Indoor positioning for emergency first responders

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor in communication systems & engineering

Submitted by

Timon Zimmermann

July 2015 – January 2016

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Image and Visual Representation Laboratory

Supervisors	Professor
Adrien Birbaumer	Sabine Süsstrunk
Martijn Bosch	

Abstract

Standard positioning systems, like GPS, are not available inside buildings, which makes the indoor positioning a non-trivial question, especially in emergency cases, such as a fire environment. A solution for the indoor positioning of first responders in such an environment is proposed.

This report reviews of existing state of the art literature concerning ad-hoc indoor positioning systems and filtering algorithms for step counting and positioning. Multiple approaches are evaluated and put into perspective using firefighting as practical application.

A description of an implementation is available and has been put through a battery of tests. Some promising and applicable trails such as step frequency or collaborative positioning are also explained and detailed with a link to existing related scientific literature.

In the final analysis, the fundamental problem of such systems lies in the multiple sources of error and its exposure to numerous variables increasing the rate of drift accrual. Some techniques exist to minimize this rate but for the time being external information to the system is obligatory in order to recalibrate the system and completely cancel it.

Introduction

Indoor positioning is a non-trivial question due to the unavailability of standard systems like GPS in buildings. Emergency first responders, such as firefighters, currently use hand-drawn maps and security lines to ensure not getting lost inside a building. Both for the commander, outside the building, as for the men and women inside, a rudimentary positioning system would be helpful.

This project is done with the aim of obtaining a Bachelor degree in communication systems granted by the Federal Institute of Technology in Lausanne, Switzerland. The whole process is performed to analyze a positioning problem for “FireHUD”, a feasibility study within EPFL in the field of emergency environment. All this under the supervision of the Image and Visual Representation Laboratory.

At the beginning of the project, the relevant literature was thoroughly analyzed, to comprehend the subtleties and critical points of such a system. From there, the feasibility and precision of existing techniques were evaluated using easily available hardware, such as smartphones.

The project plan was then structured according to the deliverables, which were defined in compliance with the requirements of “FireHUD”, and discussed with its two main researchers and supervisors of this project. Consequently, **its final purpose of this work was to produce and document a naïve implementation of a positioning system, and then to test its limits**. Due to the nature of the problem, the algorithm could not assume any prior knowledge about the building conception, such as the presence of a Wi-Fi signal, in order to be as generalizable as possible for firefighting.

The workflow was chosen as follow:

Task	**September**				**October**				**November**				**December**			
Step detection		■	■													
Stride-length est.				■												
Heading est.					■											
API discussion						■										
1st tests						■	■									
Test protocol def.								■								
Real-time adapt.									■							
Analyze results										■						
Improvements										■	■	■				
Final test session													■	■		

Bear in mind that the final application will be part of a client/server project developed by a Master student, thus requiring some discussions regarding the real-time results for the API.

Prior to the start of the implementation, to complete the work plan described above, roughly thirty hours have been spent reading papers about pedestrian dead-reckoning systems and existing trails. As a result, I had a good overview on the subject and have been able to estimate what was achievable within the timeframe set.

The research results on the state of the art as well as the said implementation can be found and are summed up in the corresponding sections of the report. Once completed and documented, the algorithm has been put to the test in different situations through a test protocol. Afterwards the results of this implementation have been tested and analyzed to explore accordingly possible improvements.

State of the art

Pedestrian dead-reckoning systems have always been an engineering challenge, whether it be the portability of such a system or the precision of the indoor tracking for the purpose of being usable in practice. The former being less and less an issue as a result of the continual miniaturization of sensors, improving portability, **precision being the crucial aspect of the task**.

Indoor positioning as an open problem, has become even more active with the widespread deployment of modern smartphones which contain necessary sensors. The challenge is now to profit from the data of theses sensors to accomplish the same level of robustness that is being attained by systems like GNSS (Global Navigation Satellite Systems), including the most known: the GPS.

Less hardware technicalities coupled with broad applications which would benefit from a precise location system, such as sports to track players in real-time by recording all their movements, tourism, hospitals to know where patients are, supermarket to optimize shelf turn-around by providing accurate customer analytics, parking lots, or for safety applications like in our project increasing the awareness around the topic. Therefore, indoor positioning systems have gained in popularity and there exists a rich scientific literature on the subject.

With the use of mainly two different sensors, the accelerometer and gyroscope, Step-and-Heading Systems (often referred as SHSs) offer a real-time estimated position of the person by computing vectors, defined by their magnitude and direction, being the distance and heading, and accumulating them end to end.

For this reason, we can split SHSs algorithm in 3 phases:

- Step detection
- Stride-length estimation
- Heading estimation

Sensors location

But before diving into existing techniques for each phase, a key point about the sensors is their position on the body. Indeed, four major positions are usually used: in the pocket [1], on the foot [2], to the torso [2], on the head [3], to the lower-back [4].

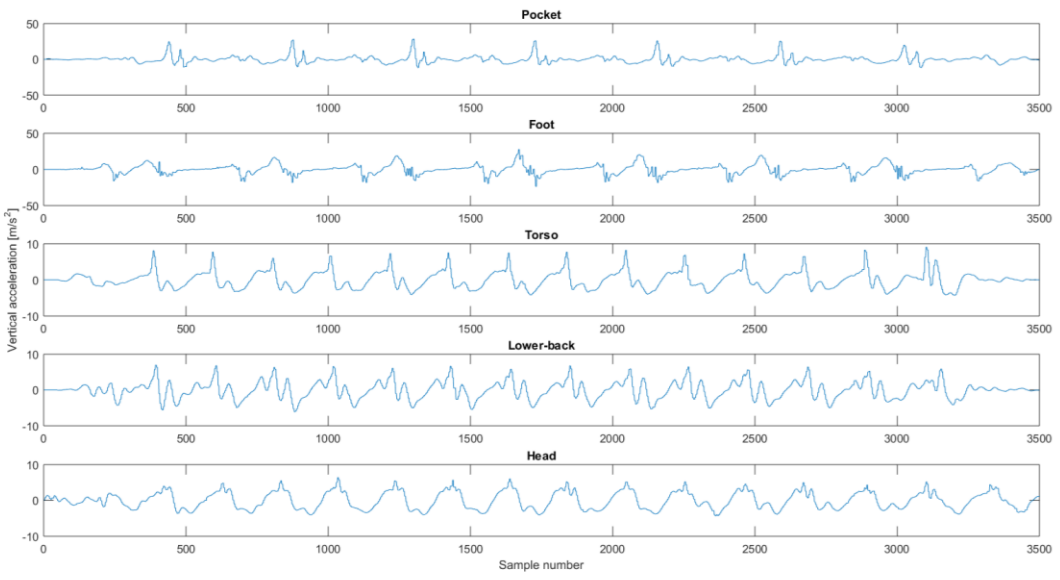


Figure I: Vertical acceleration for different body locations

The first two add the same complexity to the step detection and stride-length estimation since the acceleration will be asymmetric. Explicitly, steps done with the leg where the phone is worn will have a different acceleration than the ones with the leg without sensors (figure I). On the other hand, it also gives more information in the sense that one might use the differentiation between these steps to better interpret the signal and the subject gait cycle [2].

In addition to this distinction that has to be made, putting the smartphone in the pocket also increase the tilting of the hardware, meaning more noise and irrelevant acceleration peaks. This issue is also present when mounted on the foot as the emplacement is exposed to some movement caused by random motion of the leg. A solution to this problem is to use the magnitude of the acceleration instead of solely the vertical component [5].

The other three positions are quite equivalent in term of acceleration magnitude and periodicity, they offer steady signals making them suitable locations for sensors placement and easier to handle for a naïve approach. However, the head position suffers from arbitrary gaze orientation of the subject which will interfere with the heading estimation.

Data filtering

In the first place, acceleration must be filtered to smoothen the signal and enhance algorithm performances. Indeed, there is a lot of noise in data and therefore applying a low-pass ensures better conditions to the proper functioning of future algorithms, whether it be step detection or stride-length estimations. To do so, two techniques prevail: a Butterworth low-pass filter [6] or a moving average filter [5]. The second being easier to implement compared to the former one, Butterworth offers an almost perfectly periodic signal (figure II).

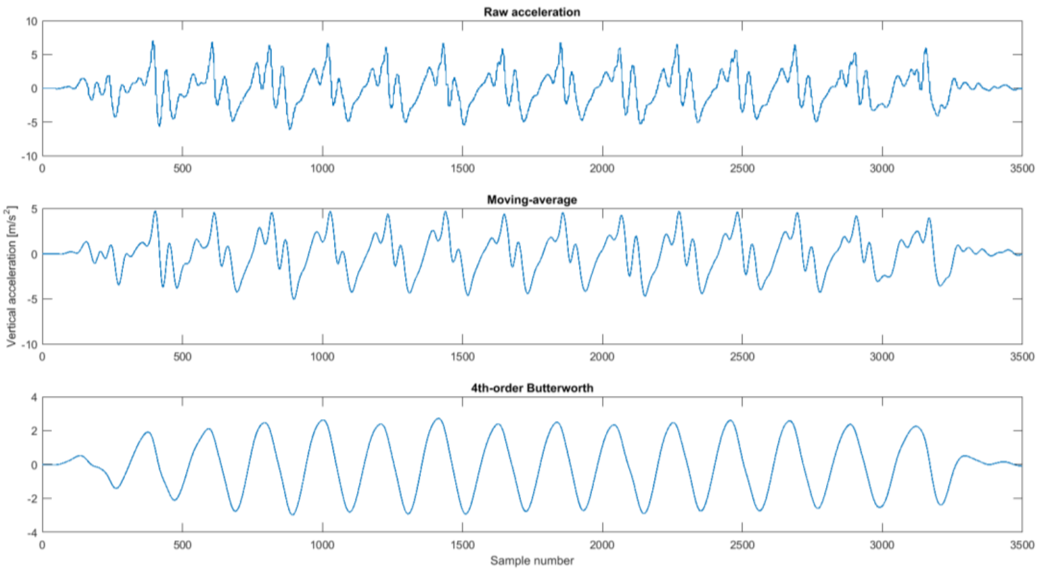


Figure II : Filters comparison on raw accelerometer data (sensors located to the lower-back)

Step detection

The step detection, or step cycle detection, is usually achieved by observing patterns in the sensor data induced by repetitive events inherent to the human gait. Usually these events are "Heel strike", "Foot flat" or the "Toe off", as seen in figure III.

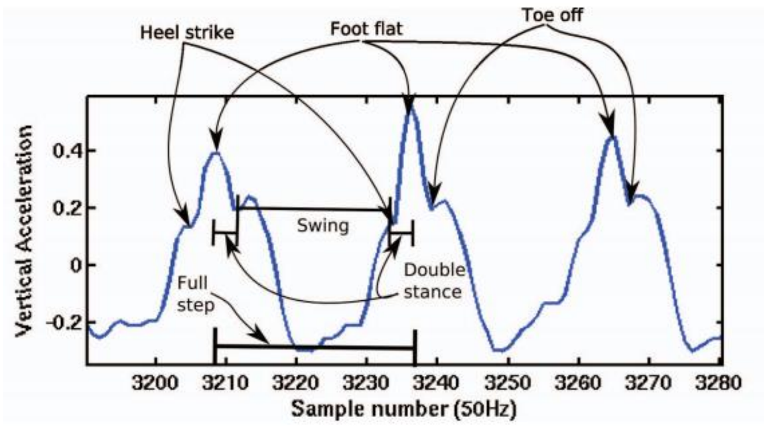


Figure III : Human gait main events identified on the vertical acceleration [4]

Given these points, techniques have been developed to segment and identify steps in the signal:

Zero crossing [7], [8]

This technique simply seeks positive zero-crossing in the acceleration value to determine the occurrence of step and its boundaries. The downside comes when trying to reject false step detection, which is usually performed with a time-interval threshold [9]. Indeed, finding an appropriate threshold varies a lot for some subjects, making prior calibrations almost imperative.

Peak detection [10], [11]

This method targets the "foot flat" occurrences to detect steps, they are typically defined by peaks in the vertical acceleration. Although any peak detection algorithm will then be able to highlight possible strikes, it's important to note that bounces of the sensor can engender multiple local peaks due to the resulting forces on foot impact. Hence, it might increase the algorithm complexity and add difficulties to properly detect steps.

Stride-length estimation

Estimating a subject stride-length for each step is a more ambitious task, a positioning system will not undergo a major change if the step detection is erroneous by 2 or 3 steps. However if each step is miscalculated by even a thin percent, the path in the long-term has the risk to be completely inaccurate and end up being useless.

The most straightforward approach considers the step length as constant. As a matter of fact, this method gives surprisingly good results for subjects adopting a natural constant walking pace but gets worst when applied to real-life simulations, such as changing walking pace (running, fast walking…). In particular, Weinberg reported ±50% fluctuation between a similar individual stride-length for different walking speed and up to ±40% variation from person to person at given walking speed [12].

For this reason, a dynamic estimate would be preferable. Consequently, Weinberg [12] compared the human leg with a lever of fixed length when the foot is on the ground. This mechanism leads to an estimate of the distance using both the maximum and minimum acceleration measured in a single stride:

$$Step\ length \cong k * \sqrt[4]{a_{max} - a_{min}}$$

The constant "k" is determined empirically for each subject

Then again, this formula presents issues when subjects are changing pace, or for the inconvenience of calibrating the constant for every given subject. To solve this problem, Scarlet came up with another formula showing a correlation between the maximum, minimum and average acceleration of a single stride [13]:

$$Step\ length \cong k * \frac{\frac{\sum_{i=1}^{N} a_i}{N} - a_{min}}{a_{max} - a_{min}}$$

The constant "k" is a multiplier, and N is the number of samples within the stride

Heading estimation

For the purpose of the implementation, magnetometer cannot be exclusively taken into account to deduce a potential heading of the subject. Indeed, materials have significantly different effects on magnetometers, with heading errors of up to 100° [14].

Instead, the gyroscope coupled to the magnetometer using a Kalman filter is widely used to obtain a meaningful orientation [10]. The Kalman filter estimates the error of a sensor value with external aiding or updates from other sources, minimizing the error of predicted future states given previous ones (good overview of such filters : [15]). This technique is known as “Sensor fusion” and might even be designed with accelerometer data on top of the two other sensors to get an even better estimation of the subject heading [16].

Putting things together

Overall, it seems like the heading and stride-length estimations are the critical parts of such systems, both leading to huge drifts in the positioning on the term when improperly predicted.

Nevertheless, estimating the step size seems to give quite accurate results compared to the heading where a lot of different approaches and complementary techniques have to be used in order to derive a good approximation of the subject orientation. The most compelling evidence is the abundant literature and scientific research specifically on the matter, backing the fact that this is the most arduous part of such a system.

For instance, techniques such as cooperative positioning, where linkage structures between multiples clients are created and are working as constraints to reduce variables [17]. Likewise, particle filtering and map matching algorithm are notorious among every other techniques for giving accurate results [18]. The downside being that the map has to be preprocessed with the intention of placing walls and other building structure information for the algorithm, which is in contradiction with the needs of FireHUD.

Implementation

The naïve algorithm is fully based on prior research and existing techniques described in the previous section. It has been developed on a HTC One M9 with Android 5.1 (but the class is compatible with former version of Android, minimum Android 2.3). Every graphics and data plot in this section are done with this implementation, with the smartphone worn to the lower-back of the subject.

The implementation works in real time so information can be retrieved at any moment by another external thread, it means that every point of the algorithm processes only on most recent data from the capture. To ensure that every new data is handled properly, a thread is spawned within a pool every three seconds. This same thread will then create a sublist of the whole dataset to exclusively pull out information from the end of the last step to the end of the newly acquired data.

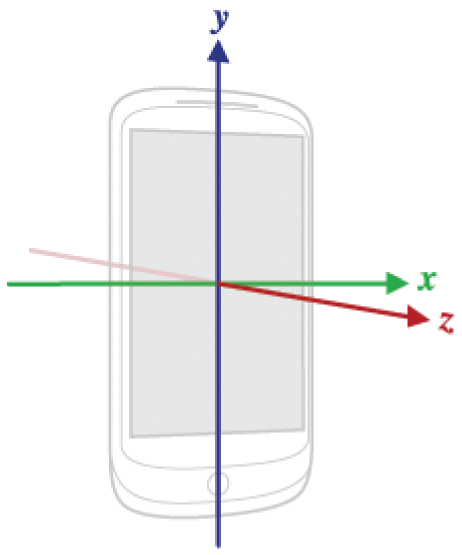


Figure IV : Android device relative axis [19]

It ensures that no information will be lost in the process since steps are forming a perfect partition of the signal. To put it differently, if we concatenate all the strides we would retrieve a perfectly continuous signal.

Data acquisition [19]

The data acquisition is achieved with an Android device and therefore the sampling rate is defined by the minimum delay between two events. As an illustration, if the delay is set to 10ms and a first motion event triggers the data capture, there will not be any other capture provoked by another movement within 10ms. In this algorithm, the sensor delay is set as low as possible (0 microsecond delay), since prior tests with slower capture rate yielded to poor results in the heading system.

The first sensor registered is the accelerometer, which includes the gravity by default. It is used for the step detection, stride-length estimation and heading estimation. However, the earth acceleration is irrelevant in this application and hence the system measures linear accelerations, or in other words, a high-pass filtered version of the accelerometer data.

Thus, the linear acceleration obeys the following relation:

$$acceleration = gravity + linear\_acceleration$$

Additionally, the raw gyroscope and magnetometer values are captured and stored even if these values are not directly used in the implementation, in favor of the android rotation vector. They are saved to explore possible improvements and determine if they could bring additional information, enhancing the results.

Finally, as previously mentioned, the rotation vector is used for the heading estimate and represents the orientation of the device as a combination of an angle and an axis. It is triggered whenever there is a rotation of the device, it is computed and adjusted with an extended Kalman filter using the accelerometer, gyroscope and magnetometer. On top of that, each capture is associated to the system time in milliseconds in order to design appropriate filters for steps.

Data filtering

Accelerometer data are being filtered with a simple moving average window (set to 15 in our tests) before any computation or algorithm logic. The formula of the filter is applied in real-time during capture, creating a small lag of half of the window's size:

$$y[i] = \frac{1}{M} \sum_{j=-(M-1)/2}^{(M-1)/2} x[i+j]$$

X is the input signal and M the window size. Producing y, the output filtered signal.

The result of such a filtering technique can be seen in the previous section, in figure II. It is sufficient for the naïve implementation but implementing a Butterworth filter might decrease algorithm complexity.

Indeed, the algorithm has to check that the second positive zero crossing is not too close to the negative one, to prevent false step boundaries due to small bounces near zero acceleration. This additional filter would not be needed if steps were detected on a Butterworth filtered signal, as seen on the figure II.

This technique is further explained in the next section.

Step detection

The step detection algorithm uses a fusion of the zero crossing techniques to segment the signal and select steps, in order to filters valid ones with peak detection approach.

Each step is determined and bounded by two consecutive positive zero-crossing, it includes a negative zero crossing within this interval as well as a maximum and minimum acceleration. Once the signal is divided into steps by searching positive zero crossing into the data, we filter steps to discard wrong ones, either caused by noise, bounces or anything else that does not fit the desired constraints.

To do so, the algorithm put each step through multiple tests: An absolute minimum value for the maxima and minima, to remove steps only due to noise (set to 0.5 m/s^2 in our tests). A relative minimum value for the maxima (maximum value for the minima), determined by the last valid minima plus a given constant (set to 1.5 m/s^2 in our tests) and finally an absolute minimum time elapsed between the negative zero crossing and the second the positive zero crossing (set to 350 msec in our tests). The latest prevents bounces around zero to falsely set the second bound, as you can see on figure V.

Every time a step is rejected because it does not fulfill necessary conditions listed above, the step is either removed or, in the case of the last test, modified accordingly to set the new correct boundaries.

Additionally, the actual step frequency is computed for each individual stride to study the influence of this frequency on the stride-length. It is executed with respect to the last three steps and the time elapsed between these steps:

$$Step_{frequency}[i] = \frac{3}{Step_{Capture\ time}[i] - Step_{Capture\ time}[i-2]}$$

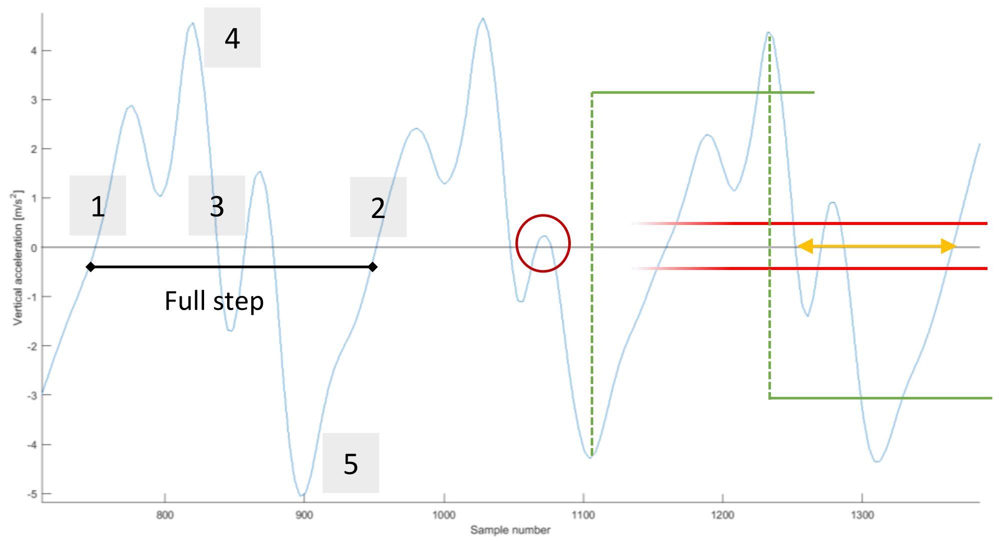


Figure V : Step description (numbers), filter details (lines) and bounce issue (red circle)

1. *First positive zero-crossing (start of step)*
2. *Second zero-crossing (end of step)*
3. *Negative zero-crossing in between*
4. *Maximum acceleration*
5. *Minimum acceleration*

Red line:	*Absolute minimum value for each peak, to immediately remove steps due to noise.*
Green line:	*Relative threshold for acceleration peaks, determined by the last valid maxima or minima ± a given constant.*
Yellow line:	*The absolute minimum time between the negative zero crossing and the second positive one (end of step)*

Stride-length estimation

The approaches used in this implementation are both the Weinberg and Scarlet methods. Indeed, since it might be interesting to gather and compare results of both methods, the algorithm computes the two and stores them in the characteristics of each step.

Both methods needing the peaks, the maximum and minimum acceleration are found with a standard max value algorithm. Firstly, between “1” and “3” for the maximum and then between “3” and “2” for the minimum. The average of the step for the Scarlet approach is computed with bounds “1” and “2” (figure V).

From there, each step is fully characterized by eight components, giving us the following data structure:

1	The sample number of the first positive zero crossing (start of step)
2	The sample number of the second positive zero crossing (end of step)
3	The sample number of the negative zero crossing within the step
4	The sample number of the maximum peak
5	The sample number of the minimum peak
6	The step length using Weinberg approach
7	The step length using Scarlet approach
8	The frequency of the last three steps

Heading

The heading estimation is operated by the “Android Sensor Fusion” [19] technology, which provides the rotation vector each time a device orientation change is detected.

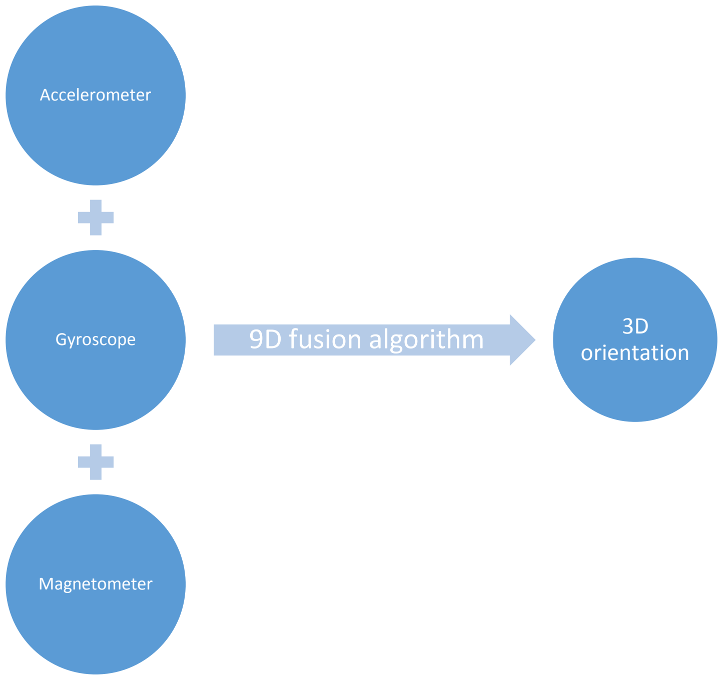


Figure VI : 9D fusion algorithm to estimate 3D orientation of the device from different sensors data

This vector is computed by a fine-tuned extended Kalman filter and the elements of the resulting rotation vector are the components of the unit quaternion. From this vector, the application creates a rotation matrix associated to the vector and get the orientation of the device from this rotation matrix, previously remapped to another coordinates system to match device X and Z axis.

```
float[] c_mRotationMatrixAdjusted = new float[16];
float[] c_rotationVector = new float[3];

SensorManager.getRotationMatrixFromVector(c_mRotationMatrix, event.values);
SensorManager.remapCoordinateSystem(c_mRotationMatrix, SensorManager.AXIS_X, SensorManager.AXIS_Z,
c_mRotationMatrixAdjusted);
SensorManager.getOrientation(c_mRotationMatrixAdjusted, c_rotationVector);
```

Where “event.values” is the variable containing the rotation vector returned by “Android Sensor Fusion”. Then “c_rotationVector” will contain the resulting orientation of the device, with each components representing the rotation around an axis in radians.

API

The real-time aspect of the implementation, with threads computing and populating arrays containing vectors, makes the API pretty straightforward.

Any external software can use the library, retrieve the steps array at any moment as well as the orientations array and the initial location, accumulate them end to end and build a path in real-time.

Experimental setup

For the experimentation of the algorithm, a battery of tests have been designed in order to offer the possibility of analyzing results of the different parts of the application. On top of that, such testing framework allows a better comparison of the future improvements impact.

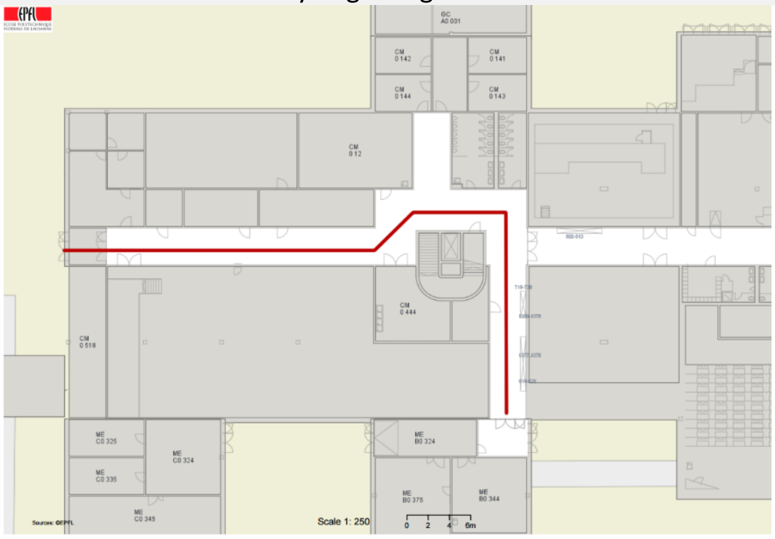
The battery of tests was performed with an HTC One M9, tightly worn to the lower-back of the subjects with a belt. All of these scenarios are located at the Swiss Federal Institute of Technology and include 4 tests related to a normal path and 2 additional tests for future stairways detection. Additional tests and experiences were performed along the project process and can be found in the appendices and the files or logs.

The subjects were chosen as follow:

	SUBJECT 1	SUBJECT 2	SUBJECT 3	SUBJECT 4	SUBJECT 5
SIZE	1m79	1m75	1m83	1m87	1m70

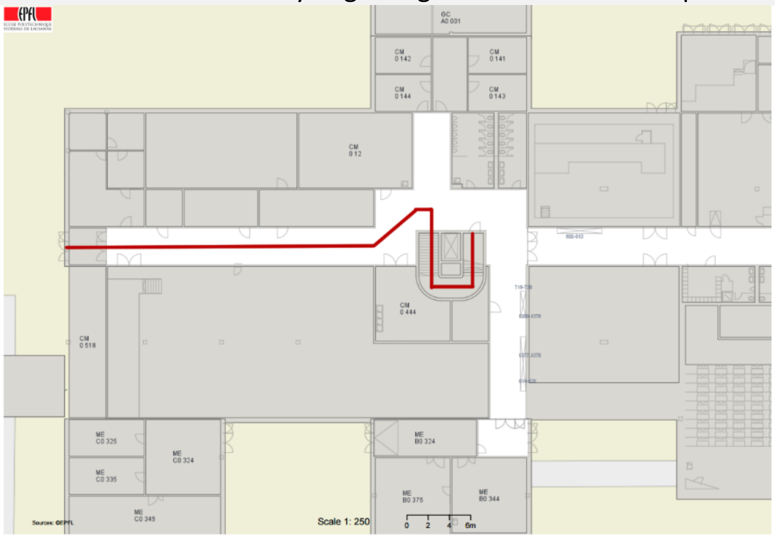
Test 1

Simple path

LENGTH	61m
STARTING POINT (GPS)	6.566278° E / 46.520495° N
DETAILS	Indoor path in a basement (magnetometer fully unavailable), include 1 door at the very beginning
PATH	

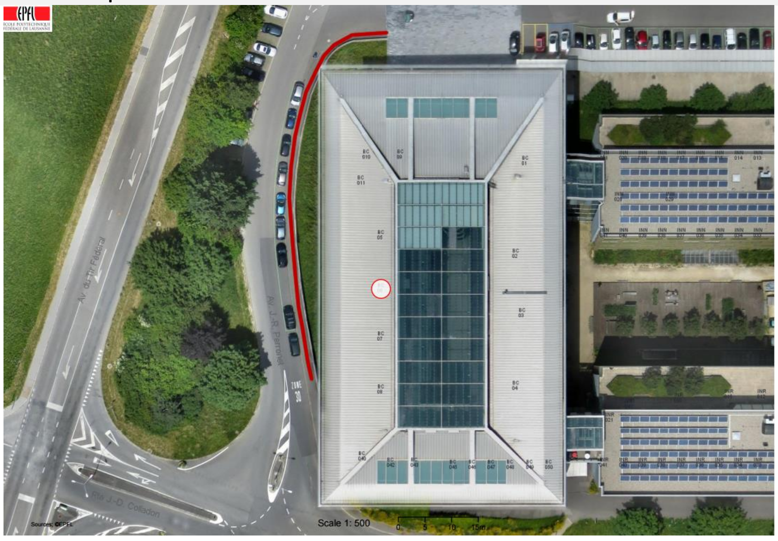
Test 2

Semi-complex path

LENGTH	52m
STARTING POINT (GPS)	6.566278° E / 46.520495° N
DETAILS	Indoor path in a basement (magnetometer fully unavailable), include 1 door at the very beginning and 2 stairs of 10 steps each
PATH	

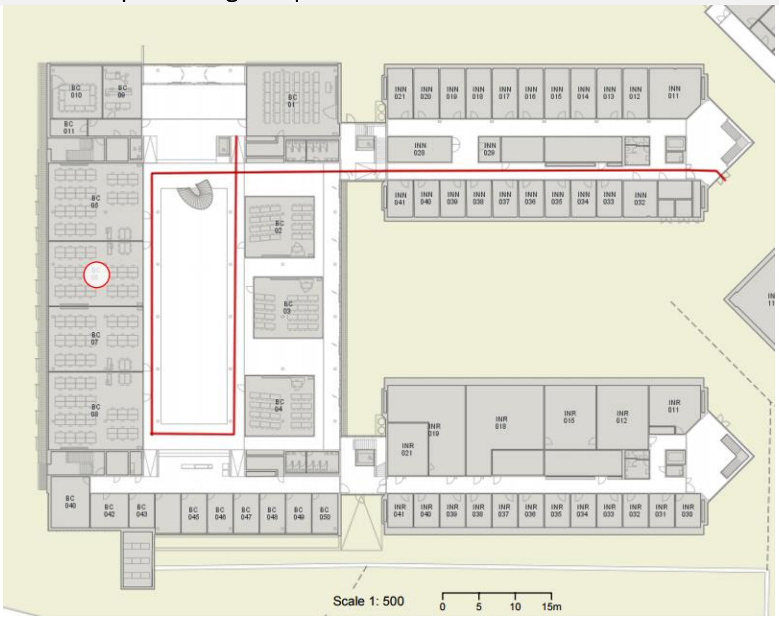
Test 3

Simple path

LENGTH	75m
STARTING POINT (GPS)	6.56176° E / 46.518993° N
DETAILS	Outdoor path which consist of only a big ramp to better study issues reported in the state of the art
PATH	

Test 4

Complex path

LENGTH	168m
STARTING POINT (GPS)	6.562883° E / 46.518688° N
DETAILS	Fully indoor, with 4 doors (including 2 automatic ones) and a small ramp 2m long ramp at the end.
PATH	

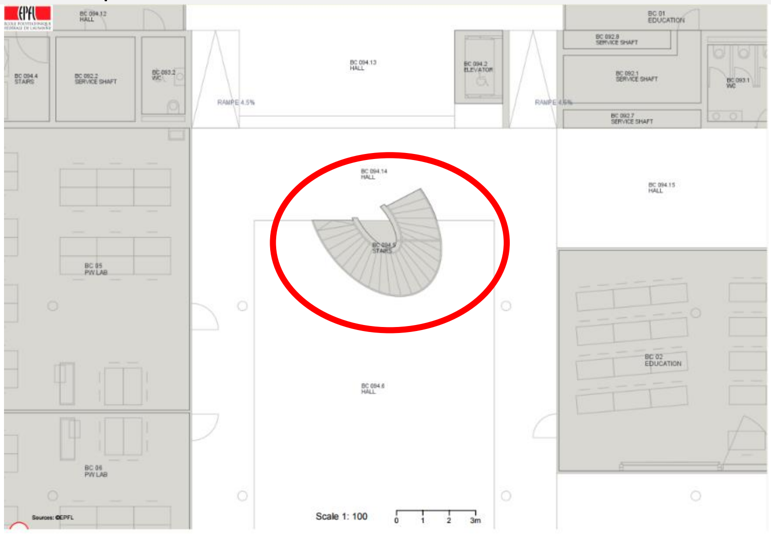
Additional test 5

Simple staircase

NUMBER OF STEPS	17
STARTING POINT (GPS)	6.562224° E / 46.51818° N
DETAILS	Outdoor staircase
PATH	

Additional test 6

Simple staircase

NUMBER OF STEPS	23
STARTING POINT (GPS)	6.561943° E / 46.518681° N
DETAILS	Indoor spiral staircase
PATH	 BC 094.12 HALL BC 094.4 STAIRS BC 092.2 SERVICE SHAFT BC 093.2 WC RAMPE 4.5% BC 094.13 HALL BC 094.2 ELEVATOR RAMPE 4.5% BC 01 EDUCATION BC 092.8 SERVICE SHAFT BC 092.1 SERVICE SHAFT BC 093.1 WC BC 092.7 SERVICE SHAFT BC 094.14 HALL BC 094.15 HALL BC 094.5 STAIRS BC 05 PW LAB BC 02 EDUCATION BC 094.6 HALL BC 06 PW LAB Sources: ©EPFL Scale 1: 100 0 1 2 3m

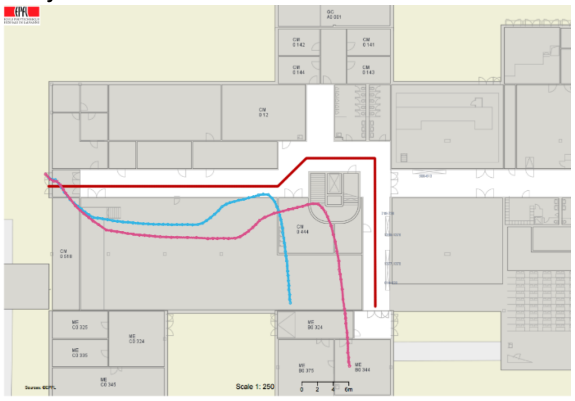
Results & discussion

All blue estimated trajectories drawn hereafter refer to the Weinberg approach and pink ones to the Scarlet technique.

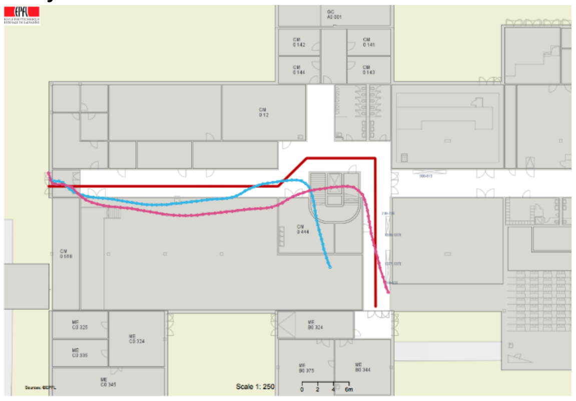
Results

TEST 1	SUBJECT 1	SUBJECT 2	SUBJECT 3	SUBJECT 4	SUBJECT 5
WEIBERG EST.	56m	54m	65m	67m	71m
SCARLET EST.	73m	65m	65m	66m	69m
REAL VALUE	61m	61m	61m	61m	61m
WEIN. DELTA	-5m	-7m	+4m	+6m	+10m
SCAR. DELTA	+12m	+4m	+4m	+5m	+8m
WEIN. ERROR	***8.2%***	***11.4%***	***6.5%***	***9.8%***	***16.4%***
SCAR. ERROR	***19.6%***	***6.5%***	***6.5%***	***8.2%***	***13.1%***

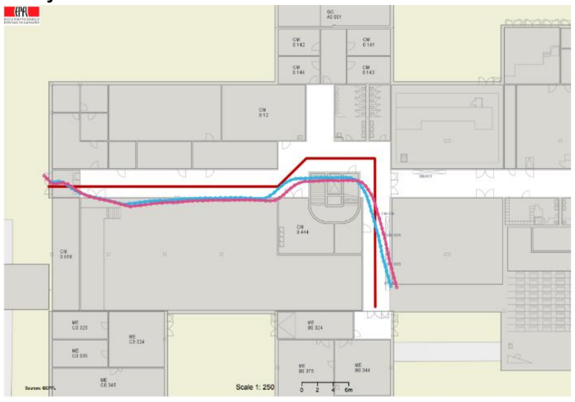
Subject 1 – Test 1



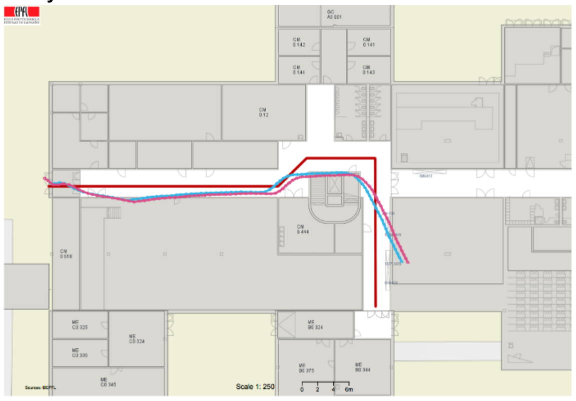
Subject 2 – Test 1



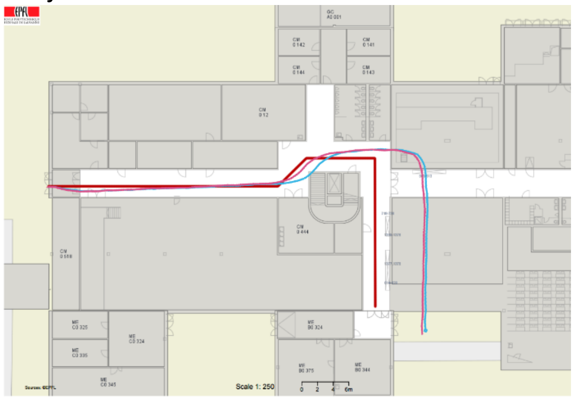
Subject 3 – Test 1



Subject 4 – Test 1

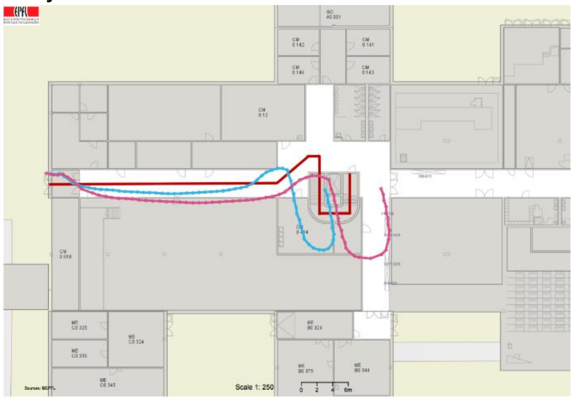


Subject 5 – Test 1

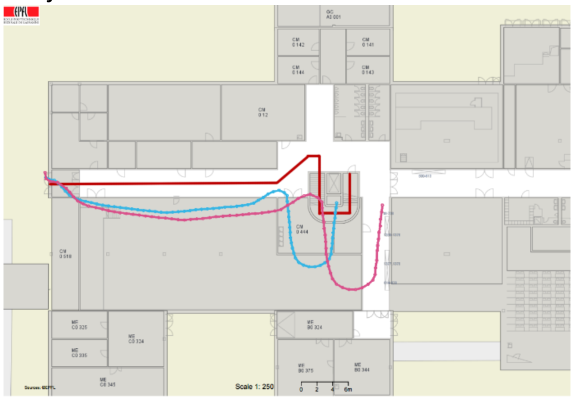


TEST 2	SUBJECT 1	SUBJECT 2	SUBJECT 3	SUBJECT 4	SUBJECT 5
WEIBERG EST.	54m	56m	65m	69m	64m
SCARLET EST.	62m	66m	70m	66m	71m
REAL VALUE	52m	52m	52m	52m	52m
WEIN. DELTA	+2m	+4m	+13m	+17m	+10m
SCAR. DELTA	+8m	+14m	+18m	+16m	+8m
WEIN. ERROR	***3.8%***	***7.7%***	***25%***	***32.7%***	***19.2%***
SCAR. ERROR	***15.4%***	***26.9%***	***34.6%***	***30.8%***	***15.4%***

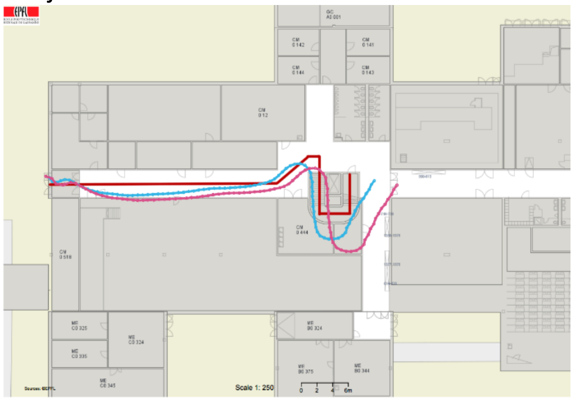
Subject 1 – Test 2



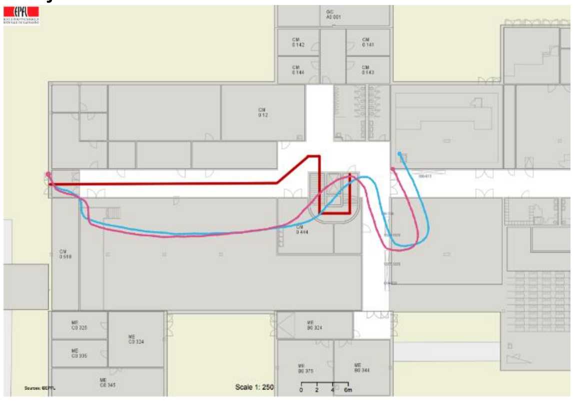
Subject 2 – Test 2



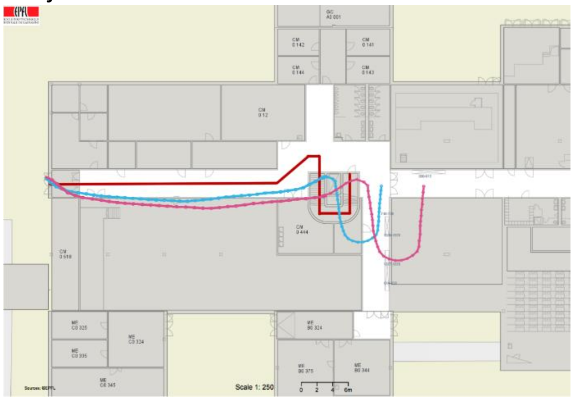
Subject 3 – Test 2



Subject 4 – Test 2

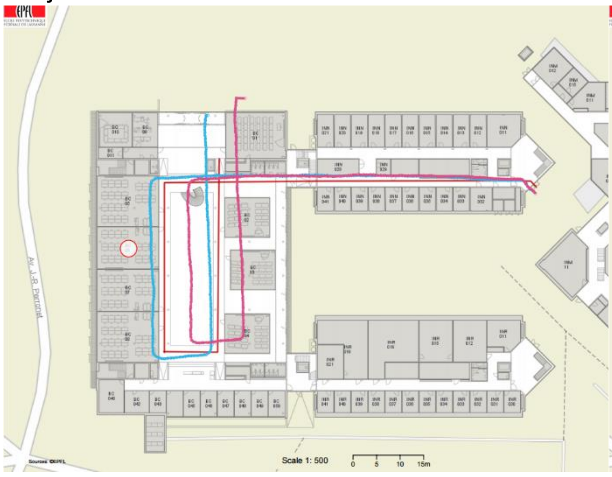


Subject 5 – Test 2

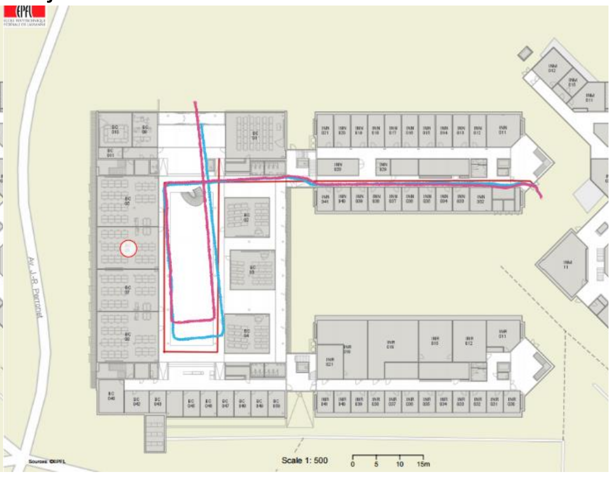


TEST 4	SUBJECT 1	SUBJECT 2	SUBJECT 3
WEIBERG EST.	180m	173m	169m
SCARLET EST.	172m	171m	160m
REAL VALUE	168m	168m	168m
WEIN. DELTA	+12m	+5m	+1m
SCAR. DELTA	+4m	+34m	-8m
WEIN. ERROR	***7.1%***	***3.9%***	***0.5%***
SCAR. ERROR	***3.1%***	***20.2%***	***6.2%***

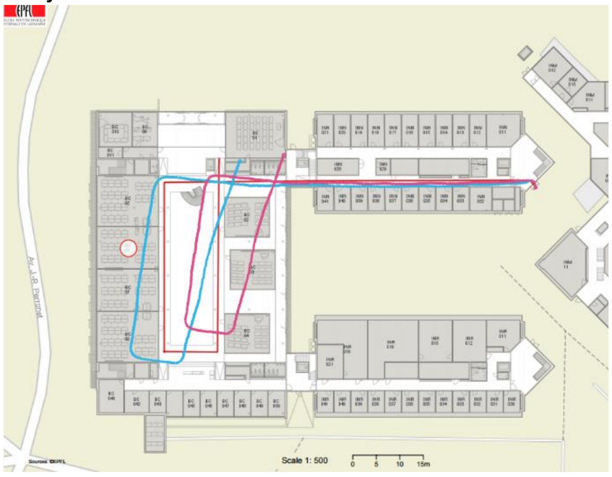
Subject 1 – Test 4



Subject 2 – Test 4



Subject 3 – Test 4



Discussion

The step detection method of the implementation seems to be quite accurate, with no more than 2 or 3 steps falsely counted. These incorrectly detected steps are mainly present at the beginning and at the end of the signal, thus might be due too strict or too loose filters parameters. Indeed, the parameters are set empirically and are constant among subjects, which is suboptimal considering the fluctuations reported from person to person [12].

The robustness of the step detection algorithm could be improved for very small strides, this occurs for instance when opening a door because subjects move the body up and down as if they were walking normally when in fact it is just very tiny steps. This problem arise at the beginning of each path in the battery test, creating huge drifts at for this reason.

To address this issue, the usage of the Z acceleration could be a good option. The pattern of the signal during the "weird" small moves is aperiodic and inconsistent where during normal gait, the Z acceleration represent a periodic and steady signal (Figure VII).

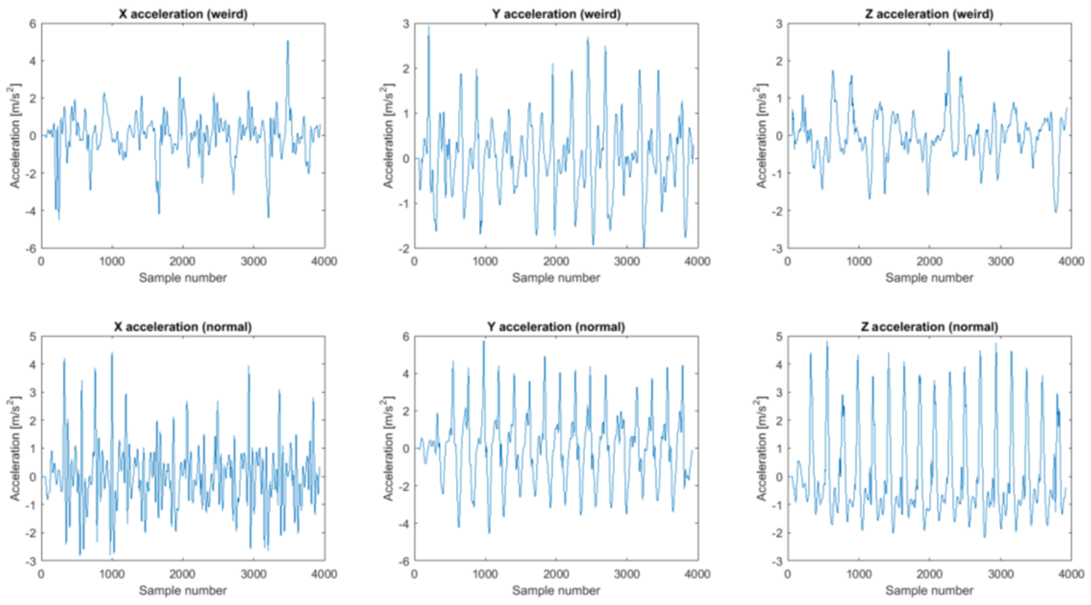


Figure VII : Comparison of the acceleration on all axis in the case of small "weird" steps and for a normal walk pattern

The stride-length estimation is worse than the step-detection, as expected with constants staying the same among all individuals, hence resulting to bad results as predicted during literature research. The results are quite accurate and lead to a good positioning approximation for a single subject but are not regular when it comes to multiple persons of different size. An important thing to notice is the consistency of the estimated path length for the Scarlet approach. The purpose of this method was to stay as consistent as possible among multiple subjects of different size and gait pattern.

Moreover, stride-length during test 2 is evaluated way too long due to the presence of stairways. The horizontal displacement being much smaller going upstairs or downstairs than when walking on a flat surface. Existing trails to this problem are discussed in the "Future work" section.

As can be seen, the heading is the weak part of the implementation, during test 1 and 2 it is clear that errors at the beginning of the path lead to huge discrepancies between the real path and the estimated one. The accumulating error in the heading creates huge drifts in the predicted position. Unfortunately with the use of the sensor fusion integrated into Android, the implementation lacks of possible tweaks in the heading system.

In like manner, when we analyze the heading during the drift, it is noticeable that in some cases the orientation provided by the rotation vector generates more drift compared to an alternative simplistic estimation using the integrated gyroscope data.

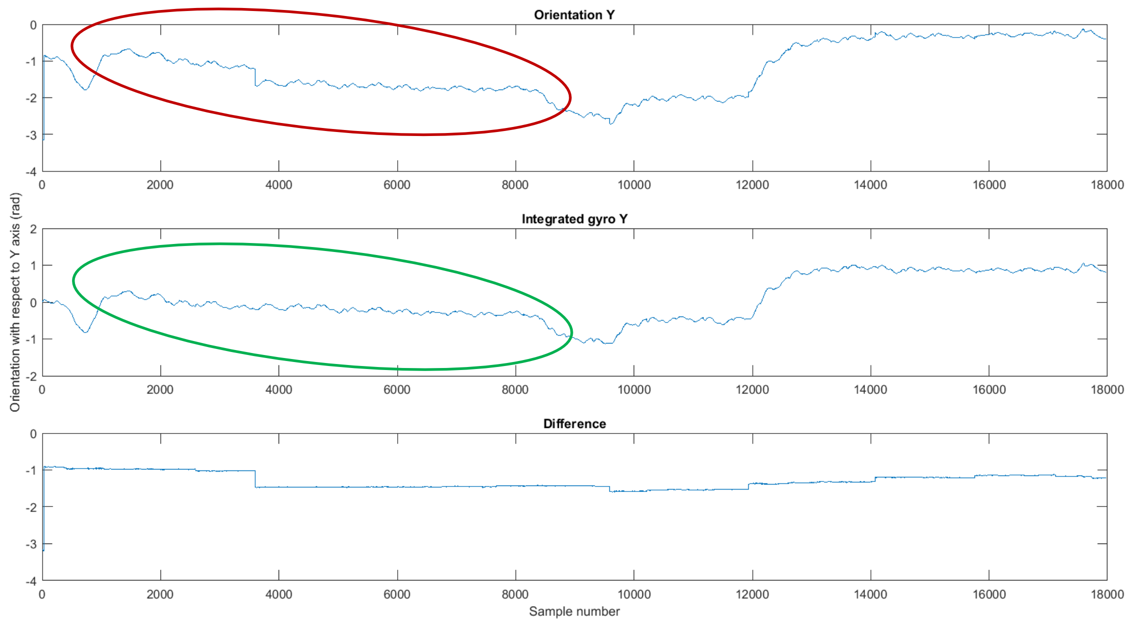


Figure VIII : Example where the drift present in the path of this subject would have been smaller without Android Sensor Fusion (red circle) compared to the integrated gyroscope data (green circle)

If we compare test 1 & 2 to the test 4, the results are surprisingly good when one would guess that the error should increase due to the length of the path being much longer. These unexpectedly low error might be due to the location where the tests occurred. A lot of interference and potential variables are to take into account before drawing conclusions, magnetic disturbance with the Sensor Fusion trying to use magnetometer, wall materials, and so on.

Another key point is the hardware limitation, the sensors provided by modern smartphones are sufficient for gaming purposes, to detect screen rotation, or to trigger an event when the phone is moving but these sensors might be lacking in precision for real applications requiring less noise and drift. It would be interesting to test the implementation on another hardware such as a Raspberry Pi (tiny and affordable computer) or any other device allowing Android software.

Future work

Frequency analysis

Basing stride-length estimation on frequencies of steps is a promising trail within medical community. Research have shown a tight relation between step frequency and walking speed, permitting to derive step-length.

Even though the relationship between the said walking speed and stride-length is non-trivial, a linear relationship is sufficient to achieve 5.6% root mean squared error, as reported [21].

Cooperative positioning

A relevant improvements to the naïve positioning system relies on the fact that firefighters are mostly in group, which makes this approach relevant to our application.

This techniques uses communication to enhance the precision, every client is aware of the positions of others to create a linkage structures working as a constraints to the displacement. Altogether, such a method can be useful to reduce the cumulative error of a standard pedestrian dead reckoning system. Another good point to this approach, the absence of additional infrastructure required. Indeed, all smartphones are equipped with Wi-Fi hardware.

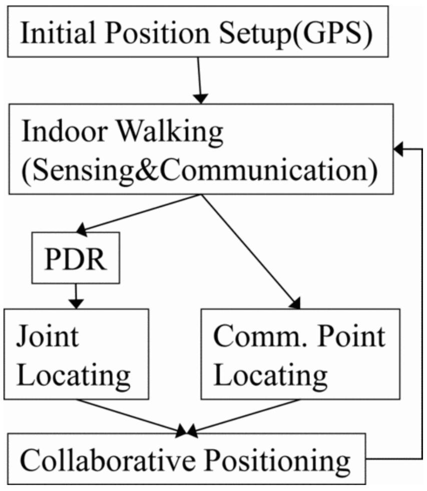

Initial Position Setup(GPS)
Indoor Walking (Sensing&Communication)
PDR
Joint Locating
Comm. Point Locating
Collaborative Positioning

Figure IX : Process flow for the collaborative positioning system [17]

3D positioning

Most of the assumptions made by many pedestrian dead reckoning breaks on slopes of 10% or more, as reported [22].

The usage of a barometer is not a good option due to the scenarios that the application will face, extreme temperature fluctuations and pressure variations are common inside burning buildings.

Staircase and slope detection

The problem occurs when walking up or down stairs or small ramp (figure X), a categorization of steps (slope step, stair step) would facilitate the projection of the step-length both horizontally and vertically.

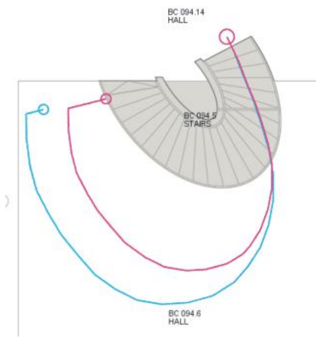


Figure X : Typical limitation of the algorithm, stride-length is being estimated too long in staircase, leading to considerable errors. (The bend at the very end of the path is due to a tilting of the smartphone)

Recently, studies reported accuracies exceeding 95% for the overall classification of five different human gait patterns: flat surface, walking up and down ramps, going up and down stairways [23]. However data are being preprocessed and the features of the walking pattern extracted, forming an empirical method. This technique has been extended to work autonomously [24], expending possibilities to the automatic distinction of different gait patterns.

All in all, contributing to a dynamic adaptation of the stride-length both horizontally and vertically, leading to a potential positioning in three dimensions.

Conclusion

Pedestrian dead reckoning system is a tough engineering task, both to fit the practical demand such as battery power requirements, processing power requirements or wearability, and to take up the challenge to be reliable in the long term.

While the techniques previously detailed are intended to minimize the rate of drift accrual, it is unrealistic to expect the total prevention of the problem. The heading is the limiting factor to a long-term indoor tracking system, generally researchers are incorporating additional information of the environment or external measurements to cope with the issue.

Existing trails to address this issue are typically “map matching” and “particle filtering”, but they require a high computational effort. For this reason, a smartphone or light hardware has not the computational power and these techniques are not suited for a real-time application with the battery power available on such devices. Promising trails exist with simplified algorithm which can run in real-time on a smartphone, leaving enough remaining processing power for an operating system [18]. Roughly explained, this technique generates many “particles” representing a possible 2D position and heading of the subject. A particle filter will then remove the ones going through walls in the floor map, increasing positioning accuracy.

The two other parts of the algorithm, the step detection and stride-length estimation stays much harder when it comes to be consistent among multiple persons, leading to overly optimistic results due to single subject testing. However with the techniques quoted in the previous section, it is clear that a sufficient accuracy is definitely reachable.

Bibliography

[1]	K. K. C. V. N. P. R. S. Anshul Rai, "Zee: Zero-Effort Crowdsourcing for Indoor Localization," 08 2012. [Online]. Available: http://msr-waypoint.com/pubs/166309/com273-chintalapudi.pdf.
[2]	J. C. K. F. G. L. Ross Stirling, "An Innovative Shoe-Mounted Pedestrian Navigation System," 04 2003. [Online]. Available: http://plan.geomatics.ucalgary.ca/papers/ross_stirling_shoemountedfinal.pdf.
[3]	S. Beauregard, "A Helmet-Mounted Pedestrian Dead Reckoning System," 03 2006. [Online]. Available: https://www.xsens.com/images/stories/PDF/A%20Helmet-Mounted%20Pedestrian%20Dead-Reckoning%20System.pdf.
[4]	R. C. G. A. L. a. J. C. A. Diego Alvarez, "Comparison of Step Length Estimators from Weareable Accelerometer Devices," 08 2006. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4463166.
[5]	W. a. H. R. Pratama A.R., "Smartphone-based Pedestrian Dead Reckoning as an indoor positioning system," 09 2012. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6339316.
[6]	M. L. X. S. a. Y. W. Rong Liu, "Signal Processing and Accelerometer-based Design for Portable Small Displacement Measurement Device," 07 2008. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4595612.
[7]	H. H. Stéphane Beauregard, "Pedestrian Dead Reckoning: A Basis for Personal Positioning," 2006. [Online]. Available: http://wpnc.net/fileadmin/WPNC06/Proceedings/33_Pedestrian_Dead_Reckoning.pdf.
[8]	C. G. P. J. W. K. H. S. H. a. J. M. L. S. H. Shin, "Adaptive Step Length Estimation Algorithm Using Low-Cost MEMS Inertial Sensors," 02 2007. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4248516.
[9]	D. C. M. H. Randell Cliff, "Personal position measurement using dead reckoning," 10 2003. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1241408.
[10]	H. J. J. D.-H. H. a. C. P. J. W. Kim, "A Step, Stride and Heading Determination for the Pedestrian Navigation System," 11 2004. [Online]. Available: http://www.gnss.com.au/JoGPS/v3n12/v3n12p34.pdf.
[11]	W. M. a. K. M. Bylemans I., "Mobile Phone-Based Displacement Estimation for Opportunistic Localisation Systems," 10 2009. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5361667.
[12]	H. Weinberg, "Using the ADXL202 in Pedometer and Personal Navigation Applications," 2002. [Online]. Available: http://www.analog.com/media/en/technical-documentation/application-notes/513772624AN602.pdf.
[13]	J. Scarlett, "Enhancing the Performance of Pedometers Using a Single Accelerometer," 2005. [Online]. Available: http://www.analog.com/media/en/technical-documentation/application-notes/47076299220991AN_900.pdf.
[14]	V. R. a. G. L. Muhammad Haris Afzall, "Assessment of Indoor Magnetic Field Anomalies using Multiple Magnetometers," 09 2010. [Online]. Available: http://plan.geomatics.ucalgary.ca/papers/gnss10_multiplemagnetometers_harisa_20sep10.pdf.
[15]	T. Babb, "How a Kalman filter works, in pictures," [Online]. Available: http://www.bzarg.com/p/how-a-kalman-filter-works-in-pictures/.
[16]	L. L. a. G. L. Tao Lin, "Multiple Sensors Integration for Pedestrian Indoor Navigation," 10 2015. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7346785.
[17]	S. R. Iwase T., "Infra-free indoor positioning using only smartphone sensors," 10 2013. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6817864.

[18]	K. C. W. M. T. G. Ascher C., "Dual IMU Indoor Navigation with particle filter based map-matching on a smartphone," 09 2010. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5646861.
[19]	G. Android, "Sensors Overview," [Online]. Available: http://developer.android.com/guide/topics/sensors/sensors_overview.html.
[20]	D. Sachs, "Sensor Fusion on Android Devices: A Revolution in Motion Processing," 08 2010. [Online]. Available: https://www.youtube.com/watch?v=C7JQ7Rpwn2k.
[21]	Q. L. Shuozhi Yang, "Ambulatory walking speed estimation under different step lengths and frequencies," 07 2010. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5695802.
[22]	Q. Ladetto, "On foot navigation : continuous step calibration using both complementary recursive prediction and adaptive Kalman filtering," 2000. [Online]. Available: http://www.ladetto.ch/LAQ/publications/ion2000_ql.pdf.
[23]	A. E. C. B. L. N. Ning Wang, "Accelerometry based classification of gait patterns using empirical mode decomposition," 04 2008. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4517685.
[24]	A. E. R. S. C. B. L. N. Ning Wang, "Classification of walking patterns on inclined surfaces from accelerometry data," 07 2009. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5201202.

Appendices

You can find additional logs and paths as well as the matlab files associated to all figures in the report.

Main folder, contains the report and hereafter mentioned sub directories	**http://tinyurl.com/PDRProject**
Test procedure with source file (Word)	**http://tinyurl.com/PDRProject-procedure**
Matlab files to produce plot and comparisons present in the report	**http://tinyurl.com/PDRProject-matlab**
Tons of logs and paths to test and compare different methods with matlab	**http://tinyurl.com/PDRProject-logs**