AnDReck: Positioning Estimation using Pedestrian Dead Reckoning on Smartphones

Carlos Filipe Correia Sequeira de Jesus Simes
carlos.simoes@ist.utl.pt

Instituto Superior Técnico, Lisboa, Portugal

October 2013

Abstract

In recent years, there was a wide adoption of GIS devices, many of which include GPS technology. Most of the time, these functions are performed by a smartphone, being carried by its user for extended periods of time. Although the common accuracy values for this technology are acceptable for some applications, such as driving guidance and nearby places identification, when it comes to pedestrian navigation they are somewhat lacking. Smartphones include a varied set of inbuilt sensors, which can be leveraged to perform positioning tasks, using the information provided by them. This work analyses existing publications and systems, across different approaches, and proposes a PDR based solution architecture. An implementation that uses an adapted peak detection algorithm, a step length estimation algorithm, along with its calibration method, and a digital compass orientation estimation method is then described, along with an Android prototype. Tests for this solution are then performed according to a proposed methodology, and the results are then evaluated against those of both systems that were the basis for implementation and related work, in order to validate its applicability.
Keywords: GPS, accelerometer, Pedestrian Dead Reckoning, Step Detection, Step Length Estimation, Android

1. Introduction

In recent years, there was a wide adoption of GIS devices, which include GPS technology. Devices that use this technology provide its user with a positioning estimate, whose accuracy is dependent on a number of factors: surrounding environment, atmospheric conditions, satellite line of sight, among others. Although the common accuracy values for this technology are acceptable for some applications, such as driving guidance and nearby places identification, when it comes to pedestrian navigation they are somewhat lacking. This is due to the differences between vehicular and pedestrian navigation: vehicle roads versus pedestrian paths, continuous movement versus cyclic movement, reduced mobility versus high mobility.

Looking at the most common device for this purpose, a smartphone, it can be observed that it also includes other components that may provide means to attain better positioning accuracy, namely acceleration and magnetic sensors. Furthermore, information gathered from these systems may be coupled to GIS data, in order to complement shortcomings between them.

Herein, we propose a system that leverages smartphone technology in order to provide positioning with increased precision. The system should be precise, easy to deploy and portable, taking advantage of the widespread usage of smartphones and its capabilities.

2. Requirements

These were the objectives identified for the system:

- **Availability**: increase the opportunities for positioning
- **Accuracy**: improved positioning accuracy and precision
- **Practicality**: system set-up should not be too troublesome
- **Portability**: the system must not hamper pedestrian locomotion

The main objective of this system is to improve the current availability of positioning systems referred previously. Current systems require a good satellite visibility during positioning. The system should be able to maintain positioning during navigation through terrain with low sky visibility.

Another requirement is to provide an accurate estimation of positioning. Again, satellite visibility

conditions current systems in their accuracy levels. Thus, this system should be able to not degrade the precision and accuracy levels in lower visibility conditions.

In order to have a wide acceptance, the system should be as ready to use from the start as possible, with minimal set-up steps. This not only enables the system to be useful as soon as possible, but also, by removing complexity, it becomes less susceptible to failures, and thus, more recalibrations.

Finally, the system should be portable during usage. Since the use case is during pedestrian navigation, it is essential that the system hampers locomotion as little as possible. Also, this ties in with the availability requirement, since a portable system will be more easily deployed everywhere.

3. Background

There are two main approaches to dead-reckoning for pedestrian applications: by integration of the acceleration values or by step detection. The integration method obtains the current acceleration value and the time since the last measurement to update the current speed.

The other approach, that avoids these errors, does not use the integrated acceleration values directly to discern distance. This is the case of Pedestrian Dead Reckoning systems, which use step events coupled with the step length to determine distance [3], combining it with some form of directional data to obtain positioning.

3.1. Pedestrian Dead Reckoning

As mentioned previously, one of the approaches to inertial navigation for pedestrian locomotion is to analyse the step movement, by detecting step events, estimating distance travelled in each of them and combining that with directional data to obtain a relative position.

Each of these components have different approaches, with different results and applications, making them appropriate to several different scenarios. The following paragraphs describe each of them in more detail, highlighting the different solutions available along with their advantages.

Step Detection Step detection using embedded systems has been a long running topic of research over the years, mostly in the computer science and signal processing domains. Although several different approaches are employed today with good results, most of them share one common trait: analysis of a signal, very frequently from an accelerometer sensor. This happens because acceleration sensors have become more and more inexpensive and, due to smartphones, ever more ubiquitous.

Acceleration values are indicative of changes in acceleration or lack thereof, and so, direct analysis intuitively makes sense: if during human locomotion there are several moving body parts, using acceleration values to detect this movement is a good guess. Also, it intuitively makes sense that along the several stages that comprise a step, there are moments where the acceleration rises, followed by a fall. These changes in signal trend can be called peaks, and they are the basis of the peak detection method of step counting, that analyses a window (or buffer) of sensor values and, upon detection of a peak, determines if a step is represented in it [10]. This determination is usually based on threshold values, sometimes fixed, while others averaged across previous acceleration values [14]. This approach can be followed for the several axes of the sensor and/or its vector's magnitude, the latter sometimes denoted "effective acceleration" [15].

Step Length Estimation In order to determine the distance travelled in a detected step, step length estimations must be performed. Early renditions of these estimations were based in physiological models that mapped and averaged step length to height and sex of a person [2] [5]. Not only should more personal attributes be employed (such as age and weight), using an average step length to estimate the length of every single step will inevitably result in errors accumulating over a number of estimates.

Another approach relates step length to its period/frequency, based on a studied model that relates them linearly [18][13], and uses other means to calculate step lengths during the training phase. One of these methods involves gathering GPS readings during straight line runs and interpolating them, in order to extract step lengths and frequencies, and inserting those samples into a linear regression model to obtain its coefficients [7].

Orientation Estimation Estimating orientation is similar to step detection in the way that it usually involves manipulation of signals to obtain a value. Most approaches employ some form of magnetometer signal manipulation, often using accelerometers to determine the direction of gravity, and adjusting its values to the ERF. In order to deal with periodic magnetic disturbances, filtering may be applied to this value in order to smooth it.

Since these magnetic disturbances may be too great in some instances, some approaches involving other sensors have been considered. One of such approaches is to fuse both gyroscope and compass values, which should compensate the noise from the magnetometers, and the integration errors of an exclusively gyroscopic approach [9]. Smartphones are beginning to integrate gyroscope sensors more frequently, as have approaches that use them [1].

Another approach to manipulating gyroscope and

compass values is to employ map matching, coupled with particle filtering [13]. This involves generating a series of objects at start time, called particles, and moving them according to the latest direction and length values. Then, several checks involving a predetermined map are done, such as overshooting turns through walls and balconies, and corrections are performed until a valid particle movement is found [15].

One final approach considered was PCA, which uses earth reference frame acceleration values to determine the direction of travel [10]. Since these acceleration values are oriented towards the earth frame, applying the reffered PCA method over 2 or 3 dimensions of their vector values will return one or more new vectors whose direction is the dominant direction of all of those values, by order. Due to the properties of the method and the values used, it can be said that one of these so-called eigenvectors represents the direction of travel [16].

3.2. Related Systems

The NavMote is a lightweight embedded device, featuring a wireless connection and sensor capabilities, designed to gather compass and accelerometer data and communicate with a wireless network of other devices, called NetMotes and RelayMotes, who then receive and process that data, to obtain a dead reckoning estimate of the pedestrian's trajectory [4].

The approach taken by this system is to reduce dead reckoning errors is to interpret step movement and detect its periodicity. Then, the step distance is estimated and recorded, so it can then be sent to a NetMote for processing, in order to save power. Wireless telemetry and map matching are also used to augment dead reckoning performance [4].

This dead reckoning approach, along with its step detection technique to reduce errors, is an interesting way to complement GPS navigation (or even be used standalone), even if the wireless and map matching components are ignored, since they require venue specific equipment and calibrations. The precision provided may not be the best, but for the specific application of locating a user inside a know room of a building it serves its purpose.

Early PDR systems were usually developed around the assumption the device would be carried in a fixed position, often strapped onto a belt buckle or helmet. This ensured acceleration values would be mostly output on the same reference frame, and thus had a fixed angle tranformation into Earth Frame Reference values. This approach, however, is misaligned with real world devices, that can not only be carried in multiple places, but also in different positions, depending on the user.

The Padati system [15] is a PDR system that uses step detection, a step length-frequency model and map matching with particle filtering as its approach, while maintaining orientation independence. This enables the system to be implemented in smartphone applications, which can be carried in several spots. Step detection is performed using peak detection on a Butterworth filtered effective acceleration signal, step length parameters are calibrated manually on a per-user basis and orientation is determined gyroscope readings coupled with the already referred particle filtering approach, which has also been used in other approaches [13].

This system presents acceptable accuracy results, without using GPS in any form, only the orientation independent effective acceleration values to detect steps and gyroscope values for direction. Once again ignoring the map matching component, the system presents itself as a very suitable PDR implementation for real-world applications, apart from the fact that its step length model is manually calibrated.

PDR systems require a step length parameter to work, using it as the measure of distance for each step. The most simple way of supplying this parameter is by using a fixed step length, possibly calculated as a function of weight, leg length and possibly other physical traits. This would be fine if the step length was constant, which is only true if pace changes do not occur. With the step length variability coming into play, accumulation of errors may have a significant impact in the total calculated length.

Taking this into account, the AutoGait system [7] derives a step length model (called Walking Profile) from the user's GPS and step data, using it to determine the duration of the each step and its length. This model then relates step length with its frequency (which is well validated [18][13]) using a linear least squares fitting of the gathered data. Tests performed by the developers showed 98% accuracy, with differences of up to 26% in error rates, when compared to a fixed step length model [7].

Such a system could be useful due its adaptability to each user, and the fact that it was developed using smartphone technology shows good promise for future similar implementations. Seeing as smartphone step counters are common, combining both could be a good approach, provided only leveled paths are considered, and the accuracy of both the step detector and GPS signal are good [16][14].

4. Architecture

When developing an architecture for a given system, one must keep in mind what objectives were set upon for it to solve, and what restrictions were placed to fulfil those objectives. Recalling the previously mentioned objectives, "the proposal for this work was study work done in related areas, includ-

ing relevant systems, and use that to build a system prototype that is capable of outputting relative positions in indoor and outdoor conditions, in the course of pedestrian locomotion.".

INS systems present themselves as an improved approach, regarding the "Availability" objective, when compared to GPS, since sky visibility does not affect them. "Accuracy" in these low sky visibility scenarios was also considered, where GPS' performance degrades as it decreases, while those of INS do not. The "Practicality" is not too hampered by the calibration steps due to their periodicity, and the "Portability" requirement is attainable through a smartphone implementation.

4.1. Architecture Overview

As discussed previously in this document, PDR is an appropriate approach to take for a solution to the presented problem. From the two approaches that were examined, the step detection with step length estimation approach will provide good results depending on the step length estimation. On the other hand, the integration with step detection approach has the issue of accumulated errors (commonly referred to as drift), which is even more aggravated with low-cost sensors.

The step length estimation problem occurs due to the fact that the most common way to estimate length is static: either an averaged value for all steps (based in factors such as gender and height [2]) or for each type of activity (walking, jogging or running) [8]. A better way to estimate step length should involve a scientific model that relates it to another observable value. Such is the case of the step frequency to length linear relation, which has been studied in the past [18][13], and even implemented [7].

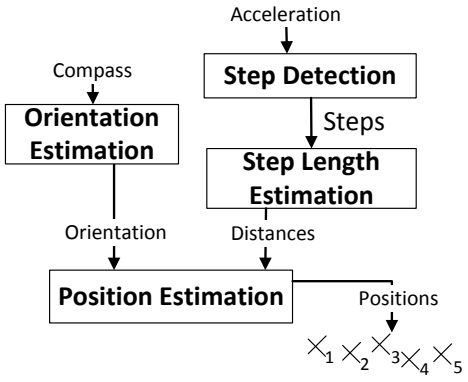

Figure 1: High-level components of the system and information flow

The problem with the integration approach occurs precisely because the drift (accumulated acceleration error) grows indefinitely. This is happens even faster if the sensors used are low-cost, since each update accumulates even higher errors. Relevant solutions use a technique called low-velocity updates [17], which consists of resetting the acceleration errors whenever a step is detected. Errors are still present, but they do not accumulate as quickly.

The AutoGait platform explores the former approach, building a step length model, which can relate the length of a step to its frequency. This may then be used during step detection to estimate the length of each movement, using the inverse value of the elapsed time between this and the last step.

Combining the output of these components (distance and direction), using trigonometric expressions, will result in a relative position output. Absolute positioning conversion is possible, as long the initial absolute position is known.

4.2. Architecture Detail

The proposed architecture is designed accounting for two different usage scenarios: Calibration and Positioning Estimation. In the Calibration scenario, step and location data are used to calibrate the step length model, while in the Positioning Estimation scenario step and orientation data, along with the calibrated model, are used to generate positions iteratively.

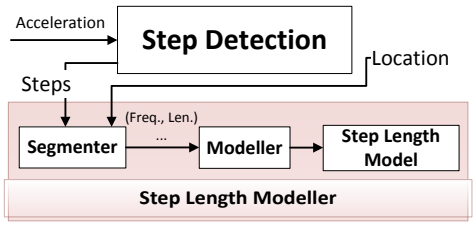

Figure 2: Calibration components and information flow

The Calibration scenario records detected steps, while gathering location data, in order to process them into an averaged step frequency and length average. Each of these samples, are then input into a modeller, that builds the step length model as these samples arrive.

After the Calibration scenario is performed and a step length model is built, the Positioning Estimation scenario can be performed. This scenario also depends on the step detector, described above, as input for the other two components: Distance and Orientation Estimation.

The Distance Estimator component receives the steps generated by the step detector and uses them for two purposes: calibration of the step length model and actual distance estimation. In the first case, location and step information is combined to

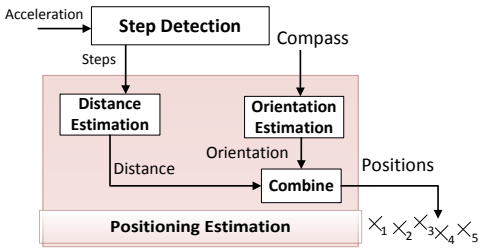

Figure 3: Positioning Estimation components and information flow

determine the approximated length of each step, which is then stored. For the second case, only step information is received, which we complement with the information we gathered during calibration, by adding a length to each step.

The Orientation Estimator receives the orientation information and detected steps, combining them as follows: for the duration of each of the received steps, combine all received orientation values into a single one.

Finally, the position estimator receives input from both the distance and direction estimators, combining them into a position using the method described previously.

4.2.1 Step Detection

The most common signal analysed to detect steps is the acceleration signal, with several different approaches. One of these approaches involves peak detection which, as the name implies, detects peaks (points in the signal that are preceded by a rise and followed by a slope) and analyses conditions in order to determine if it belongs to an instance of a step. This means that this type of step detector must be able to discern odd peaks in the signal resulting from non-step events, such as equipment shaking, foot contact or involuntary body contact.

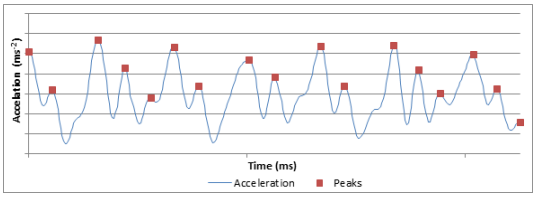

Figure 4: Example of peaks as step candidates in the acceleration signal

An example of a step detection implementation such as this is present in [14], detecting peaks as they appear within a 100 sample buffer (like the ones identified in 4), averaging the value of those peaks, and counting only those that rise above a percentage of that value. The proposed implementation is based on this approach, which uses the magnitude of the acceleration signal for the analysis, which is the norm of the 3D vector. However, observed results from this approach were not satisfactory. So, some adjustments were made in order to improve the step detection accuracy, by selecting the appropriate peaks.

First, it was observed that the signal that reached the analyser had two peaks after odd steps (as described in [12]). Since sometimes these secondary peaks would be high enough to be counted as steps in the original implementation, the following method was devised to exclude them safely: a peak can only be counted as a step, along with the other conditions, if the elapsed time since the last one is bigger than a percentage of the average for previous steps. This ensures that these "twin" peaks would be safely excluded, since highly rapid pace changes are unlikely, and maximize the number of properly timed steps that are counted. Other studied approaches in the past also analysed peak distance when validating steps [3].

Secondly, it was observed that before even steps there would be a valley (in contrast to peaks) that was considerably lower than the valleys in that pair of steps. In order to strengthen the analyser, a valley detection process was added, which would count valleys that were below a given value. Then, only two steps could be counted until the next valley was detected, indicating that a pair of steps was taken, as was stated in the beginning of this paragraph.

4.2.2 Distance Estimation

There were fewer approaches for distance estimation, but the AutoGait approach [7] claimed to have promising results. This coupled with the fact that it adapted to each user, was based on an established scientific model [18][13] and was designed for smartphone technology from the get-go (including the calibration step), making it easy to deploy, made it a very promising approach to follow.

The AutoGait system works by building a Walking Profile for each user, through the gathering of straight-line GPS segments, along with its steps. At the end of each segment, the step frequency is averaged, along with the corresponding length of each step. This average step length value is calculated by taking the length of the segment, and dividing it by the number of steps. The pair of averaged frequency and length values is then inserted into a Linear Regression model which, with enough samples, would output a linear function defined by its slope and intercept values.

It is this function that represents an approximation of the linear relation between step length and

frequency (called Step Length Profile, in the bibliography's implementation). Once the model is built, it can be consulted during step detection in order to obtain the length of a given step with frequency f, obtained by calculating the inverse of the elapsed time between this step and the last.

4.2.3 Orientation Estimation

Direction estimators are similar to step detectors in the way that they receive sensor signals and perform some sort of analysis on them. Some of these use world coordinated accelerometer signals to perform PCA and discover a dominant directional vector among the samples [11]. The most common approach, however, involves using the digital compass, which is a combination of both acceleration and magnetic signal values, as the direction value [9][1].

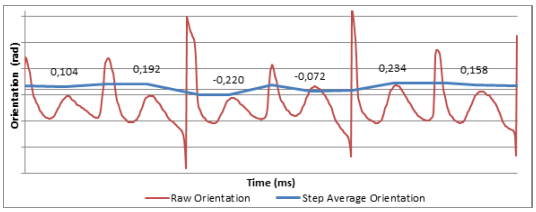

Figure 5: Combination of several Orientation values during a Step

Since several compass values are collected up to the point of Orientation Estimation itself, they should be combined into a single one. The method considered for doing this was to average all values during a step, an approach already explored in past implementations of PDR systems [6]. Figure 5 shows both Raw and Step Averaged orientation values, displayed on top of the blue plotline, in Radian units.

4.2.4 Positioning Estimation

Finally, the Positioning Estimator combines these three sources of information into positions. After each step, a distance estimation is fetched from the step length model, and, combined with the direction during that movement, a set of offset coordinates is generated. These offset values are calculated by multiplying the estimated step distance by the cos and sin of the estimated orientation value, for X and Y offsets respectively.

Assuming the user starts in the (0, 0) position, we add the offsets to this value to obtain the new position, and do the same process after each step. These positions are relative to the starting point, but if an absolute position is gathered (using GPS), a new absolute position can be calculated after each update.

5. Android Prototype: SensorDirection

The developed application uses a library (named "Analysis library") that was created in order to modularize the different architecture components. In general terms, it collects sensor data, processes and records it and the result of those operations into a file for later analysis.

This data collection should be performed during pedestrian locomotion, with the smartphone device placed in a place that favors step detection, like a trouser pocket. Beyond its main purpose of data collection, the application can also display the data generated during the calibration mode, also offering the possibility of importing and exporting that data for later use. It is also possible to archive application logs (storing them in an external folder), e-mail or clear them.

The Analysis Library With multiple signal transformations and analysis steps, a modular structure of code becomes almost essential. This enables both ease of swapping components, even during run-time, and good separation of filtering and analysing functions. If required by future developers, it also helps new developers integrate a new filter function or analyser into an existing system, provided they have a basic understanding of how the library functions.

This library has two main building blocks you can use to build a new system: Reading Sources and Analysers. Between them, Sensor Readings are exchanged, manipulated and analysed, resulting in the expected output. Reading Sources include Raw and Filter versions, which, in addition to buffering the readings, also smoothes them. Filter versions include Butterworth and Moving Average implementations. Analysers perform more complex signal processing operations, and are used to implement several architecture components. Implementations include Step Counters, Step Length Model calibration and Positioning Estimation (which includes the Orientation Estimation in itself).

5.1. Application Operation Modes

The application has two modes of operation to fulfil different purposes: one for the calibration stage and the other for actual positioning estimation. Each of them fulfils tasks which are self-explanatory from their descriptions, and they share a common structure between them.

Part of this is the fact that both of them register listeners to specific types of sensors, and that they take the output of each of those sensors and input them into the specified components of the Analysis library, using a different thread, for processing purposes. One of the components that is present in both modes is the Step Detector component, which

receives the acceleration data from the accelerometer sensor and outputs steps.

AutoGait Model Calibration This operation mode is used to calibrate the Step Length model, using both the accelerometer and GPS sensors, attaching the former to a Butterworth Filter of order 10 and a 5Hz cut-off frequency, which is itself attached to a Step Analyser. An AutoGait Modeller Analyser receives the detected steps and GPS readings, building a step length model.

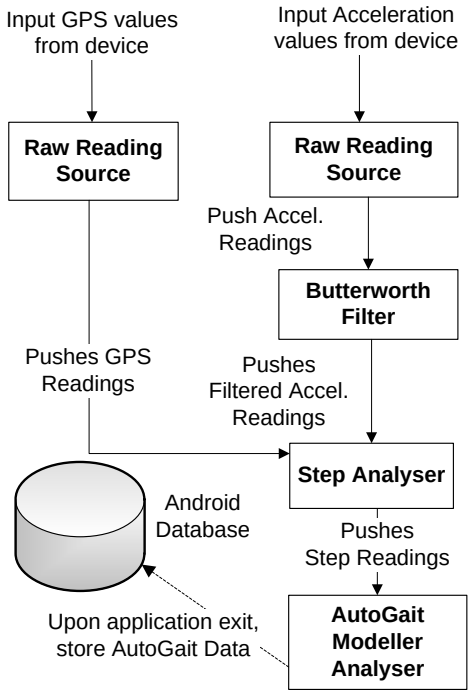

Figure 6: Storage of AutoGait samples in an Android database

When finished, the state required to rebuild an equivalent model is stored inside an Android application DB, to be retrieved the next time both modes are started.

Position Estimation This operation performs the positioning estimation itself, attaching the acceleration sensor to a similar assembly of components from the last operation mode, but instead of attaching to an AutoGait Modeller Analyser, the Step Analyser is attached to a Positioning Analyser, which also receives orientation readings.

When starting, this mode retrieves the samples required to initialize the step length model, in order for the Positioning Analyser to be able to estimate the distances of each step, necessary for the calculation of each position, as well as the average sample rate of the last run, for filter initialization purposes.

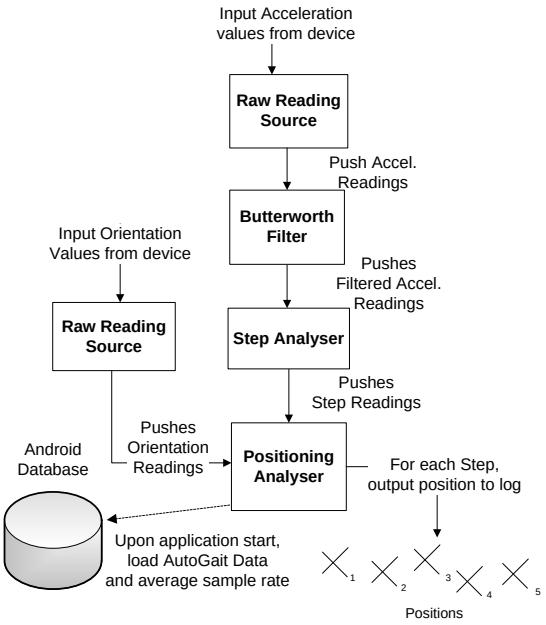

Figure 7: Retrieval of AutoGait samples stored in an Android database

6. Results and Test Methodology

The methodology chosen to validate the solution was to perform multiple tests, of varying characteristics, and examine each of the system's components separately. This makes comparison with related systems that focus on single components possible, and gives a more granular perspective on the system performance. Also, due its different characteristics, Walking and Running tests were analysed separately, reflecting the different expectations for each case, where appropriate.

Since the system components perform step detection, step length estimation and orientation estimation, each of these functions should be evaluated against the real world values, which means they need to be measured in some form. So the steps should be counted, and step distance and orientation measured. Step counting was done using a chronometer with lap times, counting a lap for each step. This has the added benefit that step times are also recorded, which makes it possible to map accelerometer output to actual steps, as well as measuring the timing accuracy of the step counter.

Step length is not trivial to measure, since it involves individual measuring of each step. A considered option to do this was to perform measurements in a sand floor: footprints would be made for each step, and measured individually afterwards. However, walking on sand significantly alters walking dynamics, which would influence the results. Instead, a previously measured track was travelled, and each of its steps' estimated length added up.

Since it affects step distance, a fairly levelled field should also be considered [7]. The relationship between the measured distance and the real distance should provide a good overview accuracy of the system, while an averaged approach for each step length value should also be considered to provide an estimate of how much error is incurred during each of them. Since the calibration procedure for this component requires GPS visibility, this should also be taken into consideration.

Finally, the orientation estimation component is also tricky to evaluate, since a course may have a certain shape and not be traversed in the same manner by each user, be it due to distractions or other random occurrences. For that reason, a simpler scenario was considered: the track to travel has several lanes, each layed out in a straight-line shape. The straight-line shape is one of the most simple shapes to be travelled, and the floor marking should minimize human error by providing guidance. The output direction is then considered on a step-by-step basis, and this trajectory compared against that of straight line. The method chosen to define this straight line is to gather averaged compass readings at the beginning of each test. This provides the direction to maintain during the whole test, and the value to compare the estimations to.

Grouping these considerations together as a requirement, a suitable location was then chosen. It should have appropriate pavement, have straight line lanes marked, as well as their distance, and should be fairly levelled. Some cycle routes have distance markings along them, but not all of them have significant stretches of straight lines, nor GPS visibility, needed to perform calibration of the step length model.

Such a site was discovered near the "Amadora Este" Metro station, with ground measurement marks every 25 meters, with a straight line of up to 600 meters. Several runs, of varying distance and cadence, were made this test site. Since the track is a straight line, there are very little differences while travelling forwards and backwards. The distances considered were 100, 400 and 500 meters for walking tests, as well as 100 and 200 meters, for running tests.

The considered tests were performed in their respective step cadences (either walking or running), in their various distances, with a Samsung Galaxy W I-8150 Android placed in one of the trousers' pockets, while inspecting the track markings for the travelled distance. In order to help prevent manual miscounting of steps, an additional Android phone with a chronometer was used, performing a lap-time for each step. The result of each test was then stored for later analysis, along with the captures from the main Android phone. The average results of the Step Detection component for the walking tests were above 97.34%, which amounts to an undetected step every 33 that occur. Subtracting the standard deviation to this value (resulting in a 95.97% accuracy value), results in missing a step every 25 taken.

Still analysing the walking tests, the average value of the number of step errors was -12, deviated by 10 steps. The confidence interval in this case is [-22;-1]. When comparing to the interval the similar scenario (trousers' front pocket, 500 meter walking) presented in the publication which was the basis for the implemented step counting algorithm [14], which was [-11;29], there is an intersection in values, hinting that both results are close. Since the interval is smaller (because of a smaller standard deviation value) this indicates that the system is more precise.

Finally, step time offsets were averaged 36.4 milliseconds, with an interval of [20;52.4]. In the worst case scenario, with the step model considered for for this evaluation (with an α parameter of 0.16, and a β parameter of 0.52), this value will result in an error of 8 millimeters, which for every 20 steps would result in a 16 centimeter error.

Running tests present results that are partly worse. An average Step Count Accuracy value of 96%, with a standard deviation of 3.66% means that the system will lose a step every 12. Assuming the same scenario of a constant step length of 1 meter, this means losing that same meter after covering 12 meters. Step Count Errors for these tests are smaller than those of the walking test, with a resulting interval of [-11.1;0.44]. This indicates a good count performance in terms of step count results. The Step Time Offset interval was [23;97] in milliseconds, bigger than that of the walking tests, indicating a poorer performance in accurately detecting step times in these conditions.

The Average Distance Accuracy value of 95.83%, with a deviation of 2.42% are below those in AutoGait's publication (98% on average) [7], but considering that for their results a different type of step counting technologies was used and did not miss any steps to be counted, they are satisfactory. Other systems showed 96% [6] and 97.5% [4] results for similar conditions, which also reinforces this statement. These results are still much better than those of a fixed step length model, as shown in AutoGait's publication, with errors ranging from -25.7% and 14.6%. This proves that using this step length estimation model is better than using no model at all.

The average Accumulated Step Length error of 20 centimeters, is suspicious, considering the standard deviation value is 2.08 meters. A possible explanation for this is that some tests present an high

value for this parameter, due to the fact that initial step times are not estimated correctly. This happens because the step time average value, described in section 4.2.1 as the first adaptation to the step counting algorithm, might not have been calculated with enough samples yet, resulting in a threshold value too restrictive to allow legitimate steps to be counted, instead counting longer steps.

Looking at the results for the running tests, the poor performance of the step length estimation model for running steps is confirmed. Distance Estimation Accuracy Averaged at 66.35%, with a standard deviation value of 2.25%, along with a an average Estimated Step Length Offset of 14.1 centimeters deviated by 2.25 centimeters, the performance is much weaker than that of the walking tests. This had already been hinted in AutoGait's publication itself [7] and others [4].

The Orientation results are presented in the form of a comparison between the raw orientation values of a digital compass, the same values smoothed through a sliding mean process and the implemented process of calculating the average of the orientation values during a particular step. The metrics considered are mostly around angular offsets from a reference orientation, but also in the form of metric distance for the Step Mean estimation method.

Comparing average Orientation Offset values between orientation offset value types, it is clear that the Step Orientation estimation method is superior to the others, strengthened by the fact that its standard deviation values are also lower, showing less spread. For these same values, running tests perform better than their walking counterparts, a fact that can only be explained by the fact that these experiments are performed in shorter distances (coinciding with lower values for 100 meter walking tests).

In order to expose the impact of these values, the lateral distance offset from the predicted trajectory was also analysed, with the worst results being 1.2 and 1.5 meters and the best 2.5 and 3.7 centimetres, which are good results. Since there are weak results in both running and walking tests, as well as long and short, no correlation was found between the test characteristics and their outcome.

When looking for the results of related work, it was hard finding some that measured orientation estimation error to evaluate that platform. Steinhoff's work [16], performs this analysis, presenting results for median, 75th and 95th Percentile orientation errors. Its best solution performs better than the one proposed, beating even the results of the running tests. This indicates that this component could use some adjustments in the future, in order to perform to its best potential.

7. Conclusions

Given the increasing need of location capabilities with both high availability and accuracy, a system was proposed, compiling several related solutions and joining them together into a single prototype. Systems with augmented GPS technology were considered, but preference was given to accelerometer sensor approach. In this category, both Acceleration Integration and PDR approaches were analysed, and due to their characteristics, preference was given to PDR.

An architecture for the proposed solution was presented, defining the several components: Step Detection, Distance and Orientation Estimation, which are then combined into Positioning Estimation. A signal processing framework was then created, in order to modularize its several components. Using this framework, an Android prototype was developed, taking advantage of the ample development documentation, API capabilities and portability, with two scenarios for the prototype were chosen: calibration and data collection.

After development, the prototype was evaluated across its step detection, distance and orientation estimation capabilities. Positive results were obtained for the Step Detection component, when compared to the publication on which it was based on[14], in particular with a smaller standard deviation. The Distance Estimation results were mostly positive, and while they were not on par with the AutoGait publication [7], other systems had closer results [6][4], and a clear improvement over a fixed length step model was still clear. Finally, Orientation Estimation results for the Step Averaging method showed an improvement over raw and sliding mean filtered orientation signals, with less average orientation offset and standard deviation. When compared to related work, however, the results were not so positive, indicating that the approach followed could be improved.

8. Future Work

Future approaches to consider should consider methods to increase positioning opportunities, such as activity detection and alternate phone positions [15]. Alternative techniques for Orientation Estimation should be considered, such as Kalman Filtering positions with GPS and low-pass filtering orientation values before estimation [6]. Finally, external Step Counters are also interesting because they increase battery life, and have alternate algorithms and positions, which increase accuracy.

References

[1] S. Ayub, A. Bahraminisaab, and B. Honary. A Sensor Fusion Method for Smart phone Orientation Estimation. *The 13th Annual Post-Graduate Symposium on the Convergence of*

Telecommunications, Networking and Broadcasting, 2012.

[2] S. Ayub, X. Zhou, S. Honary, and A. Bahraminasab. *Computer, Informatics, Cybernetics and Applications*, volume 107 of *Lecture Notes in Electrical Engineering*. Springer Netherlands, Dordrecht, 2012.

[3] S. Beauregard and H. Haas. Pedestrian dead reckoning: A basis for personal positioning. *Proceedings of the 3rd Workshop on Positioning, ...*, pages 27–36, 2006.

[4] L. Fang, P. Antsaklis, L. Montestruque, M. McMickell, M. Lemmon, Y. Sun, H. Fang, I. Koutroulis, M. Haenggi, M. Xie, and X. Xie. Design of a Wireless Assisted Pedestrian Dead Reckoning SystemThe NavMote Experience. *IEEE Transactions on Instrumentation and Measurement*, 54(6):2342–2358, Dec. 2005.

[5] P. D. Groves. *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. 2007.

[6] D. Gusenbauer, C. Isert, and J. Krosche. Self-contained indoor positioning on off-the-shelf mobile devices. In *2010 International Conference on Indoor Positioning and Indoor Navigation*, number September, pages 1–9. IEEE, Sept. 2010.

[7] W. J. Kaiser, M. Gerla, D.-K. Cho, M. Mun, and U. Lee. AutoGait: A mobile platform that accurately estimates the distance walked. In *2010 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 116–124. IEEE, Mar. 2010.

[8] A. M. Khan. Human Activity Recognition Using A Single Tri-axial Accelerometer Human Activity Recognition Using A Single Tri-axial Accelerometer. 2011.

[9] J. W. Kim, H. J. Jang, D.-H. Hwang, and C. Park. A step, stride and heading determination for the pedestrian navigation system. *Journal of Global Positioning Systems*, 3(1):273–279, 2004.

[10] M. Kourogi and T. Kurata. Personal positioning based on walking locomotion analysis with self-contained sensors and a wearable camera. *Proceedings of the 2nd IEEE/ACM International ...*, 2003.

[11] K. Kunze, P. Lukowicz, K. Partridge, and B. Begole. Which Way Am I Facing: Inferring Horizontal Device Orientation from an Accelerometer Signal. *2009 International Symposium on Wearable Computers*, pages 149–150, Sept. 2009.

[12] Q. Ladetto. On foot navigation: continuous step calibration using both complementary recursive prediction and adaptive Kalman filtering. *Proceedings of the 13th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS 2000)*, (13):1735 – 1740, 2000.

[13] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao. A Reliable and Accurate Indoor Localization Method Using Phone Inertial Sensors. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing - UbiComp '12*, page 421, New York, New York, USA, 2012. ACM Press.

[14] M. Mladenov and M. Mock. A step counter service for Java-enabled devices using a built-in accelerometer. In *Proceedings of the 1st International Workshop on Context-Aware Middleware and Services affiliated with the 4th International Conference on Communication System Software and Middleware (COMSWARE 2009) - CAMS '09*, page 1, New York, New York, USA, 2009. ACM Press.

[15] D. Pai, I. Sasi, P. S. Mantripragada, M. Malpani, and N. Aggarwal. Padati: A Robust Pedestrian Dead Reckoning System on Smartphones. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 2000–2007. IEEE, June 2012.

[16] U. Steinhoff and B. Schiele. Dead reckoning from the pocket - An experimental study. In *2010 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 162–170. IEEE, Mar. 2010.

[17] X. Yun, E. R. Bachmann, H. Moore, and J. Calusdian. Self-contained Position Tracking of Human Movement Using Small Inertial/Magnetic Sensor Modules. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, number April, pages 2526–2533. IEEE, Apr. 2007.

[18] W. Zijlstra. Assessment of spatio-temporal parameters during unconstrained walking. *European journal of applied physiology*, 92(1-2):39–44, June 2004.