UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

DESIGN AND IMPLEMENTATION OF AN ALGORITHM FOR ESTIMATION OF WALKING DISTANCE USING THE EMBEDDED SMARTPHONE SENSORS

A Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Sergi Serra Soteras

In partial fulfilment

of the requirements for the degree in

Telecommunication systems **ENGINEERING**

Advisors: Miguel Ángel García González

Federico Guede Fernández

Barcelona, January 2016

Abstract

Nowadays the importance of the physical activity in our health is known for everyone. For that reason the need of accessible tools capable to adequately control it is growing. Given the fact that the smartphone, provided with a very wide variety of sensors, has become in an almost essential element on the daily routine is the ideal device to make the measures.

In this project it has been provided an algorithm for step counting and distance estimation that uses the data collected for a smartphone. Trying to improve the precision in the step counting, it has been incorporated the rotation vector and gyroscope sensors to the common algorithm that only uses the accelerometer. The travelled distance is computed through a first estimation of the step length that is measured using the GPS where there is a better reception.

The precision for the step counting has been slightly improved, even so, it seems that it still can be improved, especially for the cases with the lowest speed. Although proposed distance estimation has not achieved the desired accuracy, the obtained results are only wrong in concrete cases and are promising.

Resum

Avui dia la importància que té l'activitat física en la nostra salut és coneguda per tothom. Per aquest motiu creix la necessitat de eines accessibles capaces de portar-ne un control adequat. Ja que el *smartphone*, provis de una amplia varietat de sensors, s'ha convertit en un element quasi indispensable en la vida quotidiana és el dispositiu ideal per fer les mesures necessàries.

En aquest projecte s'ha proporcionat un algorisme que fent ús de les dades recollides per un *smartphone* és capaç de estimar les passes realitzades i la distància recorreguda. Per contar les passes, s'han incorporat els sensors vector de rotació i giroscopi als algorismes més comuns basats únicament amb l'acceleròmetre amb la intenció de millorar la seva precisió. La distància recorreguda es calcula a través d'una primera estimació de la mida mitjana de les passes, que es mesura amb el GPS durant el tram on hi ha la millor recepció.

La precisió de les passes contades s'ha aconseguit millorar lleugerament, tot i així, sembla que encara hi ha marge de millora, sobretot per a les velocitats més baixes. Tot i que la estimació de distància proposada no ha aconseguit la precisió desitjada, els resultats obtinguts només s'allunyen en casos concrets i són prometedors.

Resumen

Hoy en día la importancia de la actividad física en nuestra salud es de sobra conocida. Por este motivo crece la necesidad de herramientas accesibles que permitan llevar un control adecuado. Dado que el *smartphone*, provisto de una amplia variedad de sensores, se ha convertido en un elemento casi indispensable en la vida cuotidiana es el dispositivo ideal para realizar las medidas.

En este proyecto se ha proporcionado un algoritmo que usando los datos recogidos por un *smartphone* es capaz de estimar los pasos realizados i la distancia recorrida. Para contar los pasos se han incorporado los sensores vector de rotación y giróscopo a los algoritmos más comunes que usan únicamente el acelerómetro con la intención de mejorar su precisión. La distancia recorrida se calcula a través de una primera estimación de la longitud media de los pasos, que es medida con el GPS durante el tramo de mejor recepción.

La precisión de los pasos contados se ha conseguido mejorar ligeramente, aun así, parece que todavía hay margen de mejora, sobre todo para la velocidades más bajas. Aunque la estimación de la distancia propuesta no ha conseguido la precisión deseada, los resultados obtenidos solo se alejan en casos concretos y son prometedores.

Acknowledgements

I would like to express my gratitude to my advisors Miguel Ángel García González and Federico Guede Fernández for their guidance, for aiding me in the decision making, and also for helping me find solutions to the different issues encountered while developing and testing the algorithm.

I would also like to thank all the volunteers, without whom I would not have had the measurements necessary to develop this thesis.

Revision history and approval record

Revision	Date	Purpose
0	18/01/2016	Document creation
1	22/01/2016	Document revision
2	24/01/2016	Document revision

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Sergi Serra Soteras	sssoteras@gmail.com
Miguel Ángel García González	miquel.angel.garcia@upc.edu
Federico Guede Fernández	federico.guede@upc.edu

Written by:		Reviewed and approved by:	
Date	18/01/2016	Date	24/01/2016
Name	Sergi Serra Soteras	Name	Federico Guede Fernández Miguel Ángel García González
Position	Project Author	Position	Project Supervisors

Table of contents

List of Figures

List of Tables:

1. Introduction

1.1. Objectives

The purpose of this project is to design an algorithm able to estimate the distance traveled by a phone using the data recollected from its sensors. The sensors used will be the accelerometer, the gyroscope, the orientation and rotation sensors and, occasionally, the GPS (Global Positioning System). This project consists on counting the steps of the smartphone's carrier and estimate the distance travelled.

The project main goals can be summarized as follows:

1. To estimate a distance with a 5% error rate even at indoor (no GPS) places.
2. To combine the data obtained from the different sensors to decrease the error rate.
3. To compare the results with other step-counting algorithms.
4. To provide an accurate tool to control the physical activity available for every kind of user.

1.2. Requirements and specifications

This project will generate a Matlab program for steps counting and estimating the distance travelled from data recollected from smartphone sensors. In order to do that estimation, it has been required to implement a data base with the different kind of movements of several people at different speeds.

The maximum desired error rate for the distance calculation is a 5% of total distance walked in any case.

1.3. Methods and procedures

All the data used and needed to run the algorithm is previously collected with an Android application developed for the Electronic Engineering Department of UPC [8]. Also, the algorithm uses a Hodrick-Prescott filter implemented on a Matlab function developed by Wilmer Henao from Universidad de los Andes, Colombia.

1.4. Work plan

Work packages:

<table>
<tr><td>Project: Bibliography research</td><td colspan="2">WP ref: (WP1)</td></tr>
<tr><td>Major constituent: Documentation</td><td colspan="2">Sheet n of m</td></tr>
<tr><td rowspan="2">Short description:
Research for existing algorithm of distance estimation especially the ones that don’t use GPS to know the typical failures and limitations of them.
Also, find the needed information to understand perfectly the functionality of the used sensors.</td><td colspan="2">Planned start date: 14/09/2015
Planned end date: 05/10/2015</td></tr>
<tr><td colspan="2">Start event:
End event:</td></tr>
<tr><td>Internal task T1: Sensors documentation

Internal task T2: Interesting algorithms documentation</td><td>Deliverables:
Project Proposal and Work Plan</td><td>Dates:</td></tr>
</table>

<table>
<tr><td>Project: Data-base elaboration</td><td colspan="2">WP ref: (WP2)</td></tr>
<tr><td>Major constituent: Data compilation</td><td colspan="2">Sheet n of m</td></tr>
<tr><td rowspan="2">Short description:
Save the general data obtained from the sensors with different kind of movements and persons.</td><td colspan="2">Planned start date: 05/10/2015
Planned end date: 13/12/2015</td></tr>
<tr><td colspan="2">Start event:
End event:</td></tr>
<tr><td>Internal task T1: Walk/Run data compilation</td><td>Deliverables:</td><td>Dates:</td></tr>
</table>

<table>
<tr><td>Project: Algorithm implementation</td><td colspan="2">WP ref: (WP3)</td></tr>
<tr><td>Major constituent: Matlab programing</td><td colspan="2">Sheet n of m</td></tr>
<tr><td rowspan="2">Short description:
Implementation on Matlab of an algorithm that estimates a distance calculating the number of steps and the distance of each step using smartphone’s sensors.</td><td colspan="2">Planned start date: 05/10/2015
Planned end date: 27/12/2015</td></tr>
<tr><td colspan="2">Start event:
End event:</td></tr>
<tr><td>Internal task T1: Reading data

Internal task T2: Counting steps

Internal task T3: Distance estimation</td><td>Deliverables:
Critical Design review</td><td>Dates:</td></tr>
</table>

<table>
<tr><td>Project: Algorithm comparison</td><td colspan="2">WP ref: (WP4)</td></tr>
<tr><td>Major constituent: Technical comparison</td><td colspan="2">Sheet n of m</td></tr>
<tr><td rowspan="2">Short description:
Verification of the algorithm compared with the expected error rate and also checking if the accuracy has been improved against other algorithms.</td><td colspan="2">Planned start date: 28/12/2015
Planned end date: 11/01/2016</td></tr>
<tr><td colspan="2">Start event:
End event:</td></tr>
<tr><td>Internal task T1: Verification

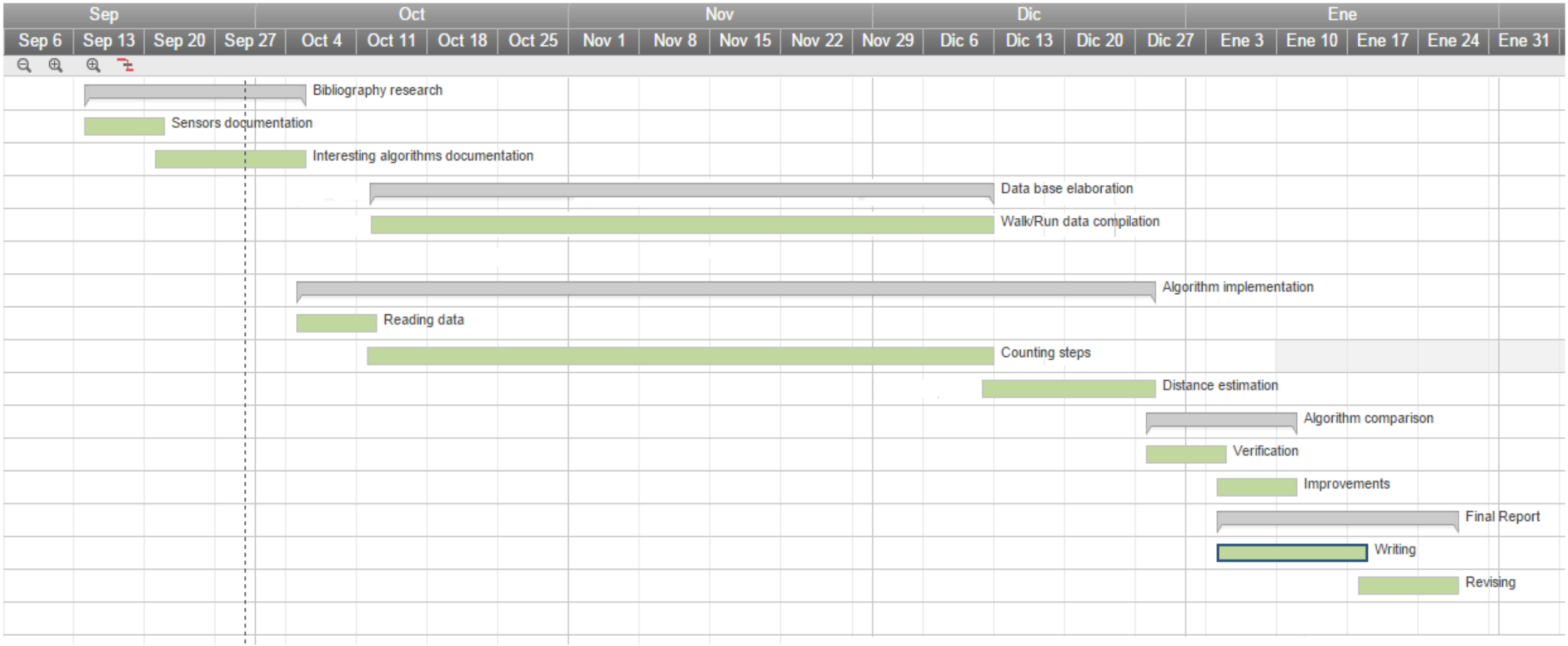
Internal task T2: Improvements</td><td>Deliverables:
Final Matlab code</td><td>Dates:</td></tr>
</table>

<table>
<tr><td>Project: Final Report</td><td colspan="2">WP ref: (WP5)</td></tr>
<tr><td>Major constituent: Writing</td><td colspan="2">Sheet n of m</td></tr>
<tr><td rowspan="2">Short description:
Writing and revising a Final Report.</td><td colspan="2">Planned start date: 01/01/2016
Planned end date: 27/01/2016</td></tr>
<tr><td colspan="2">Start event:
End event:</td></tr>
<tr><td>Internal task T1: Writing

Internal task T2: Revising</td><td>Deliverables:
Final Report</td><td>Dates:</td></tr>
</table>

Milestones:

WP#	Task#	Short title	Milestone / deliverable	Date (week)
WP1	Task1	Sensors documentation		1
WP1	Task2	Interesting algorithms documentation	Project Proposal and Work Plan	2-3
WP2	Task1	Walk/Run data compilation		4-13
WP3	Task1	Reading data		4
WP3	Task2	Counting steps	Critical Design review	5-13
WP3	Task3	Distance estimation		13-15
WP4	Task1	Verification	Final Matlab code	16
WP4	Task2	Improvements		17
WP5	Task1	Writing		17-18
WP5	Task2	Revising	Final Report	19-20


Sep
Oct
Nov
Dic
Ene
Sep 6
Sep 13
Sep 20
Sep 27
Oct 4
Oct 11
Oct 18
Oct 25
Nov 1
Nov 8
Nov 15
Nov 22
Nov 29
Dic 6
Dic 13
Dic 20
Dic 27
Ene 3
Ene 10
Ene 17
Ene 24
Ene 31
Bibliography research
Sensors documentation
Interesting algorithms documentation
Data base elaboration
Walk/Run data compilation
Algorithm implementation
Reading data
Counting steps
Distance estimation
Algorithm comparison
Verification
Improvements
Final Report
Writing
Revising

1.5. Deviations and incidences

The most relevant incidence has been to discard the orientation sensor due to its low accuracy and the occurrences of outliers in some tilting angles of the smartphone. Instead of the orientation sensor, it is used the rotation vector to compute the vertical direction.

The time spent on the algorithm implementation produced a delay on the database elaboration. The data was collected according to the needs of the algorithm under development instead of doing it at the beginning.

The end date for the database elaboration package increased and because of that the data of the movements inside vehicles was not included.

In the package of the algorithm implementation, the task of step counting took more time because it needed the elaboration of an initial algorithm using only the accelerometer and then, the incorporation of the other sensors. The time expected for the task of distance estimation decreased, but not the time of the package.

The initial filter proposed was a high order low-pass FIR filter with the objective of erase completely the high frequencies but this filter did not work properly especially for the cases with a higher speed, this filter was replaced by a Hodrick-Prescott filter in order to obtain the smoothed curve desired of the signal data.

2. State of the art of the technology used or applied in this thesis:

2.1. Review of Literature

Over the years we have realized the importance that the physical activity has on our health or on the capacity of the sportsmen. Because of that, the presence of the technology used to monitor the physical activity is becoming greater as same as its accuracy.

One of most accessible technology used to do that is the smartphone, but it is not very reliable for distance measuring, especially when the GPS is not available and the distance is calculated by counting the steps using the accelerometer. So, we consider the possibility to improve it by making use of more sensors and combine them to avoid counting errors.

There is a long list of sensors embedded in the modern smartphones, as we can see on a simple research on internet [1] or downloading specific apps that list the sensors available in smartphones where are installed. Even though, most of this sensors can't be used to monitor the physical activity. So, to step counting we focused on the accelerometer, the magnetometer and the gyroscope plus the GPS for distance estimation.

As it is said in [1], some smartphones have a pedometer embedded, the sensor that directly gives steps walked, and the alternative to this sensor is the accelerometer. In [2] the comparison of two of those kind of sensors show us that both have limitations in different conditions, but both have a similar magnitude error. According to this text, the error when counting steps using the accelerometer is basically produced by counting mistakenly as steps some other movements of the device. Considering this information, the algorithm developed in this project is based on the accelerometer with the intention to reduce the error using information from other sensors.

Most of the step counting systems from smartphones are based on the accelerometer. In the fourth comment from the forum [3] is summarized two of the most common algorithm used in this situation. The first one is the most common and simple of all of them, it consists on count the relative maximums of a signal that is the result of filtering the modulus of the acceleration data. The second algorithm is based on the comparison between the Fast-Fourier Transform of the modulus of the accelerometer data with other training data of known walking signals. The project's algorithm starts from the idea of the first option with the changes needed for improve the accuracy, the second one was not valid because it beeds previous information from the person who is going to make the measurements.

Also, there are some papers that already talk about the use of more sensors to improve the precision of the algorithm based on the accelerometer for step counting and distance estimation. In [4] uses the gyroscope and the magnetometer to obtain the perpendicular direction to the floor to compensate the tilt whereas the accelerometer is used to compute steps. Working on this direction it can avoid some of the false steps counted for the algorithm.

The Android devices are provided with some different motion sensors [5] that can be a simple alternative to the combination of the magnetometer and other sensors. The rotation vector is one of this sensors that uses the magnetometer to give and estimation of the relative position of the smartphone. In [5] is explained the meaning of the values from it, the rotation vector has four outputs, one for each coordinate of the vector plus the time. The modulus of the vector is the sinus of half the angle that the axis of the accelerometer is rotated from a reference position. The direction of the vector is the axis of the rotation. In the reference position the 'y' axis points to the magnetic north of the earth, the 'z' axis points toward the sky and the 'x' axis points to the east.

The Orientation sensor is another software based positioning sensor explained in [6]. It uses the magnetometer in combination with the accelerometer to compute the relative position of the smartphone using the magnetic north as the reference. Its outputs are azimuth, pitch and roll that are the angles between the accelerometer axes and the north. In [6] is also said that the accuracy and precision of this sensor is diminished because of the heavy processing that is involved.

The estimation of the distance walked without the GPS is obtained by the multiplication of the distance of one step and the total steps done. Most of the studies found to estimate the distance of one step need lots of known training data used to compare with the input, so without a big database this kind of estimation has a poor accuracy.

In the article [7] it is studied the correlation between vertical bounce and step length. The system uses only the accelerometer and computes a double integral to determine the vertical displacement, first, and then the step length through a simple trigonometry relation. The final results expose that has not the accuracy expected, the estimation was too much different between the different speed and individuals.

At the end, one of the most reliable and useful information was the given for my advisors for their experience and knowledge.

2.2. Making decisions

Due to the information recollected, the path followed for the algorithm development began with the implementation of simplest algorithm using only the accelerometer data. This serves to compare accuracy of the final results with the initial one. The second step was compute the vertical acceleration instead of work with the modulus with the rotation vector. Then the last sensor incorporated to improve the accuracy was the gyroscope used to determine if some kind of possible steps were caused by relative movements instead of steps in straight direction. To estimate the distance, the best solution was use the GPS in the areas with good signal to compute the step length to finally compute the total distance even in the shadows zones with no GPS.

3. Project development:

3.1. Data collection

The first step in the project begins with the collection of the needed data to run the algorithm and test the final results. All the data was obtained with the same smartphone application and the same device.

The validation of the final results is based on the elaboration of a database composed for the data collected for five different people and for four sensors: the accelerometer, the rotation vector, the gyroscope and the GPS.

To elaborate the database each people had to do two different exercises, both done on the same circuit with the same distance travelled. The first exercise consisted on walk at a comfortable speed with the smartphone in the hand while looking at it. The purpose of this exercise is to test the algorithm behaviour in the optimal, but possible, conditions.

In the second exercise, each people walk the same circuit in a different speed. Four of them walk slow with shorter steps and the other is done while running. Also, the position of the smartphone was changed in four cases, two with the smartphone in the pocket and two with the smartphone in the hand but, this time, in a relaxed position. With this exercise is tested the algorithm behaviour in some different, possible and difficult circumstances.

The circuit consist in walking an outdoor flat terrain during forty-two meters, two times on opposite directions. So, the circuit include a straight line, a 180 degree turn and another straight line for a total of eighty-four meters walked.

3.2. Involved sensors

3.2.1. The accelerometer

This sensor is the base of the step counting algorithm. It provides the acceleration that suffers in the three different axes of the space called ‘x’, ‘y’ and ‘z’. The positive ‘x’ goes from the screen to the right in the same plane, the ‘y’ axis goes from the screen to the up side in the same plane and the ‘z’ axis is in the perpendicular plane to the screen and comes out -see Figure 1-.

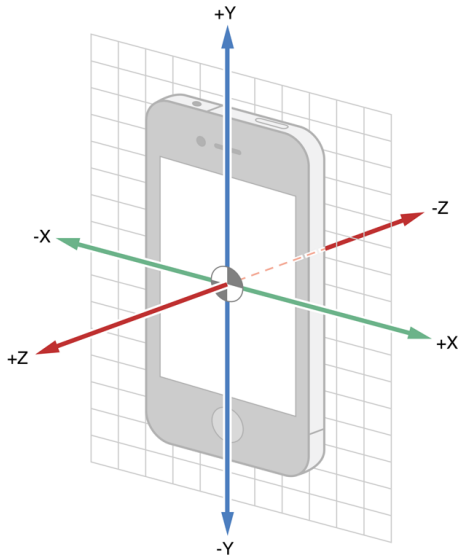


Figure 1 Accelerometer axes on the smartphone

In the smartphone's accelerometer the data is given in meters per squared second. When it is resting the modulus of the acceleration fluctuates around the value of the gravity acceleration that is always influencing the measured acceleration.

The accelerometer is good option to monitor the device motion. Each step with the device represents an increment followed by a decrement of the acceleration. Considering this behaviour, to count the steps all the acceleration local maximums in a certain bandwidth are studied to determinate if they are produced for a step or not.

3.2.2. The rotation vector

The problem with the accelerometer is that the directions of its axes changes with the position of the mobile. The first solution to avoid false positives of the step counting algorithm is to work only with the acceleration of the vertical direction. To make this possible, it is needed to know the relative position of the smartphone, given by the rotation vector.

The rotation vector represents the orientation of the device as a combination of an angle and an axis, in which the device has rotated through an angle θ around an axis (x, y, or z).

The three elements (Rx, Ry, Rz) of the rotation vector are expressed as follows:

$$Rx = x*\sin(\theta/2)$$

$$Ry = y*\sin(\theta/2)$$

$$Rz = z*\sin(\theta/2)$$

Where the magnitude of the rotation vector is equal to $\sin(\theta/2)$, and the direction of the rotation vector is equal to the direction of the axis of rotation –Figure 2-.

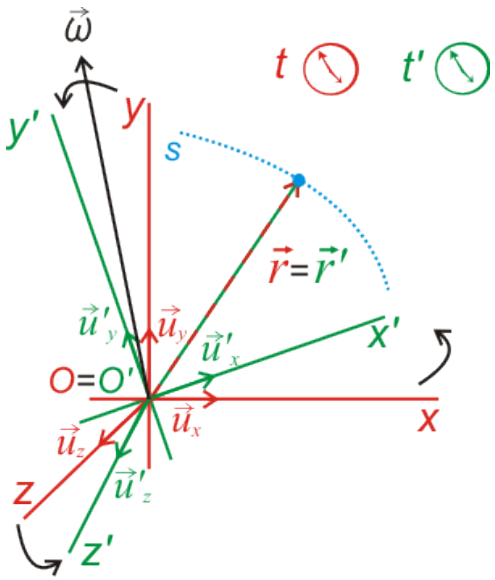


Figure 2 Rotation vector

The three elements of the rotation vector are equal to the last three components of a unit quaternion (cos(θ/2), x*sin(θ/2), y*sin(θ/2), z*sin(θ/2)). The elements of the rotation vector are unitless. The x, y, and z axes are defined in the same way as the acceleration sensor. The reference coordinate system is defined as a direct orthonormal basis and this coordinate system has the following characteristics:

- X is defined as the vector product Y x Z. It is tangential to the ground at the device's current location and points approximately East.
- Y is tangential to the ground at the device's current location and points toward the geomagnetic North Pole.
- Z points toward the sky and is perpendicular to the ground plane.

3.2.3. The gyroscope

This sensor measures the rate of rotation in rad/s around a device's x, y, and z axis. This measurement helps to differentiate the relatives movements from the real displacement in the motion detected for the accelerometer.

In this project, high values of this sensor are considered the cause of false steps that are discarded on the counting.

3.2.4. The GPS

As it is known the GPS is the sensor capable to determinate the position of the device on the earth surface. The GPS receptor is connected with a net of twenty-four satellites that

covers all the earth surface. To calculate the position the device receive a signal from at least four of those satellites that contains the identification and the time when the signal was sent of all of them. With this information is obtained the time needed for the signal to come to the receptor and, so, the distance between the device and each satellite. Knowing the position of the satellites is it possible to determinate the position of the device.

Besides the position (in latitude and longitude), the smartphone's GPS sensor also provides the accuracy of the measurement in each moment. That value depends on the number of satellites from which the device is receiving signals and the value represents maximum error of the position calculation.

Because of the need of the satellites communication the GPS is often useless in indoor locations, or at least it has a very bad accuracy. The value of the accuracy in this project is used to determine the best moment to compute the step length, then the distance travelled in indoor location can be estimated.

3.3. The filter

The signal obtained with the sensors are a very noisy especially in the accelerometer sensor. This noise could cause some false steps detections, so a low-pass filter is needed to reduce that noise. Therefore, the filter used in this project is the Hodrick-Prescott filter that is used in macroeconomics to determinate the tendency of some data. This filter erases the cyclical component and obtains a smoothed curve of a signal. Moreover, the curve obtained depends on a parameter (λ) that is the relation between the cut-off frequency and the ample frequency. Consequently, -In Figure 3- we can see the result (in red) of applying the Hodrick-Prescott filter with the λ parameter adequately chosen to a random input (blue).

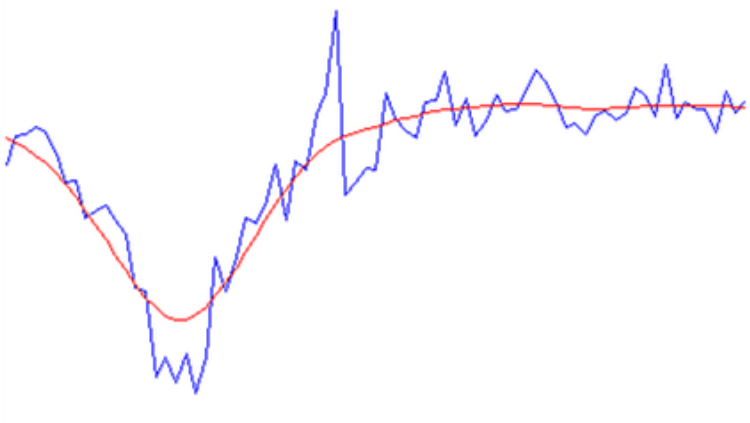


Figure 3 Example of Hodrick-Prescott filtering

3.4. Step counting

In the beginning, it was tried to work with the orientation instead of the rotation vector but the values obtained in some positions of the smartphone were not the expected. In some algorithms considered where counted local maximums in the signal of the rotation vector or the gyroscope combined with the accelerometer, but it could not be found a valid single value for the threshold. At the end, the best solution found was the one described below.

As mentioned above, the base of the step counting algorithm is the acceleration signal. Specifically, it is used the vertical acceleration. Therefore, the algorithm begins reading the data of the sensors saved on the different files generated for the Android application of the smartphone. The data from the acceleration is saved on the vectors 'ax', for x axis; 'ay', for y axis and 'az', for z axis. The components of the rotation vector are 'rx', 'ry' and 'rz'. The modulus of the gyroscope data is called 'mgy'. And for the GPS the vectors are 'LatDouble' (latitude), 'LongitudDouble' (longitude) and 'AccuracyFloat'.

To obtain the vertical acceleration for each sample of the acceleration vector is applied the rotation represented by the corresponding sample of the rotation vector. First, the angle of the rotation is computed from the modulus of the rotation vector, then is obtained the unity rotation vector 'ux', 'uy' and 'uz' that is the result of divide the rotation vector by its modulus.

$$ux = rx / \sin(\theta/2) \qquad uy = ry / \sin(\theta/2) \qquad uz = rz / \sin(\theta/2) \tag{1}$$

That unity vector is used for compute the rotation matrix (2).

$$R = \begin{bmatrix} \cos\theta + u_x^2(1-\cos\theta) & u_x u_y(1-\cos\theta) - u_z\sin\theta & u_x u_z(1-\cos\theta) + u_y\sin\theta \\ u_y u_x(1-\cos\theta) + u_z\sin\theta & \cos\theta + u_y^2(1-\cos\theta) & u_y u_z(1-\cos\theta) - u_x\sin\theta \\ u_z u_x(1-\cos\theta) - u_y\sin\theta & u_z u_y(1-\cos\theta) + u_x\sin\theta & \cos\theta + u_z^2(1-\cos\theta) \end{bmatrix}.^{1} \tag{2}$$

Once the matrix is computed, the Earth referenced acceleration is obtained by the multiplication of the matrix (R) with the acceleration vector ($\underline{a}$ =[ax, ay, az]').

$$\underline{a}' = R * \underline{a} \tag{3}$$

The vertical acceleration is the corresponding z axis of the non-rotation acceleration. Each sample of the vertical acceleration is saved on the vector 'av'.

With the vertical acceleration the algorithm selects the local maximums of the filtered signal that are greater than the threshold 10.6 and are counted as steps –Figure 4-. The chosen lambda parameter from the Hodrick-Prescott filter is 300.

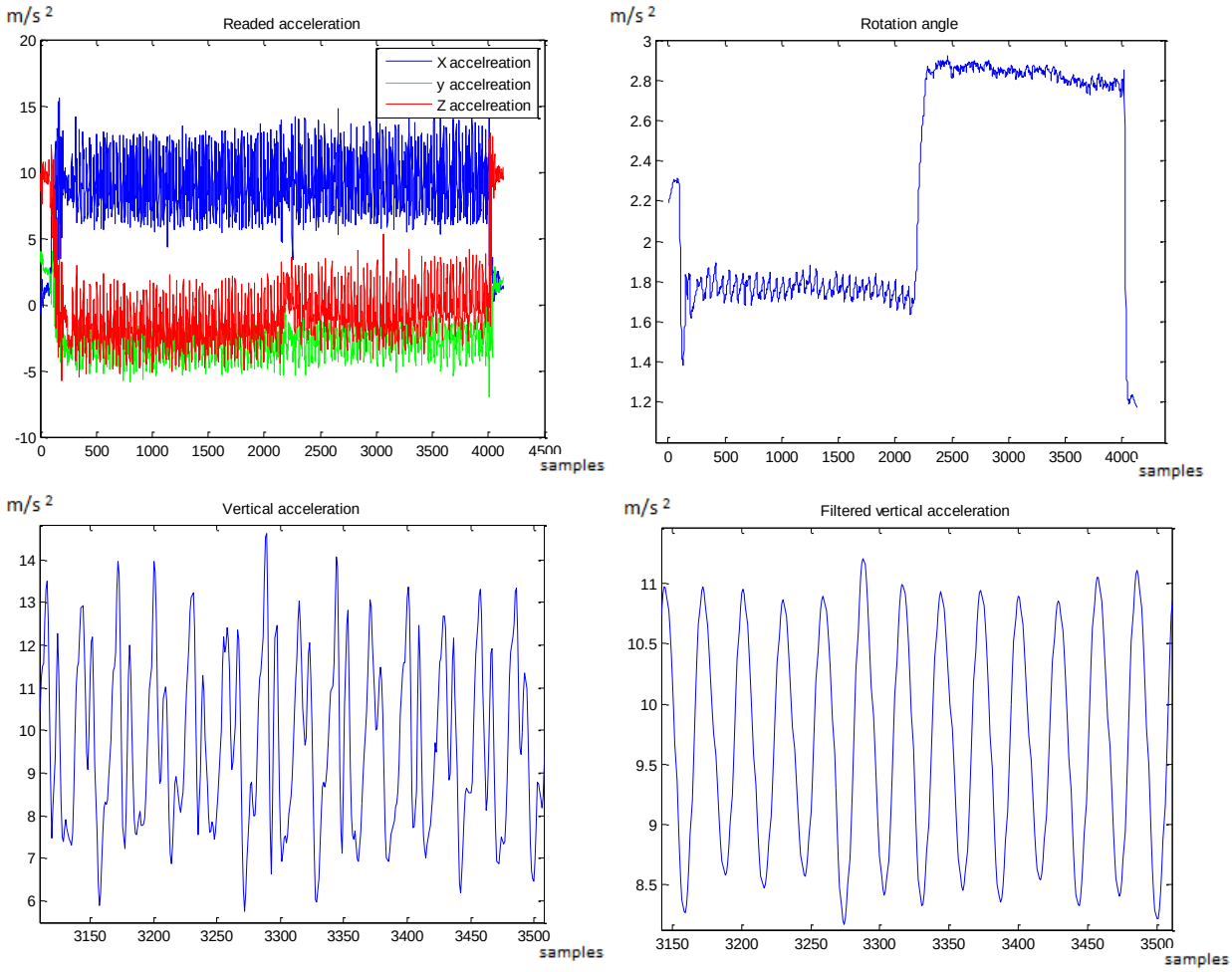


Figure 4 Counting steps example

There is also contemplated the maximums in the margin between 10.4 and 10.6. In this cases it is used the modulus of the gyroscope. Only the maximums without a value of the 'mgy' greater than 1 are counted as steps too, the other cases are considered the product of the relative motion of the smartphone. –Figure 5-

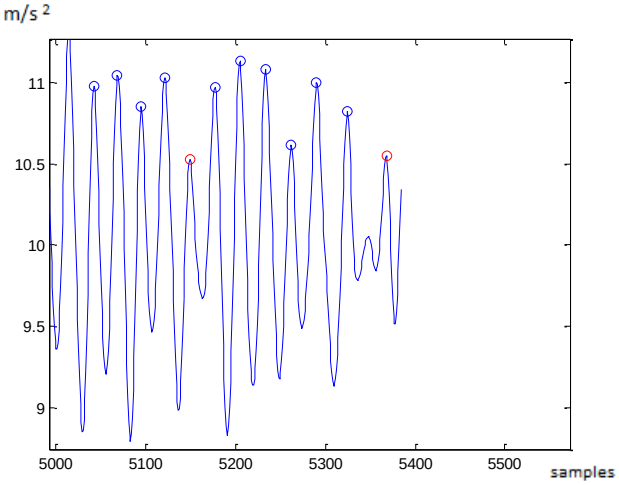


Figure 5 Counted extra steps

3.5. Distance estimation

Once the steps are counted the algorithm selects the best samples of GPS positioning, the ones with a greater accuracy (the minimum value of 'AccuracyFloat').

Taking the reference of the first good position is computed the distance with the other points to this reference –Figure 7-. To compute the distance between two points given their coordinates in latitude and longitude is used an approximated formula that consists on the multiplication of the Earth radius with the vector subtraction of both positions. With the objective to have the best precision in the calculation of the step length is selected the last point where the distance between the initial reference has not decreased at any moment before. The stretch between these two points would be used to compute the mean step length.

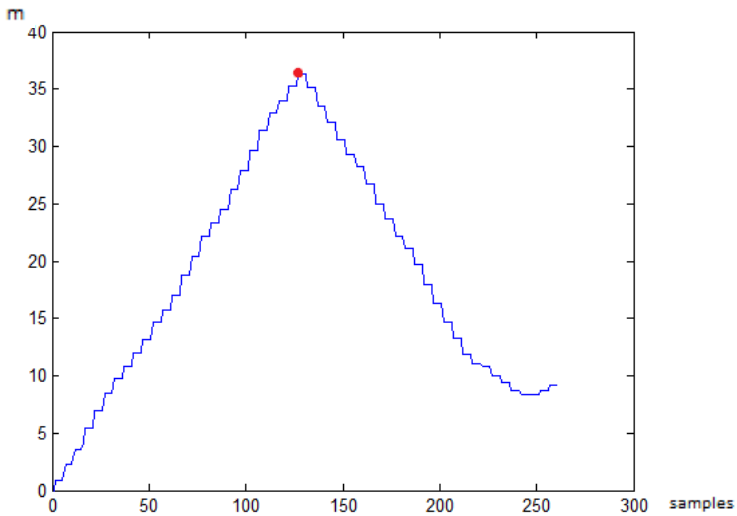


Figure 6 Evolution of the distance in the database circuit and the selected point to determinate the best stretch

The step length is obtained with the common known formula: *sl = distance / nº steps*. Because the time is known in the both limits of the stretch it can be counted the number of steps in that margin of time.

Finally, as it has been said, the total distance is computed with the multiplication of the step length times the total steps obtained.

4. Results

The results obtained for the algorithm are evaluated through the database described above. This database contains two different exercise of five different people. In –Table 1- the results from the first exercise are listed while in –Table 2- the results from the second exercise are found. Both tables show the comparison for step counting between the algorithm that uses all the sensors and a simpler one (based on the algorithm described in [3]) that only works with the acceleration, they also show the distance estimation and the relative error for each people.

In the first exercise, each participant walked at a comfortable speed while looking at the smartphone. In this case, it draws the attention the fact that there is no difference between the two step counting algorithms, both of them have a very good precision. The conditions in this exercise are considered the best possible and that is why the results from both algorithm are equal, even though, the error from the distance estimation seems to be not good enough for one of the subjects.

Subject	Real number of steps	Steps from obtained algorithm	Steps from a simpler algorithm	Travelled Distance (m)	Estimated distance (m)	Computed step length (m)	Relative error for step counting (%)	Relative error in distance estimation (%)
A	128	128	128	84	83.793	0.655	0	0.24
B	120	120	120	84	87.882	0.732	0	4.62
C	112	111	111	84	76.346	0.688	0.89	9.11
D	105	106	106	84	85.754	0.809	0.95	2.09
E	95	96	96	84	79.782	0.831	1.05	5.02

Table 1 Exercise 1 results

The conditions have been changed in the second exercise, the subjects change this speed and the mobile position. The participant C ran and the others walked slowly with shorter steps. The smartphone was positioned in the pocket for A and D and for B and E it was on the hand, but now, without looking at it. As we see in the table, there is a special case, A, where we have a very bad results. In this case the speed was too slow and the threshold used to count steps was a bit too high. Apart of it, the precision of the step counting are still

good and there is a small improvement between the algorithm that uses all the sensors and the other with only the accelerometer. The estimation of the distance is bad too when the steps are not correctly counted, furthermore, it seems to fail in another case with no special reason.

Subject	Real number of steps	Steps from obtained algorithm	Steps from a simpler algorithm	Travelled Distance (m)	Estimated distance (m)	Step length	Relative error for step counting (%)	Relative error in distance estimation (%)
A	197	189	185	84	76.251	0.403	4.06	9.22
B	166	166	168	84	87.138	0.524	0	3.74
C	73	73	73	85	85.278	1.168	0	0.33
D	147	147	148	84	94.737	0.644	0	12.78
E	130	131	132	84	82.567	0.630	0.77	1.71

Table 2 Exercise 2 results

The mean relative error with distance estimation is about 4.89 % of the total distance walked and the maximum error obtained is 12.78%.

5. Budget

Components	Cost/unit(hour)	Units(hours)	Final cost
Computer	400€	1	400€
Smartphone	400€	1	400€
Matlab Licence	0-2000€	1	2000€
Junior Engineer work	10€	300h	3000€

Table 3 Budget

The total cost taking the general Matlab license = 5800€

6. Conclusions and future development:

Seeing the results, it could be said that the accuracy of the distance estimation can still be improved. The goal in this point was to have a maximum relative error less than 5 % but it is not accomplished for the algorithm tested in this project, even so, the solution proposed was an original idea and cannot be definitely discarded to achieve the desired accuracy, as I see it the main idea could be useful in future improvements with the adequately changes. The principal limitation of this is idea is the need of a constant speed, but it is the same for all the indoor systems found on the research because it seems unavoidable the previous calculation of the step length.

Apart of it, the algorithm obtained for the step counting seems to have a good precision in general conditions, but it is true that the precision of the accelerometer is close to that. So, it has been achieved an improvement of the common algorithm used but it is not very significant for certain conditions.

Even though, it has been detected a problem in the step counting algorithm when the speed is too slow. We can see in the results that there is an important deviation in that case, some of the steps are missing because value of the acceleration is significantly lower than normal cases and the threshold used is not valid. So, one possible improvement in future developments could be the incorporation of a variable threshold that could depend on the mean values of the local maximums of the acceleration.

Bibliography:

[1] Manuel J. Gutiérrez. *"¿Cuáles son y para qué sirven los sensores de nuestros Android?". July 10, 2014.* [Online] Available: http://www.elandroidelibre.com/2014/07/cuales-son-y-para-que-sirven-los-sensores-de-nuestros-android.html. [Accessed: October 2015].

[2] *"Comparison of pedometer and accelerometer accuracy under controlled conditions".* Department of Exercise and Wellness, Arizona State University East, 7001 E. Williams Field Rd., Mesa, AZ 85212, USA. [Online] Available: http://www.ncbi.nlm.nih.gov/pubmed/12750599.

[3] [Online] Available: http://es.androids.help/q1091. [Accessed: September 2015].

[4] Ando, B., Baglio, S, Lombardo, C.O., V. Marletta. "An advanced tracking solution fully based on native sensing features of smartphone". *2014 IEEE Sensors Applications Symposium (SAS)*. Pp 141-144.

[5] Android Developers. "*Motion Sensors*". [Online] Available: http://developer.android.com/intl/es/guide/topics/sensors/sensors_motion.html [Accessed: October 2015].

[6] Android Developers. "*Position Sensors*". [Online] Available: http://developer.android.com/intl/es/guide/topics/sensors/sensors_position.html#sensors-pos-geomrot [Accessed: October 2015].

[7] Jim Scarlett. "*Enhancing the Performance of Pedometers Using a Single Accelerometer*". [Online] Available: http://www.analog.com/media/en/technical-documentation/application-notes/47076299220991AN_900.pdf. [Accessed: October 2015].

[8] Federico Guede-Fernández, Bernat Carbonés, Lluis Capdevila, Miguel Angel García-González, Juan Jose Ramos-Castro, Mireya Fernández-Chimeno. "Assessment of Energy Expended in Physical Activity by a Smartphone-Based System ". [Online] Available: http://link.springer.com/chapter/10.1007%2F978-3-319-11128-5_222.

[9] https://en.wikipedia.org/wiki/Hodrick%E2%80%93Prescott_filter

[10] https://en.wikipedia.org/wiki/Rotation_matrix

[11] http://es.mathworks.com/

Appendices (Matlab code):

Main Activity

```
clear all
close all

format long g

gyr = 'GyroscopeData-S-147.txt'; % Introduce the name of the gyroscope
data file to use
ro = 'RotationData-S-147.txt'; % Introduce the name of the rotation data
file to use
gpsfile = 'GPSDataSys-S-147.txt'; % Introduce the name of the gps data
file to use
accfile = 'AcelerometerData-S-147.dat'; % Introduce the name of the
acelerometer data file to use


[ t2, rx, ry, rz ] = OrRoGyDataRead( ro );
[ t3, gx, gy, gz ] = OrRoGyDataRead( gyr );
[ CharType, LatDouble, LongitudDouble, AltDouble, AccurayFloat,
timeLong, speed, bearing, timeSys ] = gpsRead( gpsfile );
le = length(t2);

[ax, ay, az, t]=accelData( accfile, '',  0);


if (length(t)> le)
    ax = ax(1:le);
    ay = ay(1:le);
    az = az(1:le);
    t = t(1:le);
elseif (le > length(t))
    le = length(t);
    rx = rx(1:le);
    ry = ry(1:le);
    rz = rz(1:le);
    t2 = t2(1:le);
end

if length(t3)<le
   gx = [gx zeros(1,le-length(t3))];
   gy = [gy zeros(1,le-length(t3))];
   gz = [gz zeros(1,le-length(t3))];
   t3 = [t3 t2(1,length(t3)+1:le)];
else
   gx = gx(1:le);
   gy = gy(1:le);
   gz = gz(1:le);
   t3 = t3(1:le);
end

mod = sqrt(ax.^2 + ay.^2 + az.^2);
modrotacio = sqrt(rx.^2 + ry.^2 + rz.^2);
mgy = sqrt(gx.^2 + gy.^2 + gz.^2);
```

```
th = asin( modrotacio ).*2; % |vectorRot| = sin(th/2)
ux = rx ./ sin(th/2);
uy = ry ./ sin(th/2);
uz = rz ./ sin(th/2);

for i = 1:le
    R = [cos(th(i))+(ux(i)^2)*(1-cos(th(i)))         ux(i)*uy(i)*(1-
cos(th(i)))-uz(i)*sin(th(i))     ux(i)*uz(i)*(1-
cos(th(i)))+uy(i)*sin(th(i));
      ux(i)*uy(i)*(1-cos(th(i)))+uz(i)*sin(th(i))
cos(th(i))+(uy(i)^2)*(1-cos(th(i)))        uz(i)*uy(i)*(1-cos(th(i)))-
ux(i)*sin(th(i));
      ux(i)*uz(i)*(1-cos(th(i)))-uy(i)*sin(th(i))    uz(i)*uy(i)*(1-
cos(th(i)))+ux(i)*sin(th(i))           cos(th(i))+(uz(i)^2)*(1-
cos(th(i))) ];

    ah1(i) = R(1,:) * [ax(i); ay(i); az(i)];
    ah2(i) = R(2,:) * [ax(i); ay(i); az(i)];
    av(i) = R(3,:) * [ax(i); ay(i); az(i)];
end

modhorzt = sqrt(ah1.^2 + ah2.^2);


x = hpfilter(mod,3e2);

plot(x, 'r')

figure
plot(t,av,'k')
hold on
plot(t,ah1,'r')
plot(t,ah2,'g')

avf=hpfilter(av,3e2);

figure
plot(avf)

posstepsx = find(x(2:length(x)-1)>x(1:length(x)-2) & x(2:length(x)-
1)>x(3:length(x)) & x(2:length(x)-1)>10.5)+1;
stepsmod=length(find(x(2:length(x)-1)>x(1:length(x)-2) & x(2:length(x)-
1)>x(3:length(x)) & x(2:length(x)-1)>10.5))
figure
plot(x,'r')
hold on
plot(posstepsx,x(posstepsx),'o')

possteps = find(avf(2:length(avf)-1)>avf(1:length(avf)-2) &
avf(2:length(avf)-1)>avf(3:length(avf)) & avf(2:length(avf)-1)>10.6 )+1;

posconflict = find(avf(2:length(avf)-1)>avf(1:length(avf)-2) &
avf(2:length(avf)-1)>avf(3:length(avf)) & 10.3<=avf(2:length(avf)-1) &
avf(2:length(avf)-1)<=10.6 )+1;
posnewsteps = [];
for i = 1 : length(posconflict)
    if 1 >= max(mgy(posconflict(i)-10 : posconflict(i)+10))
        posnewsteps(end+1) = posconflict(i);
```

```
    end
end

stepsavf=length(possteps)+length(posnewsteps)
figure
plot(avf)
hold on
plot(possteps,avf(possteps),'o')
plot(posnewsteps,avf(posnewsteps),'o','Color','r')


puntsgps = find(AccurayFloat == min(AccurayFloat));
rest = AccurayFloat(find(AccurayFloat ~= min(AccurayFloat)));
while length(puntsgps) < 2 & length(rest)>1
    puntsgps = sort([puntsgps find(rest == min(rest))]);
    rest = rest(find(rest ~= min(rest)));
end

if (length(puntsgps) > 1)
    latacc = LatDouble(puntsgps);
    lonacc = LongitudDouble(puntsgps);
    timeacc = timeLong(puntsgps);
    POS = [latacc; lonacc];
    v = ones(1,length(puntsgps));
    for i = 1 : length(puntsgps)
        v(i) = GPSdist(POS(:,i),POS(:,1));
    end
    ref = [find(v(2:end)<v(1:end-1)) length(v)];

    while v(ref(1))== v(ref(1)-1)
        ref(1) = ref(1) -1;
    end

    tram = find( t2 >= timeacc(1) & t2 <= timeacc(ref(1)) );

    stepstram = length(find(possteps >= tram(1) & possteps <=
tram(end))) + length(find(posnewsteps >= tram(1) & posnewsteps <=
tram(end)))

    DistTotal = (v(ref(1))/stepstram)*stepsavf
end
```

Aditional functions

```
function [ clmn1, clmn2, clmn3, clmn4 ] = OrRoGyDataRead( filename )

id = fopen(filename);
str = fgetl(id);

str = fgetl(id);
clmn1 = [];
clmn2 = [];
clmn3 = [];
clmn4 = [];
while str ~= -1
    dots=find(str==':');
    clmn1(end+1) = str2num(str(1:dots(1)-1));
    clmn2(end+1) = str2num(str(dots(1)+1:dots(2)-1));
    clmn3(end+1) = str2num(str(dots(2)+1:dots(3)-1));
    clmn4(end+1) = str2num(str(dots(3)+1:end));
    str = fgetl(id);
end

fclose(id);

end
```

```
function [ax, ay, az, t, date]=accelData( file, pathname,  optPopFile)
%function [ax, ay, az, t, date]=accelData(file, pathname, optPopFile)
%DESCRIPTION: Get the acceleration in the three axes
%INPUTS:

%    - file: Name of the file
%    - pathname: Path of the file

%    - optPopFile: 0-->Set file and pathname. 1--> Pop to chose the
desired
%    file
%RETURN:
%    - ax, ay, az: acceleration in the three axes (m/s^)
%    - t: Normalized timeStamp in seconds
%    - date: Start date data. Format equal to SQL. ex: 20140219181611
%Author: Bernat Carbonés
%Modification Date: 09/05/2014



%Load the Accelerometer data file

if optPopFile

    [file, pathname]=uigetfile('*.dat');
end

fileComplete = fullfile(pathname, file);
```

```
%Get start time in hours
t0=gethour(file);

fid=fopen(fileComplete);
a=fread(fid,'short','b');
fclose(fid);

ax=a(3:5:end)/1000;
ay=a(4:5:end)/1000;
az=a(5:5:end)/1000;
lm=min([length(ax) length(ay) length(az)]);
ax=ax(1:lm);
ay=ay(1:lm);
az=az(1:lm);

fid=fopen(fileComplete);
a=fread(fid,'ushort','b');
fclose(fid);
t=a(1:5:end)*65536+a(2:5:end);
t=(t-t(1))/1000;
clear a;
t=t(1:lm);

end



function t0=gethour(file)
%function t0=gethour(file)
%DESCRIPTION: Function done. Returns the time which was taken the file
in hours
%INPUTS:
%    - file: File to get the hour
%RETURN:
%    - t0: Time which was taken the file in hours
%Author: Federico Guede
%Modification Date: 27/02/2014


%file='AcelerometerData_13_03_21_16_05_08.dat';
%[C,count] = textscan(file, '%s %d %d %d %d %d %d', 'delimiter',
'_','EmptyValue', -Inf);
[C,count] = textscan(file, '%s %f %f %f %f %f %f', 'delimiter',
'_','EmptyValue', -Inf);
year=C{2};
month=C{3};
day=C{4};
hour=C{5};
minute=C{6};
second=C{7};
t0=hour+minute/60+second/3600;
```

```
function [ CharType, LatDouble, LongitudDouble, AltDouble, AccurayFloat,
timeLong, speed, bearing, timeSys ] = gpsRead( filename )

id = fopen(filename);
str = fgetl(id);

str = fgetl(id);
CharType = [];
LatDouble = [];
LongitudDouble = [];
AltDouble = [];
AccurayFloat = [];
timeLong = [];
speed = [];
bearing = [];
timeSys = [];
while str ~= -1
    dots=find(str==':');
    CharType(end+1) = str(1:dots(1)-1);
    LatDouble(end+1) = str2num(str(dots(1)+1:dots(2)-1));
    LongitudDouble(end+1) = str2num(str(dots(2)+1:dots(3)-1));
    AltDouble(end+1) = str2num(str(dots(3)+1:dots(4)-1));
    AccurayFloat(end+1) = str2num(str(dots(4)+1:dots(5)-1));
    timeLong(end+1) = str2num(str(dots(5)+1:dots(6)-1));
    speed(end+1) = str2num(str(dots(6)+1:dots(7)-1));
    bearing(end+1) = str2num(str(dots(7)+1:dots(8)-1));
    timeSys(end+1) = str2num(str(dots(8)+1:end));
    str = fgetl(id);
end

fclose(id);

end
```

```
function [s,desvabs] = hpfilter(y,w,plotter)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
%   Author: Wilmer Henao    wi-henao@uniandes.edu.co
%   Department of Mathematics
%   Universidad de los Andes
%   Colombia
%
%   Hodrick-Prescott filter extracts the trend of a time series, the
output
%   is not a formula but a new filtered time series.  This trend can be
%   adjusted with parameter w; values for w lie usually in the interval
%   [100,20000], and it is up to you to use the one you like, As w
approaches infty,
%   H-P will approach a line.  If the series doesn't have a trend
p.e.White Noise,
%   doing H-P is meaningles
%
%   [s] = hpfilter(y,w)
%   w = Smoothing parameter (Economists advice: "Use w = 1600 for
quarterly data")
%   y = Original series
```

```
%   s = Filtered series
%   This program can work with several series at a time, as long as the
%   number of series you are working with doesn't exceed the number of
%   elements in the series + it uses sparse matrices which improves
speed
%   and performance in the longest series
%
%   [s] = hpfilter(y,w,'makeplot')
%   'makeplot' in the input, plots the graphics of the original series
%   against the filtered series, if more than one series is being
%   considered the program will plot all of them in different axes
%
%   [s,desvabs] = hpfilter(y,w)
%   Gives you a mesure of the standardized differences in absolute
values
%   between the original and the filtered series.  A big desvabs means
%   that the series implies a large relative volatility.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
tic
if nargin < 2
    error('Requires at least two arguments.');
end

[m,n] = size (y);
if m < n
    y = y';     m = n;
end
d = repmat([w -4*w ((6*w+1)/2)], m, 1);
d(1,2) = -2*w;      d(m-1,2) = -2*w;
d(1,3) = (1+w)/2;   d(m,3) = (1+w)/2;
d(2,3) = (5*w+1)/2; d(m-1,3) = (5*w+1)/2;
B = spdiags(d, -2:0, m, m);    %I use a sparse version of B, because
when m is large, B will have many zeros
B = B+B';
s = B\y;

if nargin == 3
    t = size(y,2);
    for i = 1:t
        figure(i)
        plot(s(:,i),'r');   grid on;   hold on;   plot(y(:,i));
title(['Series #',num2str(i)]);
    end
end
if nargout == 2
    desvabs = mean(abs(y-s)./s);
end
%toc
```

```
function [ dist ] = GPSdist( pos1, pos2 )

degreedist = pos1 - pos2;
degreedist = sqrt(degreedist(1,:).^2 + degreedist(2,:).^2);
R = 6370000;

dist = R * degreedist * pi / 180;

end
```