An optimised algorithm for accurate steps counting from smart-phone accelerometry

Dario Salvi, Carmelo Velardo, Jamieson Brynes, Lionel Tarassenko

***Abstract*— Step counting from smart-phones allows a wide range of applications related to fitness and health. Estimating steps from phones' accelerometers is challenging because of the multitude of ways a smart-phone can be carried. We focus our work on the windowed peak detection algorithm, which has previously been shown to be accurate and efficient and thus suitable for mobile devices. We explore and optimise further the algorithm and its parameters making use of data collected by three volunteers holding the phone in six different positions. In order to simplify the analysis of the data, we also built a novel device for the detection of the ground truth steps. Over the collected data set, the algorithm reaches 95% average accuracy. We implemented the algorithm for the Android OS and released it as an open source project. A separate dataset was collected with the algorithm running on the smart-phone for further validation. The validation confirms the accuracy of the algorithm in real-time conditions.**

I. INTRODUCTION

In the last 10 years, no other electronic device has had the same commercial and societal impact as the smart-phone. Mobile phone penetration is greater than that of personal computers and, in the United Kingdom, it has already reached peaks of 85% for people aged 17-75 [1]. Modern smart-phones embed a vast number of sensors connected to powerful processing units and capacious memories. Even low-end devices have built-in accelerometers, proximity sensors, gyroscopes, magnetometers and Global Positioning System receivers.

This technology allows for several applications, from navigation and positioning [2] to tracking for health and fitness purposes [3]. Particularly relevant for health and fitness, is the ability to measure physical activity. Step counting is the most commonly used parameter extracted in *apps* designed to measure and influence physical activity [4].

Recently, high-end smart-phones have often incorporated low-power step counters, which are part of the motion-sensing chip-set, but older or cheaper devices still lack this functionality. To compensate for the lack of hardware resources, mobile applications like Google Fit or Samsung Health compute the step count using the main processing unit, but the accuracy and the details of their algorithms are unknown. The lack of a proper evaluation of these products makes it difficult to employ them for medical purposes, where transparency about the accuracy and the limitations of the technology are needed. While the accuracy of wearable devices is being investigated widely [5], little is still known about mobile applications.

The research community, and developers in general, would therefore benefit from a validated, open-source implementation of a step-counting algorithm for smart-phones. Even though there are a number of open-source projects related to step-counting, to the best of our knowledge, none have been validated with robust methods. The aim of this paper is to describe how an algorithm for smart-phone based step counting has been designed, optimised and validated. The algorithm we propose makes use of the data generated by the accelerometer sensor, is robust to noise, is accurate enough to be meaningful for common fitness and health applications and is computationally efficient to be executable in real-time by a smart-phone. The algorithm has been implemented in Python and Java and is available as an open-source project at: https://oxford-step-counter.github.io/.

II. RELATED WORK

There have been a number of attempts to develop step counting algorithms based on an accelerometers signal. In [6] a relationship between walking speed and the accelerometer signals is used to devise an algorithm based on thresholding the magnitude. The algorithm was tested on five walking samples ranging from 16 to 44 steps and achieved an average accuracy of 96.6%. The data collection protocol was not detailed except for the fact that the device was fixed near the centre of mass of the subject.

In [7], the authors tested a multitude of algorithms for both walk detection and step counting. The paper describes a wide range of techniques, from simple thresholding in the time domain, to frequency domain analysis to non-linear template matching and machine learning. To compare them, authors decided to benchmark nine algorithms: windowed peak detection, mean crossing counts, normalized autocorrelation, dynamic time warping, short term Fourier transform, continuous wavelet transform, discrete wavelet transform, hidden Markov model, and k-means clustering. They collected 130 data recordings from 27 subjects holding a smart-phone in 4 ways: in-hand, in a front pocket, in a back pocket, and in a handbag. A video recording was taken of each session for reference. Authors determined that the windowed peak detection, the hidden Markov model and the continuous wavelet transform methods worked the best across the scenarios, with a median error of about 1.3%. Given the computational complexity of the two latter techniques, authors concluded that the windowed peak detection is the most efficient algorithm.

All the authors are with the Institute of Biomedical Engineering, Department of Engineering Science, University of Oxford, Oxford, United Kingdom. dario.salvi@eng.ox.ac.uk, carmelo.velardo@eng.ox.ac.uk, jamieson.brynes@some.ox.ac.uk, lionel.tarassenko@eng.ox.ac.uk.

This work was supported by NIHR Oxford Biomedical Research Centre.

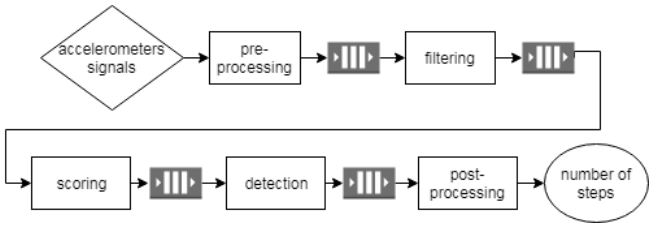

Fig. 1. Block diagram of the step counting algorithm. Between each pair of stages there is a buffer to allow parallelism.

In [8], Gu et al. present an algorithm based on the peak detection method that is also robust to false positives. Their approach consists in adding constraints to the peak detection in terms of periodicity (time difference between two neighbouring peaks), similarity (peak distance between two windows of acceleration), and continuity (number of neighbouring windows of acceleration readings with the variance surpassing a threshold). These features allow filtering out "false walking", e.g. when users remain still and use their phone for writing text messages, calling, watching videos, playing games, etc. Eight volunteers were recruited and asked to walk 300 steps with a smart-phone in different motion states: (a) walking with the phone in a fixed phone pose, (b) walking with the phone in arbitrary phone poses and (c) walking for some segments with false walking states. The average error produced by their algorithm ranged from 3.54% in the walking state to 14.04% in the false walking state. They compared these performances with common fitness applications, which performed worse especially when in the presence of false walking.

Unfortunately, none of these studies have made the source code of the algorithms available, nor the data sets used for their validation.

III. METHODS

A. Step counting algorithm

To estimate the number of steps from the smart-phone accelerometer signals, we used the windowed peak detection technique because of its efficiency, as suggested by [7]. Particularly, our algorithm is based on the approach proposed by [9] and can be split into 5 stages, each responsible for a particular function (see figure 1).

To allow fast processing of the data, each block can be instantiated in a parallel thread, while buffers are used to hold the data temporarily between each block. For each stage we present a set of options and parameters that need to be optimised.

1) Pre-processing stage: Common smart-phones provide accelerometry signals over three orthogonal axes; however the step-counting algorithm is concerned with the magnitude rather than any single directional component because the physical orientation of the device is unknown. Although common smart-phone operating systems allow a desired sampling rate for the accelerometer to be set, there is no guarantee that the sampling rate will be kept with precision.

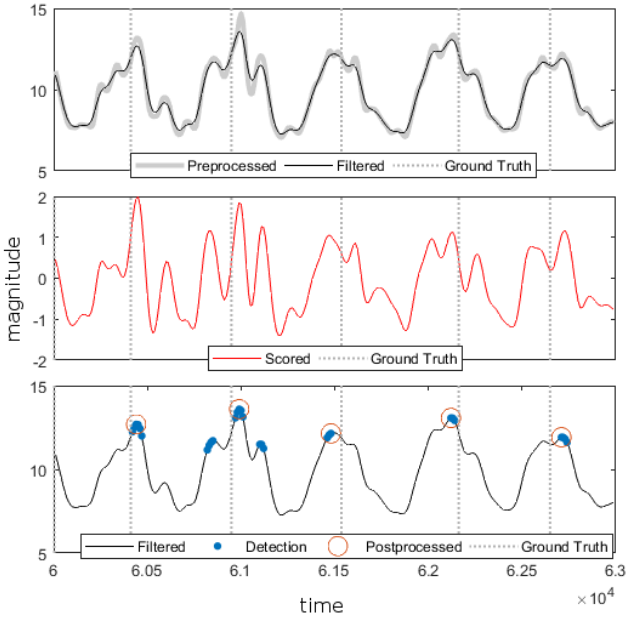

Fig. 2. Signal at each processing stage. The ground truth shows the moment when a step was detected by the ground truth device. This signal may be slightly delayed due to latency introduced by the wireless connection.

Therefore this stage is responsible for computing the magnitude of the triaxial accelerometry signal and ensuring a constant sampling frequency by means of linear interpolation.

Within this work we have assumed the sampling frequency to be 100 Hz, but the algorithm does not depend on this value.

2) Filtering stage: Accelerometers are subject to noise from a variety of sources (mechanical, electrical, thermal, etc.), therefore some noise-reduction technique is needed especially at frequencies that are not related to human walking or running. In order to reduce the noise level, we have implemented a finite impulse response (FIR) low-pass filter with a cut-off frequency of 3 Hz, which allows a variety of walking speeds. This value should include even the pace of the speediest walkers, which was identified as 5.4 mph (8.7 km/h) by the American College of Sports Medicine [10], with a ratio of 2000 steps per mile (1250 steps per km). An example of the raw accelerometer signal after interpolation and filtering is shown in figure 2.

We consider different types of window functions for evaluation: moving average, Gaussian window, Hann window and the Kaiser-Bessel window.

3) Scoring stage: The function of the scoring stage is to evaluate the *peakiness* of a given sample. The result of this stage should increase the magnitude of any peaks, making them more evident for the subsequent peak detection. Different methods were considered:

- Maximum Difference: This considers the N previous samples and determines the maximum difference between these and the sample in question, then computes the same for the N samples to the right and averages

the two maximum differences as the result. If p_i is the peakiness of sample x_i then for $1 \leq k \leq N$:

$$p_i = \frac{\max_k(x_i - x_{i-k}) + \max_k(x_i - x_{i+k})}{2} \quad (1)$$

- Mean Difference: This method is similar to the Maximum Difference method, except that instead of taking the maximum of the difference, it uses the mean of the differences. This preserves the overall shape of the waveform.

$$p_i = \frac{\sum_{k=-N,k\neq i}^{N}(x_i - x_{i+k})}{2N}, \quad (2)$$

- Modified Pan-Tompkins Scoring: This is an adaptation of a well-known algorithm used for peak detection in electrocardiogram waveforms [11]. Given a window of N samples, the algorithm zero-means the data within the window, sets all the negative data points to zero and squares the samples.

An example of the output from the scoring stage is shown in figure 2.

4) Detection stage: This stage identifies potential candidate peaks to be associated with a step by statistically detecting outliers. As the algorithm processes the signal, it keeps track of a running mean and standard deviation. These two quantities are used to determine whether any given sample is an outlier. If the difference between the sample and the mean is above c times the standard deviation, then it is marked as a potential step. The parameter c is considered for optimisation.

5) Post-processing stage: This stage slides a window of a fixed size, t_{window}, across the potential peaks and only keeps the maximum sample within the window. As the small dots in figure 2 show, all of the potential peak points are clustered around the rise to the main peak. This stage selects the local maximum among them. By tuning the window size, one can limit the number of steps within the window, thus not allowing peaks less than t_{window} apart. This approach can be tuned to accommodate the periodicity of human walk. We set the window length to 200 ms, corresponding to a maximum walking speed of 5 steps/s, which is faster than the speediest walking pace (3 steps per second). The effect of this stage can be seen with the circles in figure 2.

B. Ground truth data collection

The proposed algorithm was fine-tuned and validated against a self-collected dataset composed of raw accelerometry data and a ground truth signal. Previous works have used two approaches to ground truth collection: manual step counting [8] or video annotation [7]. Although simple to implement, these methodologies either do not provide step-by-step information, which may help understand the behaviour of the algorithm, or are time intensive and prone to error.

We used a custom-built electronic device able to derive step-by-step information. The device is placed in the shoes of the participants, allowing the detection of each single step. Once the walking signal is captured and converted to a binary format (i.e. foot down / foot up), the device sends the result to a smart-phone application through a Bluetooth Low Energy (BLE) link. The same application can be used to record the raw accelerometry data from the phone. The parallel recording of the two signals on one device allows to have them synchronised effortlessly.

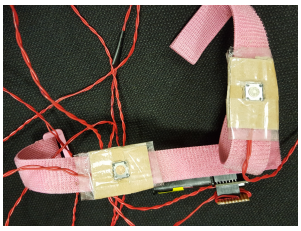

Fig. 3. First version of the ground truth device. Push buttons are placed under the tip of the shoes and connected to the board through wires.

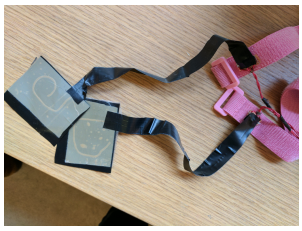

Fig. 4. Second version of the ground truth device. Pressure-sensitive foils are placed in the shoe under the heel. A conductive thread is used to connect the foil to the board.

The ground truth device was built using the RFduino BLE-enabled Arduino-compatible prototyping board. The device underwent two iterations: in the first, two common push buttons were placed under the tip of the shoe. While effective in detecting steps, this solution did not prove robust as the buttons broke after a few recording sessions. For the second iteration we placed a pressure-sensitive foil, Velostat, between the foot and the insole. Velostat is a polymeric foil impregnated with carbon black that makes it electrically conductive, and is used primarily for packaging. The properties of the material are such that if subject to an electric current, it changes the value of its resistance when pressed or stretched. Two 5x5 cm Velostat foils were wired to the analogue-to-digital converters of the board where special-purpose firmware was developed to distinguish between 'weight-on' and 'weight-off' states. The firmware uses an adaptive threshold to determine the transition between the two states. The threshold adapts as the medium point between the minimum and the maximum value. Every five seconds, the minimum is increased and the maximum decreased of 10% of the difference. This approach guarantees that the threshold adapts to changes in the signal (e.g. if the foil moves inside the shoe).

The device samples the resistance of the two foils at 50 Hz and transmits it over BLE to the phone. The mobile app logs both the accelerometer signals and the ground-truth signal and provides a simple user interface for controlling the recording session (a start/stop button and a timer). The software also allows these files to be sent remotely to an HTTP endpoint after having compressed them.

Figures 3 and 4 show the 2 versions of the device and figure 5 shows one of the signals generated by stepping on Velostat foils.

The device was validated by comparing its output with manually counted steps. Steps were counted by two observers and disagreements resulted in the test being discarded. Exper-

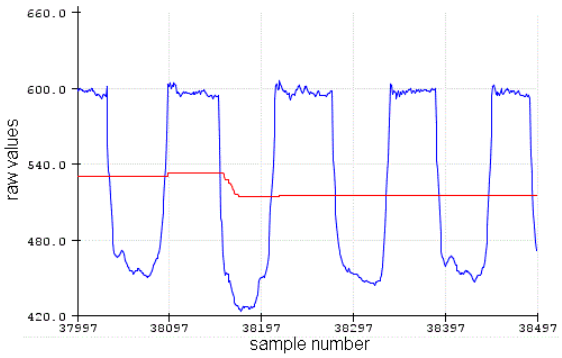

Fig. 5. Digitised value of the voltage measured on one Velostat foil (blue) and the computed threshold (red).

iments were performed over short distances with the number of steps varying between 22 and 79 (average 36 ± 14 standard deviation) and time varying between 14 and 41 seconds (average 24$\pm$8 s).

Of 17 experiments, 12 showed no difference in the number of steps counted by the device and the researchers, 4 had a 1-step difference and 1 had a 2-step difference. This yields to an average accuracy of 99% and standard deviation of 2%. We considered this accuracy to be sufficient for the purpose of this work.

C. Data collection

In order to optimise the parameters of the algorithm with different scenarios, we collected data from three researchers walking for two to three minutes with the phone held in six different positions: in a hand, in a front pocket, in a back pocket, in an armband, in a shoulder purse, and in a neck pouch on a lanyard. These scenarios are similar to the ones proposed by [7]. This resulted in a dataset containing 36 distinct recordings, each with approximately 2.5 minutes of accelerometer and ground-truth data. Six extra experiments were dedicated to assess the differences between three different phones (Google Pixel, Samsung Galaxy S6, LG Nexus 5) in similar conditions: phone held in hand, same floor and same user.

To validate the algorithm when running on the phone, 12 extra recordings were collected by two researchers holding the phone in the same six positions as the ones used for optimising the algorithm.

D. Algorithm optimisation

To select the optimal set of parameters for our algorithm an exhaustive grid search across the parameter space was performed. The explored range of the parameters is shown in table I together with parameters that were kept constant. The sampling frequency was fixed at 100 Hz for all the tests.

Both the algorithm and the benchmarking software were developed in Python version 3. The optimisation of the parameters was performed on the 36 recordings of the scenarios dataset. In order to expedite the exhaustive search, a distributed cloud-computing platform (Google Cloud Compute) was employed to spread the processing load across multiple machines. Each permutation of parameters was ranked according to the mean error obtained on all the recordings.

TABLE I

SET OF PARAMETERS OPTIMISED FOR EACH STAGE AND MINIMUM, MAXIMUM AND STEP VALUE.

Stage	Parameters	Min	Step	Max
Filtering	Window size	13	8	53
Scoring	Window size	3	8	51
Detection	Threshold	1.2	0.2	1.4

TABLE II

PHYSICAL CHARACTERISTICS OF THE VOLUNTEERS WHO COLLECTED THE DATA AND AVERAGE $\pm$ STANDARD DEVIATION OF THE DURATION AND NUMBER OF STEPS OF THE COLLECTED TRACKS.

Age	Weight(kg)	Height(cm)	Duration(s)	Steps
38	86	180	163$\pm$20	537$\pm$50
33	80	186	171$\pm$27	481$\pm$58
21	68	172	153$\pm$18	547$\pm$79

IV. RESULTS

The details of the three researchers (all male) who collected the data are summarised in table II together with average and standard deviation of the duration and number of steps.

Table III shows the optimal set of parameters obtained by the distributed optimisation phase. With these set of parameters, the average accuracy is $95\% \pm 4.5\%$ standard deviation.

If we split the analysis according to the position of the smart-phone during the recordings we obtain the results in table IV (each scenario has six recordings). It is possible to observe that the *in-hand* and *purse* positions give rise the lowest accuracies. These may be caused by the harmonics

TABLE III

SET OF PARAMETERS THAT OBTAINS THE MINIMUM AVERAGE ERROR.

Stage	Parameters	Value
Filtering	Window size	M=13
	Filter type	Gaussian
	Filter SD	0.35
	Cutoff frequency	3Hz
	Gain at cutoff	-60dB
Scoring	Type	Mean Difference
	Window size	$N = 35$
Detection	Threshold	1.2
Post-Processing	t_{window}	200 ms

TABLE IV

ACCURACY OF THE OPTIMISED ALGORITHM PER POSITION, N=6 FOR EACH POSITION

Position	Average accuracy $\pm$ SD
In hand	$93\% \pm 3.8\%$
Front pocket	$95\% \pm 2.8\%$
Back pocket	$98\% \pm 1.2\%$
Purse	$90\% \pm 7.2\%$
Armband	$95\% \pm 2.5\%$
Neck pouch	$98\% \pm 1.0\%$

TABLE V

ACCURACY OF THE OPTIMISED ALGORITHM PER PHONE MODEL

Phone model	Accuracy recording 1	Accuracy recording 2
Google Pixel	97.9%	95.30%
Google Nexus 5	96.1%	97.3%
Samsung S6	99.6%	95.5%

introduced in the walking frequency range by the arm swinging when holding the phone, or by the purse swinging back and forth under the arm.

If we analyse the six recordings that use different smartphones (results in table V) we can conclude that the choice of the phone does not affect the accuracy of our algorithm significantly, which confirms the findings of [7].

The average accuracy obtained with the 12 validation tracks collected with the algorithm running on the phone is $93 \pm 13\%$. This confirms that the algorithm performs with a similar accuracy also when running on a smart-phone. The high variance is due to one track taken in the back pocket when the algorithm counted almost twice the number of steps. The reason may be due to the fact that the pocket was loose and allowed the phone to rebound and thus introduced components in the signal at twice the frequency of the gait.

V. CONCLUSIONS AND FUTURE WORK

Our paper builds on the results of [7]. We agreed to their recommendation of using the Windowed Peak Detection and we explored further different design choices and parameters. We confirm that the accuracy of the algorithm is above 90% for the dataset on which it is optimised and, in addition, that the performance is similar when the algorithm runs on the phone with data it was not optimised on. We also show a different approach for the collection of ground truth data that is more scalable and we provide all our datasets and code as open source to the community.

There are limitations to this study that could benefit from further investigation. All our recordings were gathered during walking on a hard floor and it will be necessary to investigate softer surfaces like carpet or grass. To optimise parameters, further scenarios will be added to the analysis such as running and climbing stairs. Also, longer datasets would be needed from more subjects to confirm the findings.

We have not evaluated scenarios of false walking activities as was done in [8]. For these scenarios, Gu et al. offer some valid additions to the Windowed Peak Detection that would need to be added to our implementation.

REFERENCES

[1] P. Lee, "Mobile Consumer Survey 2017: The UK cut," Deloitte [Last accessed: 27 September 2017], Tech. Rep., 2017. [Online]. Available: http://www.deloitte.co.uk/mobileuk/?_ga=2.95282970.782041114.1506509099-13344498.1506509099

[2] H.-E. Chen, Y.-Y. Lin, C.-H. Chen, and I.-F. Wang, "Blindnavi: A navigation app for the visually impaired smartphone user," in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '15. New York, NY, USA: ACM, 2015, pp. 19–24.

[3] J. P. Higgins, "Smartphone Applications for Patients' Health and Fitness," *The American Journal of Medicine*, vol. 129, no. 1, pp. 11–19, Jan. 2016.

[4] J. Bort-Roig, N. D. Gilson, A. Puig-Ribera, R. S. Contreras, and S. G. Trost, "Measuring and Influencing Physical Activity with Smartphone Technology: A Systematic Review," *Sports Medicine*, vol. 44, no. 5, pp. 671–686, May 2014.

[5] P. Alinia, C. Cain, R. Fallahzadeh, A. Shahrokni, D. Cook, and H. Ghasemzadeh, "How accurate is your activity tracker? a comparative study of step counts in low-intensity physical activities," *JMIR mHealth and uHealth*, vol. 5, no. 8, p. e106, 2017.

[6] N. Z. Naqvi, A. Kumar, A. Chauhan, and K. Sahni, "Step counting using smartphone-based accelerometer," *International Journal on Computer Science and Engineering*, vol. 4, no. 5, pp. 675–681, May 2012.

[7] A. Brajdic and R. Harle, "Walk detection and step counting on unconstrained smartphones," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. ACM, 2013, pp. 225–234.

[8] F. Gu, K. Khoshelham, J. Shang, F. Yu, and Z. Wei, "Robust and accurate smartphone-based step counting for indoor localization," *IEEE Sensors Journal*, vol. 17, no. 11, pp. 3453–3460, 2017.

[9] G. Palshikar, "Simple algorithms for peak detection in time-series," Tata Research Development and Design Centre, Tech. Rep., 2009.

[10] W. W. Hoeger, L. Bond, L. Ransdell, J. M. Shimon, and S. Merugu, "One-mile step count at walking and running speeds," *ACSM's Health & Fitness Journal*, vol. 12, no. 1, pp. 14–19, 2008.

[11] J. Pan and W. J. Tompkins, "A real-time QRS detection algorithm," *IEEE transactions on biomedical engineering*, no. 3, pp. 230–236, 1985.