2016 IEEE 40th Annual Computer Software and Applications Conference

Self-adaptive Step Counting on Smartphones under Unrestricted Stepping Modes

Zihao Tang, Yao Guo, Xiangqun Chen
Key Laboratory of High-Confidence Software Technologies (Ministry of Education),
School of Electronics Engineering and Computer Science, Peking University, Beijing, China
{tangzihao, yaoguo, cherry}@pku.edu.cn

***Abstract*—Pedometer apps on smartphones and wearable electronics are increasingly popular nowadays, as they are widely used for health monitoring and location-based systems. Most pedometer apps are based on inertial sensors and each step counting algorithm works precisely under restricted stepping modes because steps are detected and validated after comparing their parameters to pre-determined optimal values related to particular conditions. In this paper, we propose self-adaptive step counting in order to improve step counting accuracy under unrestricted stepping modes on smartphones. Based on our human stepping model, we propose self-adaptive method that can detect new steps by monitoring vertical acceleration and validate new steps by comparing to self-adaptive values, which are adjusted dynamically after each step occurs. We show the flexibility of the proposed approach by incorporating it into two existing step counting algorithms WPD and DTW. We also propose a new stepping cycle recognition (SCR) algorithm that is self-adaptive and performs the best under variant stepping modes. With experiments under different stepping modes including fixed and variant modes, we show that self-adaptive methods perform significantly better compared to original methods using fixed optimal values.**

I. Introduction

Pedometers are useful for everyday exercise measurement and health monitoring, as well as indoor location tracking and navigation systems [1][2][3][4][5]. For example, using step counters, we can calculate the number of calories we consumed or the average step lengths. With the increasing ubiquity of smartphones, pedometer apps on Android and iOS are becoming increasingly popular [6]. Currently, there are hundreds of pedometer apps on Google Play [7], most of which count steps by analyzing data obtained from inertial sensors such as accelerometer and gyroscope. Besides smartphones, pedometers are also implemented on many wearable electronics.

Most existing algorithms [8][9][10][11][12] work by comparing acceleration data to pre-determined values, such as the peak, the frequency or even the whole trace. For example, dynamic time warping (DTW) is a popular algorithm often used for measuring similarity of two time series such as the variation of acceleration in stepping. DTW can be used for step counting to match a gait template that is identified manually for every tester [8][13]. These methods work accurately under fixed stepping modes with parameters conforming to optimal values.

However, pedometer apps need to work under unrestricted stepping modes including various walking and running conditions. The placement and holding positions of smartphones could also vary, such as holding in hands, pockets or handbags [8][14]. Furthermore, different people have different stepping habits. The optimal parameter values are different for variant stepping conditions. As most existing algorithms cannot adjust optimal values dynamically, they have difficulties to adjust to arbitrary unrestricted stepping modes.

In this paper, we introduce *self-adaptive step counting* to increase the accuracy of pedometer apps on smartphones to accommodate to unrestricted stepping modes. In particular, we apply the self-adaptation idea to two existing step counting algorithms, namely WPD and DTW [8], and also propose a new algorithm called step cycle recognition (SCR), which is self-adaptive in nature and able to adapt to new stepping modes more accurately.

When implementing the existing techniques, we also take advantage of the gravity sensor and linear accelerometer that have been equipped on new smartphones such as Google Nexus 5 [15]. With the gravity sensor, we detect steps by monitoring vertical acceleration based on the human stepping model and validate steps by comparing the detected steps with sample steps similar to existing algorithms. The main difference is that the samples steps used in the adaptive algorithms are dynamically adjusted after new steps occur.

In addition to demonstrate the performance of self-adaptation on existing algorithms, we also propose and implement a more precise and stable step counting algorithm called *stepping cycle recognition* (SCR). Because SCR is self-adaptive in nature, it can overcome the limitations in the existing WPD and DTW algorithms.

We have tested the self-adaptive algorithms in over 100 cases under 7 different stepping modes including 5 walking modes, a running mode, and a variant mode with a combination of different conditions. In the evaluation of 285 data traces, we show that all proposed methods work much more precise than the Android native counter, while the self-adaptive versions are significantly better than the original algorithms with fixed thresholds. In addition, the proposed new SCR algorithm performs the best in most situations, especially for complicated variant stepping mode.

We make the following main contributions in this paper:

- This paper introduces self-adaptive step counting to in-

0730-3157/16 $31.00 © 2016 IEEE
DOI 10.1109/COMPSAC.2016.187

crease the accuracy of pedometer apps on smartphones to accommodate to unrestricted stepping modes. The self-adaptive method is widely applicable, as it can be integrated into existing popular algorithms such as DTW and WPD, replacing their fixed optimal values with dynamic values to increase their accuracy.
- We also propose a new Stepping Cycle Recognition (SCR) algorithm that is more accurate and stable than existing algorithms.
- We demonstrate the effectiveness of self-adaptation with experiments involving various stepping configurations.

The rest of the paper is organized as follows. Section II presents the background and related work of step counting on smartphones. Section III describes how self-adaption applies to existing step counting algorithms such as WPD and DTW. Section IV describes limitation of the adaptive WPD and DTW algorithms and presents the basic idea and implementation of the proposed SCR algorithm. Section V presents the experimental data including performance and energy consumption of different algorithms. Section VI discusses the results and future work, and Section VII concludes the paper.

II. Background and Related Work

A. Step Counting

Traditional pedometers count steps by inertia with a metal pendulum or steel ball inside, while modern pedometers have built-in MEMS 3-axis accelerometer and microcomputer to distinguish a real step from different types of vibrations [5][16][17].

Pedometers apps on smartphones work by analyzing data from built-in inertial sensors, using algorithms such as peak detection, zero-cross counting, acceleration differential and/or step distance computation [9]. The counters are incremented when the values of all parameters are beyond the threshold and/or in the range [8][9][10][12].

In a recent work, Jayalath *et. al* presents a gyroscope based step counting algorithm under different activities and variant speed and their average accuracy is above 96% [10].

The study of walking patterns recognition is relevant to our work. By using acceleration signals, it is possible to recognize different walking patterns such as walking slowly, walking fast and going up and down stairs [18][19][20].

The most relevant to our work is from Brajdic and Harle [8]. They tested the performance of 9 step counting algorithms using accelerometer on smartphones under 6 different phone placements. The best overall algorithm is windowed peak detection (WPD) with error rates of less than 3%. Although they have also targeted at different phone placement, they have not considered various walking conditions such as the speed of walking or running. In fact, we have integrated our self-adaptive method with two of the algorithms from their work.

B. Using Linear Accelerometer and Gravity Sensors

Although most existing algorithms have been using accelerometers, this paper tries to take advantage of the gravity sensors and linear accelerometers, which have been equipped on the newest smartphones such as Google Nexus 5 [15].

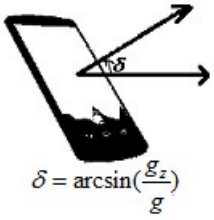

Fig. 1. Orientation of a smartphone.

We obtain acceleration without influence of gravity (a_x, a_y, a_z) from linear accelerometer and components of gravitational acceleration in tri-axial directions of the smartphone (g_x, g_y, g_z) from gravity sensor. Based on the data, we compute the horizontal acceleration (a_h), vertical acceleration (a_v) and orientation (δ) of the smartphone [15][21]. a_h is an absolute value and a_v is positive when it is upward and negative when it is downward [17]. Furthermore, δ is the directed angle of the smartphone orientation and the horizontal plane (as shown in Fig. 1), which ranges 0 to 90° when the smartphone faces up and 0 to -90° when it faces down.

Zhou *et. al* presented a method to compute the orientation (δ) using the gravity sensor with average errors of about 10° in their work [22]. We use the following equations to calculate the horizontal acceleration (a_h) and vertical acceleration (a_v):

$$g = \sqrt{g_x^2 + g_y^2 + g_z^2}$$
$$a = \sqrt{a_x^2 + a_y^2 + a_z^2}$$
$$a_v = \frac{a_x g_x + a_y g_y + a_z g_z}{g}$$
$$a_h = \sqrt{a^2 - a_v^2}$$

C. The WPD Algorithm

Brajdic and Harle [8] used a centered moving average (optimal value 0.31s) to smooth the acceleration magnitude. They then applied a *windowed peak detection* (WPD) algorithm (optimal value 0.59s) to get the peak values associated with heel-strike. When walking regularly, the peak values vary in a small range close to their optimal values.

In their work, the WPD algorithm with error rates of less than 3% is the best overall algorithm for step counting regardless of smartphone placement [8]. As using linear accelerator and gravity sensor instead of accelerator, our improved WPD algorithm analyzes the peak value of horizontal acceleration, vertical acceleration, total acceleration and orientation. To be more precise, we also focus on the frequency data, such as the time length of a step and the stride frequency [23].

D. The DTW Algorithm

Dynamic time warping (DTW) is another popular algorithm often used for measuring similarity of two time series such as the variation of acceleration in stepping [13].

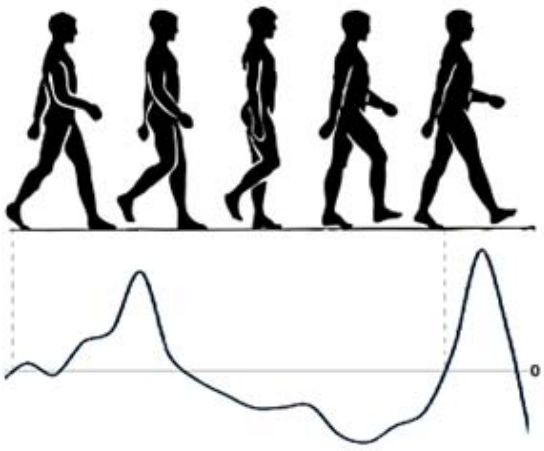

Fig. 2. Variation of vertical acceleration of a step.

Brajdic and Harle also implemented a DTW-based step counting algorithm to match a gait template that was identified manually for every tester [8]. To reduce errors, they varied minimal step length (optimal value 0.25s) and maximal distance (optimal value 0.92s) between 2 steps [8].

In contrast, we use the series of horizontal acceleration, vertical acceleration and orientation for matching. Besides, we also make use of stride frequency to decrease errors.

III. Self-Adaptive Step Counting

This section first presents our human stepping model, then discusses how to apply self-adaptive methods on the existing WPD and DTW algorithms based on new parameters.

A. Human Stepping Model

Before introducing self-adaptive methods, we first establish a human stepping model as the basis of step counting. As usual, we define a step as a motion that starts when lifting a foot off the ground and ends when the foot returns to the ground [24][18].

Fig. 2 shows that vertical acceleration turns to upward direction from near-zero or downward direction when lifting a foot off the ground, and then turns to downward direction when the foot is stepping forward, and finally turns to near-zero from downward direction when the foot returns to the ground [19].

As a result, we consider the time when the vertical acceleration is just turning to upward direction from downward direction as the end time of a previous step and the start time of the next step. Based on earlier publications, the time length of each step is typically between 0.3s to 1.2s.

In the following sections, we use the human stepping model to detect the start/end time of steps and apply self-adaptive step counting to the validation of steps.

B. Self-adaptive WPD

The basic idea of the WPD algorithm is comparing the measurement values of several parameters to their optimal values. A step will be validated only if all measurement values are close to optimal values. Comparing to the original WPD algorithm, two main changes have been made in our self-adaptive version of the WPD algorithm:

- We check multiple-dimensional parameters including acceleration in different direction and orientation. Besides, we check stride frequency instead of using moving average.
- We use self-adaptive optimal values that are adjusted dynamically during stepping.

Based on the gravity sensor and the linear accelerator, we check nine stepping parameters including five peak values of acceleration, two peak values of orientation and two parameters of stride frequency. Fig. 3 illustrates the acceleration and orientation during stepping and the nine stepping parameters. There will be a validated step only when all measurement values are approximate to their optimal values, and after which the step counter will increase. The nine parameters used in this algorithm are listed as follows:

- VAU_{max} : The maximum of upward acceleration.
- VAD_{max} : The maximum of downward acceleration.
- HA_{max} : The maximum of the absolute value of horizontal acceleration.
- HA_{min} : The minimum of the absolute value of horizontal acceleration.
- TA_{max} : The maximum of total acceleration.
- ORI_{max} : The maximum of directed angle between the orientation of smartphone and horizontal plane.
- ORI_{min} : The minimum of directed angle between the orientation of smartphone and horizontal plane.
- TLS : The time length of a step (from the start time to the end time).
- SF : The time length of contiguous steps (from the start time of the previous step to the start time of the next step).

Fig. 4 shows the variation of horizontal acceleration and vertical acceleration in walking and running, in which we see that acceleration change faster and larger during running than walking. Apparently, peak values and the time series of each step are quite different. Therefore optimal values of one given mode may not adapt to another mode. To adapt to every condition, we adopt self-adaptive optimal values instead of constant optimal values.

Fig. 5 shows that when stepping regularly, the variation of the smartphone orientation is fluctuating regularly. Similarly, frequency parameters including the time length of step (TLS) and the stride frequency (SF) vary in a small range respectively in a series of steps [19]. To some extent, if the measurement SF of a new step is very close to its optimal value, we could validate the step even several measurement values are not close to their optimal values.

Fig. 6 shows the basic workflow of the self-adaptive WPD algorithm. The biggest difference is that in non-adaptive methods, the sample steps always remain unchanged, while in self-adaptive methods, we dynamically adjust optimal values

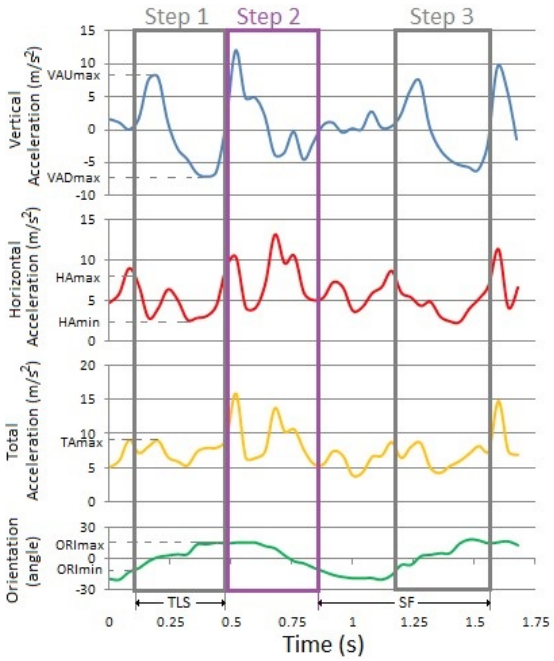

Fig. 3. Illustration of acceleration and orientation, as well as the nine stepping parameters.

or sample steps after each step even the step is invalidated.

Table I shows the initial optimal parameter values reached after multiple experiments and the units of parameters.

We introduce a probabilistic model to compute the probability of a step when the starting time and the end time of step are determined. The probabilistic model is based on the proximity of every measurement value (x) of parameters to the optimal value (μ) (see Table I). The result is between 0 to 1 and a new step will be validated if the result is above 0.7. The initial model is:

$$p = \prod_{i=1}^{8} {p_i}^{\alpha_i} + \beta \cdot (p_9 - \gamma)$$

where $p_i (1 \leq i \leq 5)$, $p_i (6 \leq i \leq 7)$ and p_8 are the distance of parameters about acceleration, orientation and TLS of their measurement values from their optimal values, respectively. $\alpha_i (1 \leq i \leq 8)$ adjusts p_i and p is calculated by ${p_i}^{\alpha_i} (1 \leq i \leq 8)$ directly. p_9 is the distance of the measurement value of SF from the optimal value of SF.

If the measurement value of SF is much smaller than its optimal value, the time may probably not be the start time or the end time of a step. Thus no step occurs and p_9 will be very close to 0. Conversely, if the measurement value of SF is close to its optimal value, there probably will be a step because of the periodicity of stepping. β and γ are used to adjust p_9. The values of $\alpha_i (1 \leq i \leq 8)$, β and γ are shown in Table II.

After reaching the probability threshold of a new step, self-adaptive WPD adjusts the optimal value of every parameter. To avoid impractical optimal values of parameters about

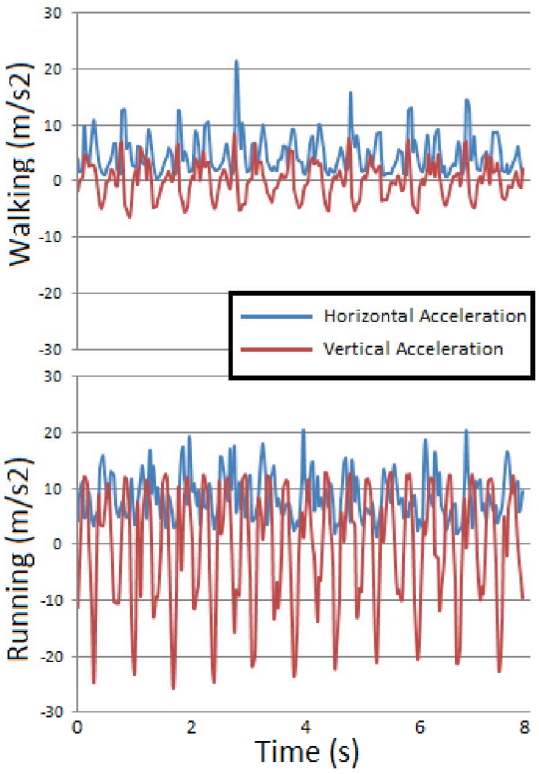

Fig. 4. Comparison of horizontal and vertical acceleration during walking and running.

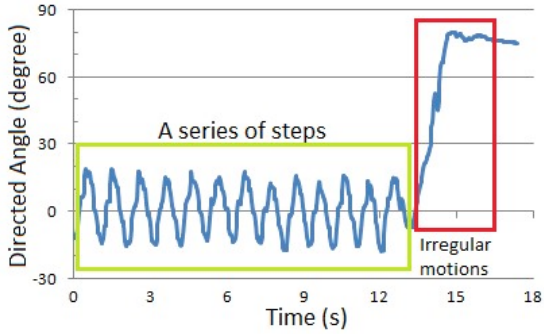

Fig. 5. Comparison of directed angle of steps and irregular motions.

acceleration, they are related to their initial optimal values:

$$\mu_{acce} = \frac{0.8 \cdot \sum\limits_{t_k \leq 5s} \frac{p_k}{t_k} \cdot \mu_k + 0.2 \cdot \sum\limits_{t_k \leq 5s} \frac{p_k}{t_k} \cdot \mu_0}{\sum\limits_{t_k \leq 5s} \frac{p_k}{t_k}}$$

As parameters about orientation and stride frequency may have great difference in different series of steps, their initial optimal values and their optimal values calculated at the beginning of a series of steps have low significance. Their optimal values are not related to their initial optimal values:

$$\mu_{ori/time} = \frac{\sum\limits_{t_k \leq 5s} \frac{p_k}{t_k} \cdot \mu_k}{\sum\limits_{t_k \leq 5s} \frac{p_k}{t_k}}$$

TABLE I
PEAK VALUES AND FREQUENCY INFORMATION USED IN THE SELF-ADAPTIVE WPD ALGORITHM.

Parameters	Initial optimal values	Range of optimal values	Proximity of measurement values (x) to optimal values (μ)	Units
$VAU_{max}(p_1)$	6	1 to 20	$p = \begin{cases} 1, if(0.75\mu \le x \le 2\mu) \\ \max\{0, 1-2\cdot(\frac{x-2\mu}{2\mu})^2\}, if(x > 2\mu) \\ \max\{0, 1-2\cdot(\frac{0.75\mu-x}{0.75\mu})^2\}, if(x < 0.75\mu) \end{cases}$	m/s^2
$VAD_{max}(p_2)$	4	1 to 10		
$HA_{max}(p_3)$	6	$max\{4, HA_{min}\}$ to 20		
$TA_{max}(p_4)$	9	1 to 30		
$HA_{min}(p_5)$	2	0 to $min\{6, HA_{max}\}$	$p = \begin{cases} 1, if(x \le 2\mu) \\ 1-0.25\cdot(x-2\mu)^2, if(x > 2\mu) \end{cases}$	
$ORI_{min}(p_6)$	-	-90 to ORI_{max}	$p = \begin{cases} 1, if(\lvert x-\mu\rvert \le 15) \\ \max\{0, 1-\frac{(\lvert x-\mu\rvert-15)^2}{500}\}, if(\lvert x-\mu\rvert > 15) \end{cases}$	degree
$ORI_{max}(p_7)$	-	ORI_{min} to 90		
$TLS(p_8)$	0.4	0.16 to 0.8	$p = \begin{cases} 1, if(\lvert x-\mu\rvert \le 3) \\ \max\{0, 1-\frac{(\lvert x-\mu\rvert-3)^2}{50}\}, if(\lvert x-\mu\rvert > 3) \end{cases}$	s
$SF(p_9)$	0.56	0.24 to 0.8	$p = \begin{cases} 1-\frac{(x-\mu)^2}{8}, if(\lvert x-\mu\rvert \le 2) \\ \max\{0, 0.5-0.8\cdot\frac{\mu-x-4}{x}\}, if(\mu - x > 4) \\ 0.5, otherwise \end{cases}$	

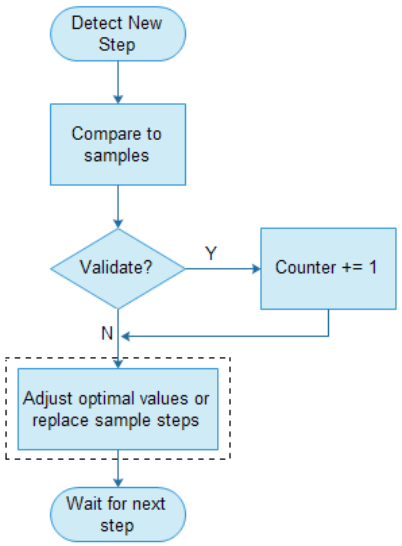

Fig. 6. Basic workflow of self-adaptive WPD. If the phase surrounded by dotted box is omitted, it becomes a non-adaptive WPD algorithms.

As stepping is on both legs and the smartphone is usually put next to one side of the body, such as in a pocket, hand or handbag, the testing data of different side of the body may result in differences. To reduce errors, we use two groups of optimal values to distinguish the different features of the motions of two legs. Each step belongs to one group. Consequently, the group of continuous steps is alternating under normal conditions.

C. Self-Adaptive DTW

To demonstrate that the self-adaptive idea is widely applicable, we also apply it to a DTW algorithm [8]. We reduce

TABLE II
VALUES OF $\alpha_i (1 \le i \le 8)$, β AND γ.

α_1	0.5
$\alpha_2, \alpha_3, \alpha_4, \alpha_5$	1
$\alpha_6, \alpha_7, \alpha_8$	$\min\{1, \frac{\sum_{t_k \le 5s} \frac{p_k}{t_k}}{0.07}\}$
β	0.6
γ	0.5

the searching space of DTW to increase efficiency. The time complexity becomes $O(n)$ instead of $O(n^2)$. We manually identify 10 groups of trace data from our experiments as original sample steps (OSS), with which we could validate new steps at the beginning. The time series of the latest validated steps are used as *self-adaptive sample steps* (SASS). They can validate the following steps and can be replaced by the following validated steps.

Each element of the trace data is a triple that consists of horizontal acceleration (a_h), vertical acceleration (a_v) and orientation (δ). The distance of two elements is defined as:

$$d_1(x,y) = 0.8 \cdot \max\{0, \frac{|ha_x - ha_y|}{\sqrt{\max\{\frac{1}{4}, \frac{ha_y}{2}\}}} - 1.5\}$$

$$d_2(x,y) = \max\{0, \frac{|va_x - va_y|}{\sqrt{\max\{\frac{1}{4}, \frac{|va_y|}{4}\}}} - 1.5\}$$

$$d_3(x,y) = \begin{cases} 0.08 \cdot \max\{0, |\delta_1 - \delta_2| - 5\}^2, SASS \\ 0, OSS \end{cases}$$

$$dist = \left(d_1^2 + d_2^2 + d_3^2\right)^{\frac{1}{4}}$$

Dynamic programming is used to compute the distance of two groups of trace data as above, where x and y refer to

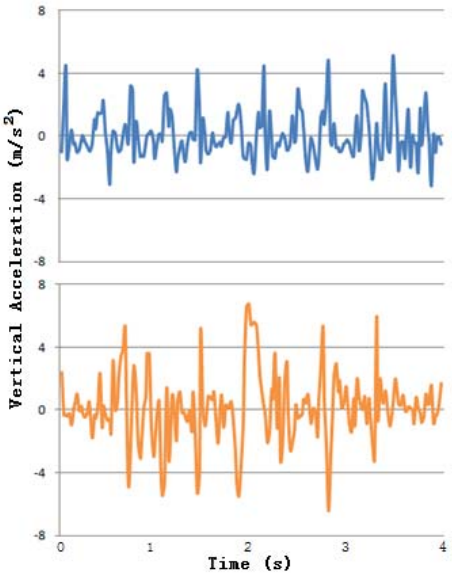

Fig. 7. Variation of vertical acceleration during stepping (top figure) and irregular motions (bottom figure) respectively.

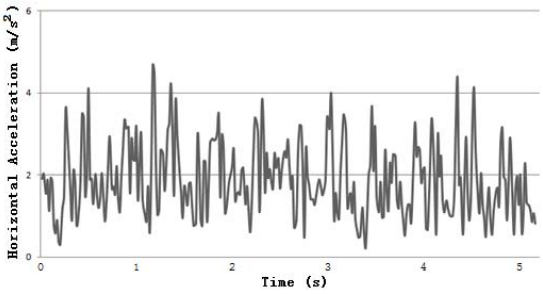

Fig. 8. Variation of horizontal acceleration during stepping.

TABLE III
VALUES OF α UNDER DIFFERENT VALUE OF $|x-y|$

Value of α	Value of $\lvert x-y\rvert$
3	$[0, 0.05s)$
9	$[0.05s, 0.1s)$
24	$[0.1s, 0.15s)$
∞	$[0.15s, \infty)$

the *x*th and *y*th element of each group, respectively. Smaller distance of trace data and smaller difference in time length means that two steps have more similarity. A step will be validated if it is similar to several sample steps. In our study, we use weighted average proximity as the criteria. We sort the proximity values in reverse order. Larger proximity value has larger weight and the weighted average of more values is more significant. Old sample steps will be replaced by the latest validated steps such that the algorithm becomes self-adaptive.

IV. THE SCR ALGORITHM

During experiments, we found that both the WPD and DTW algorithms have some limitations. WPD algorithm mainly focuses on peak and frequency parameters, while DTW mainly focuses on the variation of trace data. In this section, we will discuss the limitations of WPD and DTW, and then introduce a new algorithm called *Stepping Cycle Recognition*(SCR) to overcome their limitations.

A. Limitations of WPD and DTW

We use the following two examples to demonstrate the limitations of WPD and DTW.

Fig. 7 shows that the peak value of vertical acceleration may not vary a lot between regular stepping and irregular motion, which means that the variation of the trace data of vertical acceleration is more significant in validation of new steps than the peak value of vertical acceleration. Consequently, we need to consider the whole trace data of vertical acceleration instead of only the peak value.

Fig. 8 shows that the variation of horizontal acceleration may not be regular even during stepping. When validating new steps, we check horizontal acceleration to confirm whether there is forward trend and backward trend. So we only need to check the peak value of horizontal acceleration but not the whole trace data. In addition, the mean value and the standard deviation of orientation has greater significance in validating new steps than the peak value or the variation of the trace data mentioned above.

B. Distance of Steps

Based on the discussion above, we propose a new formula to calculate the distance of two steps, which is the sum of three terms. The first term is the distance of the trace data of vertical acceleration. We use the following equation to get the distance of vertical acceleration of two moments.

$$d_{a_v}(S[x].a_v, T[y].a_v) = \begin{cases} \alpha \cdot \sqrt{2} \cdot |S[x].a_v - T[y].a_v|, if(S[x].a_v \cdot T[y].a_v < 0) \\ \frac{\alpha}{2} \cdot \frac{|S[x].a_v - T[y].a_v|}{\sqrt{\max\{\frac{1}{2}, \min\{|S[x].a_v|, |T[y].a_v|\}\}}}, otherwise \end{cases}$$

We then apply DTW to get the distance of the vertical acceleration of step S and step T. $S[x]$ and $T[y]$ refer to the *x*th moment of step S and the *y*th moment of step T, respectively. Table III shows the value of α, which has positive correlation with the difference of x and y.

The second term is the distance of the peak value of horizontal acceleration. The third term is the overlap range of the orientation of two steps. The value of the third term is negative if the orientation of two steps have higher overlap range, while the value is zero if the orientation of two steps have no overlap range.

$$d_{\max\{a_h\}}(S) = 4 \cdot \max^2\{0, 5 - \max\{S[x].a_h\}\}$$

$$d_{\max\{a_h\}}(S,T) = \frac{|\max\{S[x].a_h\} - \max\{T[y].a_h\}|}{4 \cdot \sqrt{\min\{\max\{S[x].a_h\}, \max\{T[y].a_h\}\}}}$$

$$d_{\min\{a_h\}}(S) = \max^2\{0, \min\{S[x].a_h\} - 5\}$$

$$dist_{a_h}(S,T) = d_{\max\{a_h\}}(S) + d_{\max\{a_h\}}(T) + d_{\max\{a_h\}}(S,T) + d_{\min\{a_h\}}(S) + d_{\min\{a_h\}}(T)$$

C. Implementation of SCR

Based on the discussion, we propose a new *stepping cycle recognition* (SCR) algorithm, which is shown in Fig. 9. It is able to dynamically adjust the sample steps or the optimal values according to the steps occurring recently and compare the new steps to self-adaptive sample steps when validating new steps. On this basis, we can recognize the cycle of stepping by counting similar steps occurring in recent seconds. Specifically, a new step will be validated if the distance between the new step and sample steps is smaller than a threshold d_1 and there are at least $N(N \leq 3)$ steps with a distance to the new step smaller than a threshold $d_2(d_2 > d_1)$ occurring in recent ts. Sample steps will be dynamically adjusted after the validation of new steps, which is similar to the self-adaptive DTW algorithm. SCR is different in that new steps may be validated a few seconds after they occur.

The optimal values of d_1 and d_2 are 8 and 5, respectively. The values of t and N are related to the time length l of new steps. t_s is the tiny interval between contiguous steps and the value of t_s is 0.08s. When recognizing the cycle, if there are at most $N-1$ steps (not including excluded new steps) occurring in recent ts with a distance to a new step less than d_2, the new step does not follow the periodicity and will be excluded. As the DTW method is used in computing the distance of vertical acceleration, SCR replaces old sample steps with new steps after the validation of new steps, similar to self-adaptive DTW.

V. Evaluation

This section presents the experiments, including performance and power consumption of the algorithms proposed in this paper.

A. Experimental Setup

We have implemented our algorithms on Android and conducted experiments with Google Nexus 5 under six fixed stepping modes and a variant stepping mode. The six fixed stepping modes include walking with smartphone in trousers front pocket, walking with smartphone in trousers back pocket, walking with smartphone in coat pocket, walking with smartphone held in hand, walking with smartphone in handbag and running with smartphone in trousers front pocket, respectively. The variant mode is composed of traces from the six fixed modes mentioned above. A trace in the variant mode is a combination of 2 to 4 fixed stepping condition.

Horizontal acceleration, vertical acceleration and orientation in every 40 milliseconds are recorded from the beginning of each experiment to the end. There are 50 to 150 actual steps in each trace under fixed modes and there are 200 to 400 actual steps in each trace under the variant mode. The external environment such as temperature and light intensity is non-constant.

We have conducted 136 experiments under six fixed modes and the variant mode. Then we mix these traces in different orders and generate 149 additional traces under different modes including fixed and variant modes. In total, we use 285 traces in our evaluation.

TABLE V
Minimum and average power consumption in different scenarios.

Status	Minimum power (mA)	Average power (mA)
Screen off	49.9	104.0
Screen on	135.1	197.0
Android native step counter	29.5	155.3
Self-adaptive WPD	31.7	198.8
Self-adaptive DTW	94.6	180.0
SCR	32.1	200.9

B. Performance Comparison

Table IV shows the comparison of six different algorithms including the native Android step counter, the original and adaptive WPD (aWPD) algorithms, the original and adaptive DTW (aDTW) algorithms, and the proposed SCR algorithm. The error rate is calculated as following. c_m and c_t represent the measurement values of steps and the actual number of steps, respectively.

$$error\ rate = \frac{|c_m - c_t|}{c_t} \times 100\%$$

Under all seven conditions, three self-adaptive algorithms work much better than the Android step counter. Compared with original non-adaptive algorithm, the self-adaptive algorithms are able to dynamically adapt to various conditions including different speed and different placement. Non-adaptive methods only adapt to a few conditions that conforms to optimal values.

C. Power Consumption

We measure the power consumption of our algorithms on Google Nexus 5 by measuring the working current using the Monsoon Power Monitor [25]. The scenarios include screen off, screen on, Android native step counter, self-adaptive WPD, self-adaptive DTW, and the SCR algorithm. Fig. 10 shows the variation of average power consumption every 0.1s when different operations are performed. Table V shows the minimum and average power consumption in different scenarios.

The working current is close to 50mA initially when the phone screen is off and the pedometer is not running. It jumps to 120mA when the screen is on. The result shows that our algorithms consume around 25∼45mA additional power compared to the Android native counter, which is generally acceptable as our algorithms perform much more accurately.

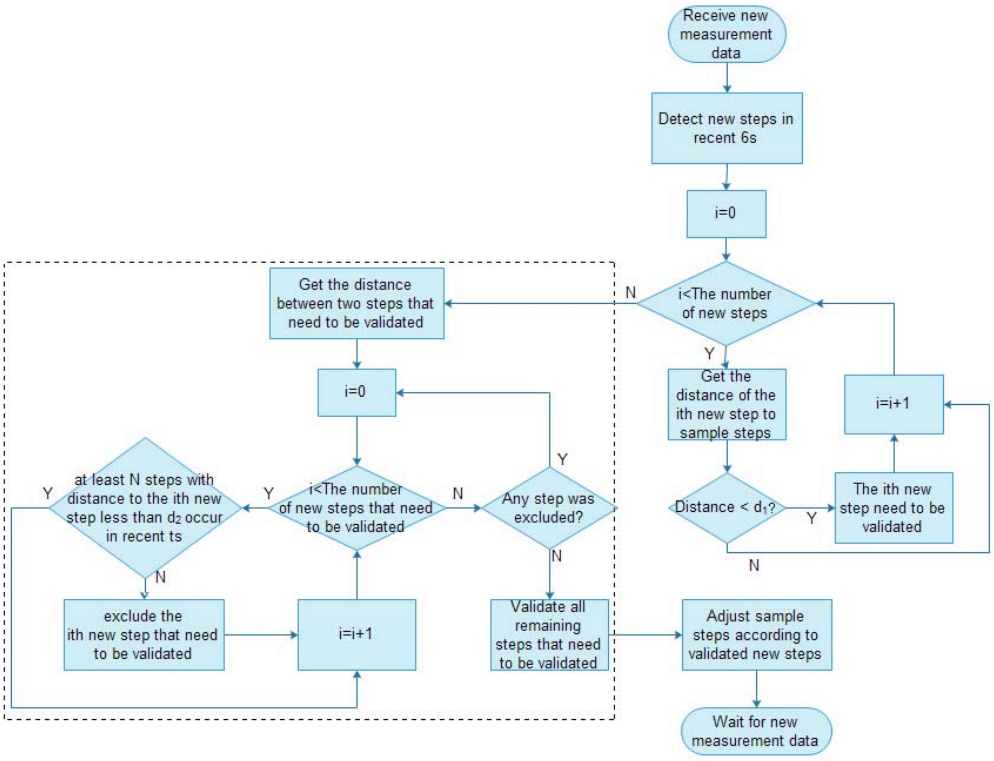

Fig. 9. Basic workflow of SCR. The procedure of the recognizing cycle is shown in the dotted box.

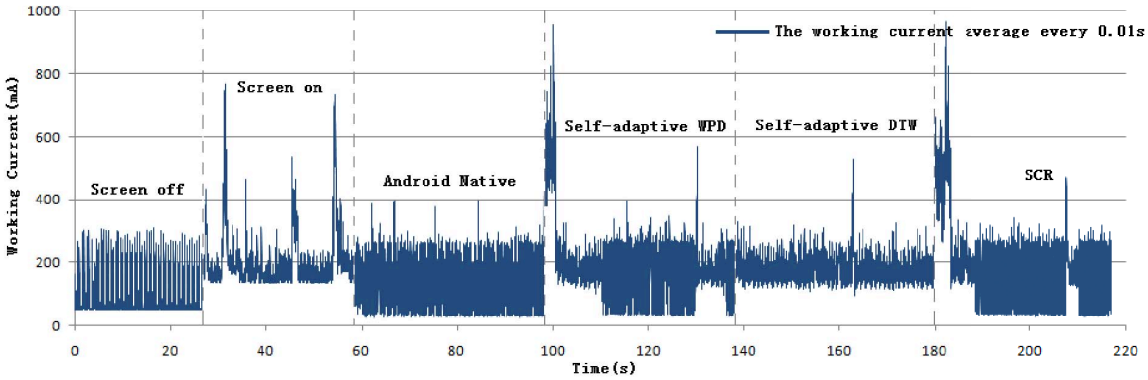

Fig. 10. Power Consumption in different scenarios.

VI. Discussions

In the first five cases with regular walking, we can see that the Android native step counter is not accurate enough in most cases, which indicates that it is not sufficient to just calling the APIs when doing step counting. In all cases, the adaptive versions perform significantly better than the non-adaptive versions using fixed values. However, since the sample steps or the optimal values of non-adaptive algorithms are in accord to walking, the error rates of non-adaptive algorithms are not too high. On average, the SCR algorithm achieves the best accuracy, but the self-adaptive WPD and self-adaptive DTW algorithms are very close and even better in several cases. Our results are similar to the original WPD algorithm implemented by Brajdic and Harle under the same 4 walking conditions [8].

In the running case, because the condition varies a lot compared to walking (Fig. 4), the results are less accurate for most algorithms, especially the original WPD algorithm. The SCR algorithm works better than the adaptive WPD and DTW algorithms while running because the trace data of each running step varies more rapidly than peak and frequency parameters, while we only need to check the peak value of horizontal acceleration without the whole trace.

In the last case when variant stepping mode is used, it includes a variety of combinations with different phone placements and walking modes. We can see that the self-adaptive algorithms adapt well to the changes, while the original algorithms are even worse than the Android native counter (Fig. 11). We find that when the stepping condition changes,

TABLE IV
TESTING RESULTS OF DIFFERENT ALGORITHMS UNDER DIFFERENT CONDITIONS. (*Note that aWPD and aDTW represent the self-adaptive versions of WPD and DTW, respectively.*)

Stepping mode	Smartphone placement	Quantity of data	Data type of error rate (%)	SCR	aWPD	WPD	aDTW	DTW	Android Native	Best Algorithm
Walking	Trousers front pocket	38	Average	1.40	3.02	6.18	3.04	7.74	4.96	SCR
			Maximum	3.44	17.80	28.32	9.04	21.25	27.40	
			Median	1.09	1.58	3.02	2.20	6.46	3.12	
			Standard deviation	0.98	3.34	6.87	2.43	4.96	5.77	
	Trousers back pocket	46	Average	2.41	2.64	4.46	3.32	9.75	5.71	SCR
			Maximum	6.30	8.49	28.90	7.84	19.74	21.34	
			Median	1.95	2.40	2.67	3.04	9.65	4.17	
			Standard deviation	1.75	1.98	5.60	1.86	4.16	4.68	
	Coat pocket	45	Average	2.45	1.55	8.10	2.61	7.07	3.88	aWPD
			Maximum	4.70	4.52	23.37	5.69	18.10	7.62	
			Median	2.65	1.47	8.07	2.39	7.25	4.14	
			Standard deviation	1.15	1.05	6.56	1.52	3.57	1.99	
	Hand Held	36	Average	2.87	3.31	7.46	2.90	10.63	5.51	SCR
			Maximum	6.97	7.05	26.92	6.77	44.81	9.17	
			Median	3.00	3.04	4.54	2.83	6.54	5.90	
			Standard deviation	1.31	1.37	6.81	1.51	10.73	2.21	
	Handbag	31	Average	2.07	2.09	1.89	1.86	1.99	8.67	aDTW
			Maximum	5.44	3.96	5.01	5.09	4.18	17.64	
			Median	2.04	2.02	1.99	1.86	1.90	9.62	
			Standard deviation	1.48	1.04	1.22	1.18	0.99	5.22	
Running	Trousers front pocket	38	Average	2.74	2.96	87.97	5.45	7.00	15.47	SCR
			Maximum	11.09	11.44	100.00	14.83	16.91	57.26	
			Median	1.73	1.80	87.35	4.71	7.59	8.75	
			Standard deviation	2.74	2.80	11.74	4.12	4.77	14.82	
Variant	Variant	57	Average	2.37	2.49	18.32	3.11	6.23	5.33	SCR
			Maximum	9.18	10.89	59.12	11.87	20.63	28.37	
			Median	1.84	1.90	9.30	2.71	5.57	4.64	
			Standard deviation	1.87	2.58	18.21	2.51	4.52	4.60	

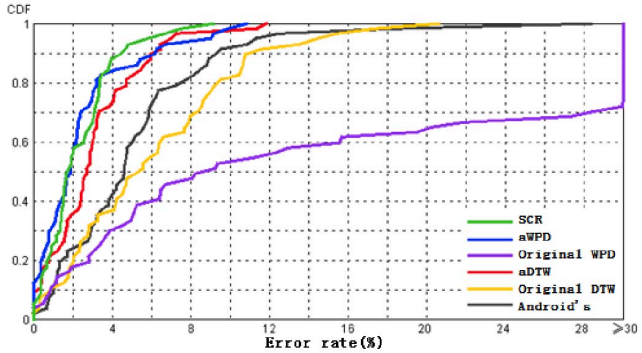

Fig. 11. CDFs of different algorithms under variant stepping condition.

the sample steps as well as the optimal values of self-adaptive methods are adjusted according to new condition. However, there are still inevitable measurement errors occurring at the time of condition change. The proposed SCR algorithm achieves the best accuracy and is also the most stable in this case.

VII. CONCLUSION

This paper introduces self-adaptive stepping to improve the accuracy of stepping under unrestricted walking and running modes on smartphone. We propose self-adaptive versions of two existing step counting methods, as well as a new SCR method that performs the best in our evaluations. Experimental results show that self-adaptive methods achieve much better results while introducing small power overhead.

Our future work includes improving the current algorithms further and investigating in how to automatically apply suitable algorithms under every unrestricted condition.

VIII. ACKNOWLEDGEMENT

This work is partly supported by the National Basic Research Program of China (973) under Grant No. 2015CB352201, the National Natural Science Foundation of China under Grant No.61421091.

REFERENCES

[1] A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, and P. Havinga, "Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey," in *Architecture of computing systems (ARCS), 2010 23rd international conference on*. VDE, 2010, pp. 1–10.

[2] A. Srivastava, J. Gummeson, M. Baker, and K.-H. Kim, "Step-by-step detection of personally collocated mobile devices," in *HotMobile '15*, 2015, pp. 93–98.

[3] U. Ryu, K. Ahn, E. Kim, M. Kim, B. Kim, S. Woo, and Y. Chang, "Adaptive step detection algorithm for wireless smart step counter," in *Information Science and Applications (ICISA), 2013 International Conference on*. IEEE, 2013, pp. 1–4.

[4] D. Dueker, W. James Gauderman, and R. McConnell, "Accuracy of a new time-resolved step counter in children," *Pediatric exercise science*, vol. 24, no. 4, p. 622, 2012.

[5] S. Haołyan, L. Wenłshuai, and Z. Yi-ming, "Using 3-axis accelerometer adxl330 to high accuracy pedometer," *Chinese Journal of Sensors and Actuators*, vol. 19, no. 4, 2006.

[6] M. Song-yan, "The design and implementation of a pedometer application based on ios platform," *SOFTWARE*, vol. 33, no. 12, pp. 66–68, 2013.

[7] "Google Play," http://play.google.com.

[8] A. Brajdic and R. Harle, "Walk detection and step counting on unconstrained smartphones," in *UBICOMP*, 2013, pp. 225–234.

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

[10] S. Jayalath, N. Abhayasinghe, and I. Murray, "A gyroscope based accurate pedometer algorithm," in *International Conference on Indoor Positioning and Indoor Navigation*, vol. 28, 2013, p. 31st.

[11] R. K. Ibrahim, E. Ambikairajah, B. G. Celler, and N. H. Lovell, "Time-frequency based features for classification of walking patterns," in *Digital Signal Processing, 2007 15th International Conference on*. IEEE, 2007, pp. 187–190.

[12] E. Raffin, S. Bonnet, and P. Giraux, "Concurrent validation of a magnetometer-based step counter in various walking surfaces," *Gait & posture*, vol. 35, no. 1, pp. 18–22, 2012.

[13] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10, no. 16. Seattle, WA, 1994, pp. 359–370.

[14] I. Apostolopoulous, D. S. Coming, and E. Folmer, "Accuracy of pedometry using a head-mounted display," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2015, pp. 2153–2156.

[15] "Android developer sensor event," "http://developer.android.com/reference/android/hardware/SensorEvent.html".

[16] C. Tudor-Locke and L. Lutes, "Why do pedometers work?" *Sports Medicine*, vol. 39, no. 12, pp. 981–993, 2009.

[17] E. Fortune, V. Lugade, M. Morrow, and K. Kaufman, "Validity of using tri-axial accelerometers to measure human movement–part ii: Step counts at a wide range of gait velocities," *Medical engineering & physics*, vol. 36, no. 6, pp. 659–669, 2014.

[18] Y. Li, Y. Liu, T. Yuan, and W. Wang, "Multiple classifier based walking pattern recognizing algorithm using acceleration signals," *Acta Electronica Sinica*, vol. 8, pp. 1794–1798, 2009.

[19] X. Yang, "Human motion patterns recognition based on single triaxial accelerometer," *South China University of Technology, Guang Zhou*, 2011.

[20] Y. J. Lee, "Detection of movement and shake information using android sensor," *Advanced Science and Technology Letters*, vol. 90, pp. 52–56, 2015.

[21] "Android sensor service source code," "https://android.googlesource.com/platform/frameworks/native/+/master/services/sensorservice".

[22] P. Zhou, M. Li, and G. Shen, "Use it free: Instantly knowing your phone attitude," in *MobiCom*, 2014, pp. 605–616.

[23] S. Shin, C. Park, J. Kim, H. Hong, and J. Lee, "Adaptive step length estimation algorithm using low-cost mems inertial sensors," in *Sensors Applications Symposium, 2007. SAS'07. IEEE*. IEEE, 2007, pp. 1–5.

[24] Y. Zhen-Jie, Z. Zhi-Peng, and X. Li-Qun, "An effective algorithm to detect abnormal step counting based on one-class svm," in *Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on*. IEEE, 2014, pp. 964–969.

[25] "Monsoon Power Monitor," https://www.msoon.com/LabEquipment/PowerMonitor/.