# Weekly report: June 13-June 19, 2015

Ivan Rios S.

June 19, 2015

## 1 Goals for the week

- Implement trip detection algorithm based on the new structure agreed on last week.

## 2 Activities

- Implement trip detection algorithm incrementally introducing a new variable each time.
- Analyze the results obtained in each iteration.
- Optimize results based on the modification of different parameters

## 3 What I learnt

- There is some noise, generally random high values, in the group of data collected every minute; for this reason, I chose to design the algorithm using a sliding window approach instead of the original point-by-point approach. With this approach, it is easier to identify random noise generated in the data. This noise can be identified by continually updating the data and confirming if these high values are showing up consistently or they just happened during a small period of time.
- The number of values collected every minute varies from 15 to 22 which means that within a group of this size we could expect to have random variations and hence, it is important to consider several minutes of data as part of each window in order to guarantee that the data is not noise. For this reason, in order to cover at least 4 minutes of data at the time; the selected size of the window (number of values analyzed at the time) is 90. Also, this value could have a higher value but 90 guarantees that the maximum time of inactivity (no movement) is not surpassed; in this case, being consistent with the previous

- For the sliding window of values considered every time, I created an object that contains all the data collected, and also some summary statistics such as average, standard deviation, number of values over the limit established for that particular object, among others. This object is instantiated for each one of the variables (eg. gyroscope movement, charging current, etc) and can be modified so that we have data available as required. This approach will be useful in future work, since inheritance can be used to add functions required for specific type of data that requires special modifications (from the technical point of view this will be advantageous for whoever takes over the project).
- When each object is created, an upper limit is defined to identify when a record has passed the "normal" range of values that characterize a bike that is not moving. For example, the module of a gyroscope record varies from 0 to 0,15, and anything above that limit could be considered as movement.
- Each object also keeps track of the number of values in the window that surpass the previously mentioned limit. For example, in the case of gyroscope data, the object tracks the values of the module that are over the value of 0,15. Since the variables used in this stage do not show much variation when the bike is not moving, we can rely on this data to define if the bike is moving. The algorithm allows a maximum number of values (out of all the ones in the window) to be over the limit before we can conclude (from that variable's data) that real movement exists.
- Based on the proposed approach, I implemented the algorithm in the following way:
  - I started by considering only charging current, to identify if the battery is being charged, and the module of the movement detected by the gyroscope. I started with gyroscope since that is the most reliable data and I wanted to determine the accuracy that it could offer by itself.
  - Then, I added linear acceleration as the next parameter. This parameter was used as a confirmation of the conclusions obtained from the gyroscope data, and also to be able to detect movement in a faster manner. This means that when either the gyroscope or the accelerometer detected movement, I started a "potential trip" with the earliest time at which movement was detected (by either one of the two sensors).
  - Finally, I added average speed of a trip. I used the average since we concluded that the "instantaneous" speed was not reliable enough to get conclusions, and also because we needed to have a validation in order to filter out trips that do not meet the criteria of a regular bike trip (eg. when the bike is mounted on a moving car). Speed is

the most appropriate variable since total distance and times,can vary from participant to participant, and from trip to trip; therefore, there would not be completely accurate value to identify this kind of trips. For the algorithm, I defined that the maximum average speed of a trip, to be considered a bike trip, should be 30 km/h; this value was obtained from the maximum speed allowed by the electric system of the bike. Even though, this speed can be surpassed for periods of time, the average should not go over this limit on a regular bike trip.

- The following table summarizes the results obtained by the algorithm:

| | 27-Apr | 29-Apr | 1-May | 6-May | 7-May | 8-May | 11-May | 13-May |
|---|---|---|---|---|---|---|---|---|
| Expected | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 |
| Gyro/Current | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 |
| +Acceleration | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 |
| + Avg Speed | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 |

For this analysis, I used the data from Professor Golab that was introducing more challenges to the previous algorithm. Specifically, a particularly difficult date was May 6th since it includes a long trip, a short trip, and a trip with the bike in the car. From this data, we can obtain the following conclusions:

- The accuracy of the trips calculated is very high. I manually checked trip by trip (down to the minute) and was able to confirm that the results were completely accurate with no false positives.
- Gyroscope data and charging current are very accurate by themselves. With only these two variables the algorithm was able to identify all the trips but it was not able to filter out the car trip.
- Adding a new variable (acceleration in this case) slightly increases the accuracy but the overall results are the same. For this reason I did not consider necessary to implement more variables.
- Average speed is required and accurate when filtering out trips on a car. In this case, after including this variable, the additional trip identified on May 6th was taken out of the final results.

Finally, some other parameters had to be defined (these parameters are necessary in any approach chosen for this algorithm). The parameters are:

- Minimum length of a trip (seconds): minimum duration of constant movement to actually consider it a trip. The current value is 300 seconds.
- Maximum time allowed without movement (seconds): this parameter is used to determine if sequential trips should be merged together. The value

chosen is 280 seconds. An example of this would be that if the algorithm finds two trips with a pause between them of under 280 seconds, it will join the trips, and consider them as one since the length of that pause is allowed to happen within the same trip.

As we can see, this parameters are subjective and depend on the definition of a trip so I tried to choose values that were consistent with the previous algorithm. However, they can easily be modified any time.

## 4 Proposed goals for next week

- Make new algorithm available in the beta section of the web page
- Complete documentation of the algorithm.