal, value):
    """
    A function to count the number of times a list of floats crosses
    the value x = a where a is a user defined value
    :param signal: A list of floats representing movement in 1-D space
    :param value: A user specified value to check whether it is crossed by
    the trajectory
    :return: An integer representing the number of times the trajectory
    has crossed x = a
    """
```

where signal is a list of floats and value is a float. Both functions return an integer representing the number of value crossing events.

2. Example Usage

a. Simple Usage

Import ValueCrossingDetector.py:

```
import ValueCrossingDetector
```

This imports all the functions within that file. The simplest usage is to provide a list of floating point numbers (the signal) to the *zero_crossing_detector* function. This function returns the number of times the signal crosses 0:

```
signal = [5, 4, 3, 2, 3]

zcd = ValueCrossingDetector.zero_crossing_detector(signal)
```

```
print(zcd)
```

The output in the above example is 0:

b. Usage with a user specified value

If the task is to perform a Crossing Detection but with a value different from zero, it can be done using the value_crossing_detector function:

```
signal = [5, 4, 3, 2, 3]
value = 2.5

vcd = ValueCrossingDetector.value_crossing_detector(signal, value)

print(vcd)
```

The output from this is 2, indicating that the value 2.5 is crossed by the signal twice.

If the signal hits the given value but does not cross it, it does not count as a crossing:

```
signal = [5, 4, 3, 2, 3]
value = 2

vcd = ValueCrossingDetector.value_crossing_detector(signal, value)

print(vcd)
```

The above code block returns 0.

c. Additional Examples

Additional examples for edge cases can be found in the *test_zero_crossing_detector.py* unit tests file.