Tuning the Step Counter for `mlExplore.py`

Thomas Haaland

August 2020

1 Introduction to the step counter

To use the step counter and rotation detector you need the files `mlExplore.py` and `mlExploreSettings.txt`. The code for the step counter is in the file `mlExplore.py`, while tolerances are set in `mlExploreSettings.txt`. The step counter can be imported for any other project. For tuning the step counter you can run the `mlExplore.py` file directly to get a graph feed to see how the acceleration data behaves.

2 The `mlExploreSettings.txt` file

The `mlExploreSettings.txt` is used to save settings and customise thresholds. An example of the settings file is the following:

```
# This file contains tunable parameters for the class FeatureBox
# in the file mlExplore.py. Any lines starting with '#' is not
# read by the program. The program has two different
# functionalities: step detection and rotation detection.
# The first functionality has four different ways of detecting
# a step:
#
#    Accel is acceleration magnitude.
#    Jerk is rate of change in acceleration magnitude
#    Area is the sum of all accelerations
#    Rotation is the rotation magnitude.
#
# Whenever Accel is on a peak, the detector checks whether the
# values for Accel, Jerk, Area and Rotation are within the
# tolerances. If they are, the peak is registered as a step.
#
# Step detector tolerances:
#             Accel     Jerk    Area    Rotation
minimumStep   0.05      0       0       0
maximumStep   120       120     120     120
```

```
#
# The rotation detector works by checking whether the rate of
# rotation around the Y axis is larger than the tolerances.
# If the rotation rate larger than the Left column or smaller
# than the Right column, rotation is reported as being detected.
#
# Rotation tolerances:
#                Left       Right
minimumRotation  3          -3
```

All lines starting with `#` are not read by the program and are comments meant for the user. This leaves three lines which contain information used by the program:

- minimumStep
- maximumStep
- minimumRotation

2.1 The step counter

The first two lines are for the step counter, while the last line is for the rotation detector. The primary way to detect a step is looking at the Accel column.

```
#               Accel
minimumStep     0.05
maximumStep     1
```

This is the column which the step counter uses to check whether there is a step or not. Whenever the maximum acceleration is between the minimumStep and maximumStep values, a step is counted. The remaining columns are helper thresholds to help tune the counter if needed. If they are not to be used, they should have a minimum value of 0 and a maximum value of 120. I will discuss advanced tuning later in the document.

2.2 Rotation detection

The third line `minimumRotation` controls the thresholds for rotation detection. It works by detecting the rate of rotation. Faster rotation gives a larger absolute value, so if you need it to be more sensitive, you need a smaller absolute value (3 and -3 are smaller absolute values than 2 or -2).

3 Tuning

To perform tuning you run the `mlExplore.py` program file by typing

```
~/$ python3 mlExplore.py
```

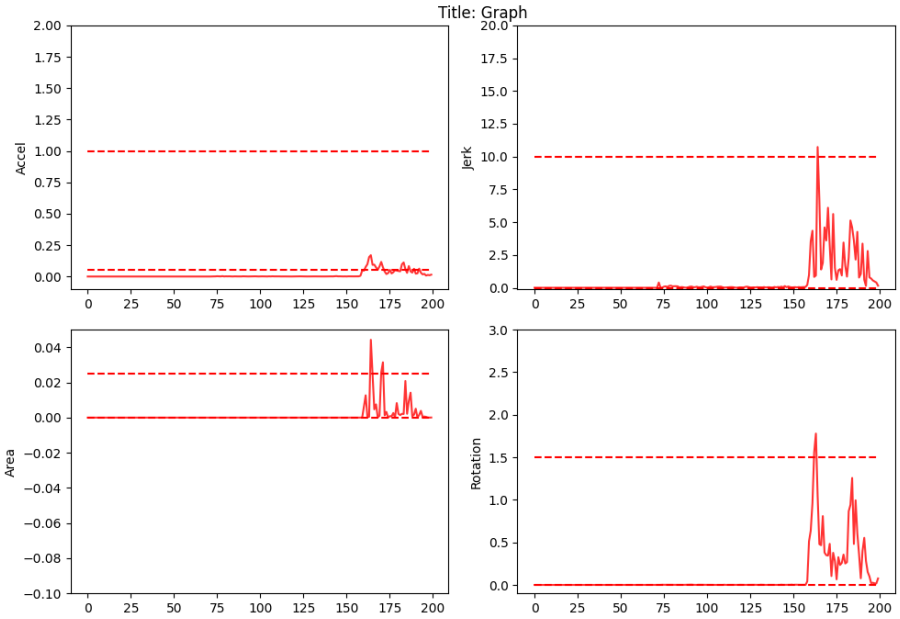

Figure 1: Datastream of acceleration data from SensorUDP.

The program should detect your IP automatically. You then need to connect your phone using the SensorUDP app. Once connected the program will display a data stream (Figure 1).

The top left data stream labelled Accel is the primary method of detecting step and anytime this graph has a top it will check thresholds. The thresholds are displayed as dotted lines for an upper and a lower boundary. The three other streams are auxiliary for a finer tuning of the step counter and can be helpful if step detection proves difficult using only the Accel variable.

When you run the program, the program will display counters for steps, left rotation and right rotation in the terminal window as shown in code below.

```
Computer name is : MyComputer-01
Receiver IP:  190.168.100.10
Port:  6000
Steps:  3  Left:  2  Right:  2
```

3.1 Tuning the step counter

When tuning the step counter it is recommended to be two people. Once you get `mlExplore.py` to run with your phone, you should get a window like Figure 1 to display tolerances and the datastream. As mentioned before, the step counter only checks for a step when Accel is on a peak. When you take a step, notice

where the peak is. If the peak is below the lowest line, you need to decrease the minimumStep value in the Accel column of `mlExploreSettings.txt`. In the same manner, if the peak is above the top dotted line, the maximumStep value in the Accel column needs to increase.

When you need to change the values in the Accel column of the `mlExploreSettings.txt` file, use a text editor to input new values. Before changing the values.

To tune the program step by step:

- Start `mlExplore.py`
- Connect the phone using SensorUDP
- Take a step and see where the peak falls in the Accel window
- Stop the program by first closing the sensor feed, then pressing `Ctrl-C` in the terminal
- Open `mlExploreSettings.txt` using a text editor
- Change the values in the Accel column
- Rerun the program and repeat

While Accel is the primary means of detecting the steps, the three other variables can be helpful. Rotation, for instance, can be used to filter out step events while attempting to trigger the rotation detector, for instance. In the same manner, Jerk can be used to filter out events like bumping the phone and Area works in a similar manner to Accel, but are more sensitive to low values, while Accel are more sensitive to large values.

3.2 Rotation detection

If the rotation detector is too sensitive, increase the values of the minimum-Ration line in `mlExploreSettings.txt` file. The procedure for step detection follows that of step detection, but for rotation instead. The values you need to change is the values in `mlExploreSettings.txt` at the end.

```
# Rotation tolerances:
#                Left       Right
minimumRotation  3          -3
```

For greater sensitivity change the numbers to closer to zero and opposite for lower sensitivity.

4 Using the step counter in other programs

If you want to use the step counter in a different program you can do so easily. You first need to import the step counter. To import the step counter into a python project write:

```
import mlExplore as mlE
```

With the step counter imported, you need to set up the data stream by writing:

```
sock = mlE.socketUDP()
dataStream = mlE.FeatureBox()
```

Now the datastream is ready to accept information from your phone. However, when accepting data we need to update whenever there is new data. To make this happen write the following loop:

```
pData = ''
while True:
    data = sock.recv(1024) # buffer size is 1024
    # If new data is same as old data, start the loop over and do nothing
    if data == pData:
        continue
    # if the data is new, the program will continue from here
    dataStream.update(data)
```

The `dataStream.update(data)` command updates the datastream with new data. Now we need to check to see if we have a step, left rotation or right rotation. This is done simply with the `dataStream.detectStep()`, `dataStream.detectRotLeft()` and `dataStream.detectRotRight()` commands. The step detection loop will end up looking like

```
import mlExplore as mlE
sock = mlE.socketUDP()
dataStream = mlE.FeatureBox()
pData = ''
while True:
    data = sock.recv(1024) # buffer size is 1024
    if data == pData:
        continue
    dataStream.update(data)
    if dataStream.detectStep():
        # Do stuff if step is detected
    elif dataStream.detectRotLeft():
        # Do some other stuff if left rotation is detected
    elif dataStream.detectRotRight():
        # Do some other stuff still if right rotation is detected
```

This code lets you do stuff depending on whether there is a step, left rotation or right rotation detected.