A Novel System Architecture for Real-time, Robust and Accurate Step Detection for PDR based Indoor Localization

M. P. R. S. Kiran*, P. Rajalakshmi*, Mukesh Kumar Giluka†, Bheemarjuna Reddy Tamma†
*WiNet Laboratory, Dept. of Electrical Engineering
†Dept. of Computer Science & Engineering
Indian Institute of Technlogy Hyderabad, Hyderabad, India
Email: ee12m1021, raji, cs11p1002, tbr@iith.ac.in

***Abstract*—Indoor localization is currently of good interest concerning both business viability and end user experience. In this paper, we propose an accurate and robust step detection algorithm for smartphone based pedestrian dead reckoning (PDR) systems. The developed algorithm makes use of the acceleration measured from the smartphone and uses a statistical threshold based classification to detect the steps accurately. The statistical thresholds used are derived from extensive field trials with subjects of different age groups and found to provide good accuracy when used in real-time. For analyzing the performance of the proposed algorithm, we have implemented the algorithm on Android platform and performed extensive field trials. The analysis proves that the proposed algorithm identifies the user steps in real-time with an accuracy of more than 99% with minimum memory requirements.**

***Index Terms*—PDR, SLAM, Step detection, Indoor localization.**

I. Introduction

Indoor localization has a multitude of applications such as indoor mapping and navigation in commercial buildings, location based services, rescue during hazards, etc., [1]. Although many techniques to provide indoor localization have been investigated in the literature, no global solution is still available and is currently an active research area. Multiple problems such as lack of accuracy, huge manual intervention, need for pre-deployed stable infrastructure, aggregation of errors over the time still make it a challenging problem [2], [3]. The accuracies of traditional GPS based positioning systems are highly confined due to the indoor structures and are completely useless for indoor positioning [4]. Later, WiFi based indoor localization techniques have gained prominence due to the existence of already deployed WiFi infrastructure in many of the indoor spaces such as homes, commercial complexes, etc., [5], [6]. Although WiFi based localization techniques provide accuracies up to 3m, main drawbacks involve increased power consumption and creation of huge fingerprinting database [7]. Also, the indoor environmental characteristics of many commercial spaces change rapidly owing to human mobility and changes in asset positioning such as furniture, tables, etc. These result in accuracy degradation of WiFi positioning systems and requires aggregation of new fingerprint database every time the environment changes. Hence, WiFi based indoor localization techniques have not witnessed many real-time deployments. Similarly, other studies existing in literature investigated the usage of Bluetooth and RFID for localization in indoor spaces [8]–[11]. In all these studies, for achieving localization new infrastructure needs to be deployed while the problems like fingerprinting and accurate estimation of path loss models persist.

However, with the increased usage of smartphones, the pedestrian dead reckoning (PDR) based indoor localization techniques are gaining importance. Using PDR with smartphones for localization has significant advantages such as wide availability of necessary infrastructure with the end users, low cost, no need for new infrastructure deployment and low power consumption. In the PDR technologies, the inertial sensors present on the smartphones can be utilized to track or detect the user activities such as walking, running, moving in elevators etc [12], [13]. In all the PDR techniques, the first step involves the accurate detection of steps and then a proper stride length estimation model is used to calculate the distance traveled by the user. Accurate calculation of the step count, distance traveled by the user and heading direction play a significant role in determining the user trajectory accurately. Multiple techniques are investigated in the literature for step detection using smartphones [14]–[19]. In [14], authors use the method proposed in [20] for detection of steps where the magnitude of acceleration is considered for formulating a Finite State Machine (FSM). In the analysis, the authors considered a stream of continuous walking data. Usually in a realistic scenario, the users when navigating with the help of an indoor map, will have a quasi-continuous movement and the algorithm needs to be robust enough to detect the steps when the user movement is discontinuous. In [15], authors developed a statistical thresholding based step detection algorithm which estimates the total steps taken by the user accurately when the user motion is continuous, but fails when discontinuities occur. Authors in [16], proposed a 3 step calculation of steps which include identifying the state of user with a decision tree classifier, exploiting features such as periodicity and estimating number of peaks in the accelerometer readings. Although the

computational capabilities of smartphones are increasing, in order to perform classification and feature extraction in real-time is difficult, also leads to significant power consumption. Also, for determining the periodicity, the system introduces a notable lag which disrupts the user experience. In [17], authors proposed an FSM to determine the current orientation of the smartphone and use the appropriate axis for determining the steps. While determining the steps, authors considered a single threshold and time interval for detection of steps. The problem with single threshold involves identification of false peaks as user steps leading to error accumulation. In addition, if the orientation change of mobile phone is not detected accurately, the FSM results in an erroneous state resulting in a non-recoverable error. In [18], authors used a slightly low complex approach where the Local Coordinate System (LCS) is projected to Global Coordinate System (GCS) and used a simple thresholding for detection of steps. For the analysis of performance, authors did not consider any scenarios where false steps occur. Authors in [19] used Continuous Wavelet Transform (CWT) and simple peak thresholding for detection of steps. CWT introduces complexity in implementation and this technique is also prone to error accumulation when false peaks are generated.

Hence, there is a need for a low-complex real-time step detection algorithm that is robust enough to false peaks and discontinuous user movement. Primary contributions of this paper include:

- Develop a low complex step detection algorithm which can 1) accurately detect the steps in the presence of false peaks generated due to mishandling or slipping of phone, 2) accurately function in both continuous and discontinuous user motion
- Validation of proposed algorithm using real-time implementation and field trials
- Comparison of proposed method with existing implementations

As this study primarily aims at developing a robust step detection algorithm for indoor navigation applications, we assume, the user is handling the phone in front of him. This hand-held position is reasonable and also in good agreement with the literature as the user to check his position continuously views his screen [18]. The remainder of this paper is organized as follows. Section II describes the proposed architecture for detection of steps. Section III describes the experimental setup, and analyzes the performance of the proposed model in various scenarios. Finally, Section IV concludes the paper by discussing the future scope of this work.

II. Holistic View of the System Architecture

In this section, we discuss the proposed system architecture for robust, accurate and real-time step detection shown in Fig. 1. The proposed architecture consists of seven functional units, and are briefly discussed in the following sections.

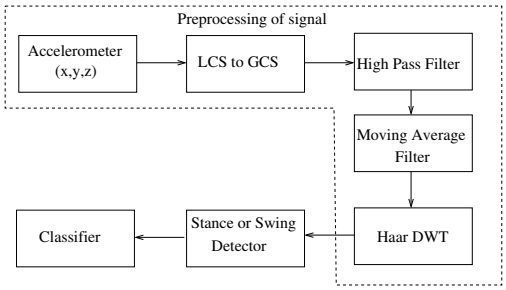


Fig. 1: Proposed system architecture for robust and low complex step detection

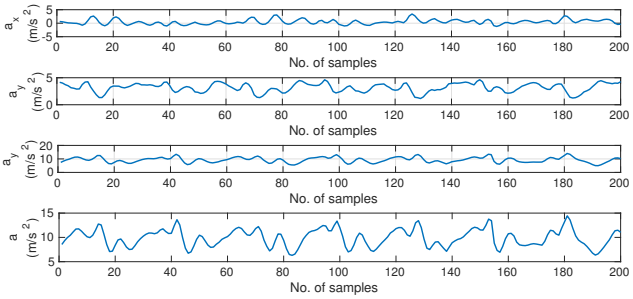


Fig. 2: a_x, a_y, a_z and a obtained while the user takes seven steps with the smartphone held in his hand

A. Accelerometer

An accelerometer is a low-cost inertial measurement unit (IMU) device which measures the forces applied to the smartphone in the three axes including the effect of gravity. It is widely available in many of the smartphones due to its low cost and low power consumption. Usually, the human walking always follows a distinctive pattern, where the forces applied will be symmetrical on both the legs. These applied forces create an acceleration during the movement of legs which get coupled into the human body. As the user holds the phone in his hand, these applied forces will also result in the acceleration of the phone and these recorded accelerations using a smartphone help us in the real-time identification of the steps taken by the user. Although the accelerometer records the acceleration generated due to user walking, it also includes the acceleration generated due to other forces such as random phone movement while holding in hand which we term it as noise. Fig. 2 shows the raw acceleration data along x-axis (a_x), y-axis (a_y), z-axis (a_z) and norm (a) consisting of 7 steps obtained from the smartphone while the user is walking with the smartphone held in his hand. Although one can observe the peaks and valleys corresponding to the user steps, it is still noisy and needs proper filtering to acquire the signal of interest for accurate step detection.

The dynamics of walking can be classified into a stance and a swing phase. For every step taken, the entire duration for which the heel is in the air is regarded as swing phase, and the moment the heel strikes the ground is considered as stance phase. During the swing phase, the acceleration

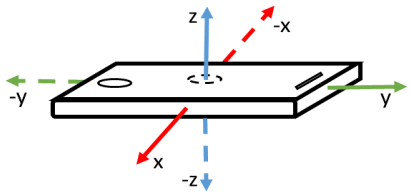


Fig. 3: Orientation of accelerometer axis within a smartphone

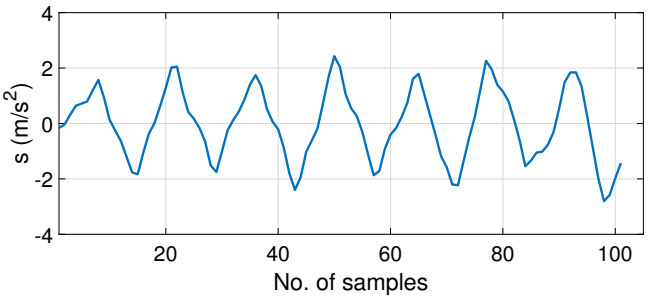


Fig. 4: Filtered and smoothened accelerations in the vertical axis with reference to the ground comprising of seven user steps

increases initially and then decreases before the heel strikes the ground thereby resulting in a valley when the heel strikes the ground.

B. Local Co-ordinate System (LCS) to Global Co-ordinate System Conversion (GCS)

The acceleration obtained due to the footsteps can be well captured in the vertical axis to the ground. Fig. 3 shows the orientation of the three axes within a smartphone. Ideally, if the smartphone is held perfectly parallel to the ground, the effect of the gravity will be observed only in the a_z. In Fig. 2, as the user is holding the phone in hand, the effect of the gravity is maximally observed in a_z compared to other two axes. While the user is walking, the phone orientation changes and z-axis of the smartphone will no longer be the vertical axis, leading to the distribution of gravity into all the three axes. Hence we perform a rotation of 3D acceleration signals obtained from local coordinate system (LCS) to global coordinate system (GCS) using the rotation matrix $\mathbf{R}$. $\boldsymbol{a^G}$, which represents the accelerations after transforming into GCS can be calculated as shown below.

$$\boldsymbol{a^G} = \boldsymbol{Ra} \tag{1}$$

C. High-pass Filter (HPF)

The data acquired from the accelerometer of the smartphone comprises of the effect of gravity. Hence, we use a low complex high-pass filter to remove the gravity from the GCS rotated accelerations as shown below.

$$\boldsymbol{g_i} = \alpha \boldsymbol{g_{i-1}} - (1-\alpha)\boldsymbol{a_i^G} \tag{2}$$

$$\boldsymbol{a_{hpf}^G} = \boldsymbol{a_i^G} - \boldsymbol{g_i} \tag{3}$$

Here, $\boldsymbol{g_i}$ represents the vector of 3-dimensional gravity extracted from the accelerometer data at the time instant *i*, and $\boldsymbol{a_{hpf}^G}$ represents the vector of 3-dimensional high-pass filtered data without the effects of gravity.

D. Haar Discrete Wavelet Transform

In general, the human walking pattern comprises of frequencies varying from 1.2 to 2.5 Hz and all the other high frequencies may result in false peaks. Hence, we make use of discrete wavelet transform (DWT) to perform the time frequency analysis and remove the high-frequency noise. Although the processing capabilities of the smartphones are increasing, power consumption is still a major challenge. The main significance in choosing Haar as the mother wavelet lies in its low-complex decomposition. The vertical axis of the filtered and GCS rotated acceleration ($a^G_{hpf,z}$) is decomposed in a single dyadic scale (2^1) and the approximate coefficients are used for the detection of steps.

E. Moving Average Filter

Although the noises are removed, to smoothen the acceleration signal, we make use of a moving average filter (MAF). Fig. 4 plots the smoothened acceleration along the vertical axis (s) with reference to the ground after all the HPF, Haar DWT and MAF are performed.

F. Stance or Swing Detector with Classifier

We make use of the smoothened signal (s) for estimating the peaks and valleys. As soon as the new data is available from the accelerometer, the preprocessing of the signal is performed, and the smoothened sample (s_n) is obtained. To overcome the limitations of existing studies where the probability of detection of false steps is higher, we classify a step in three phases. We will first detect a peak and then detect if any valley exists for the detected peak. After detection of the peak and valley, we then use a binary classifier to classify it as a true step. The sample s_n can be classified as a peak or valley if it satisfies the below condition where w indicates the window length.

$$s_n = \begin{cases} peak, & \text{if } s_n > max(s_{n-w:n-1}, s_{n+1,n+w}) \\ valley, & \text{if } s_n < min(s_{n-w:n-1}, s_{n+1,n+w}) \end{cases} \tag{4}$$

After the detection of a valid peak or valley, Algorithm 1 and 2 provide the mechanism for detection of a valid step. The Algorithm 1 is used for the initialization of variables. a_p and a_v indicate the amplitude of the peak and valley detected respectively and are initialized to 0 at the beginning of the process. n_p and n_v indicate the time stamp of the latest peak and valley identified. The total number of steps detected ($steps$) is initialized to 0, and w indicates the window length considered for detection of peak and valley. The variable $prevState$ indicates the previous state detected which includes $peak$ or $valley$ and is initialized to $init$ at the beginning of the process.

As the human walking pattern follows periodic dynamics, the maximum latency between a peak and valley within a step will be constrained by a maximum delay specified by $maxDelay$.

Algorithm 1 Parameter Initialization

```
function initialize()
    a_p ← 0
    a_v ← 0
    n_p ← 0
    n_v ← 0
    steps ← 0
    w ← 5
    prevState ← init
    maxDelay ← f_s / 2
end function
```

After the initialization, every preprocessed acceleration sample at time instance n (s_n) is classified for identification of steps as shown in Algorithm 2. The entire process can be classified into five scenarios depending on the state of the process as follows

1) Scenario 1 - s_n is identified as a peak during initialization: During the process initialization the $prevState$ is equal to $init$, and hence we update the peak parameters a_p, n_p to s_n and n respectively. Also, we update $prevSate$ to $peak$ as a valid peak is detected.

2) Scenario 2 - s_n is identified as a peak and $prevState$ is also a peak: If s_n is detected as a peak while the $prevState$ is also a $peak$ signifies that no valid $valley$ is present in between the two peaks (current peak at time instant n and previous peak detected at n_p). Hence, we check if the amplitude of the current peak is greater than the previous peak and classify the previous peak detected as a false peak. Also, we update the peak parameters a_p, n_p to s_n and n respectively. If the current peak amplitude is found to be smaller than the previous peak, we consider the current peak as a false peak.

3) Scenario 3 - s_n is identified as a peak and $prevState$ is a valley: If s_n is detected as a peak while the $prevState$ is a $valley$, we consider the current peak as a new peak and update the peak parameters a_p, n_p to s_n and n respectively.

4) Scenario 4 - s_n is identified as a valley and $prevState$ is also a valley: If s_n is detected as a valley while the $prevState$ is also a $valley$. it signifies that no valid $peak$ is present in between the two valleys. Hence, we check if the amplitude of the current valley is lesser than the previous valley and classify the previous valley as a false valley. Also, we update the valley parameters a_v, n_v to s_n and n respectively. If the current valley amplitude is found to be greater than the previous valley, we consider the current valley as a false valley.

5) Scenario 5 - s_n is identified as a valley and $prevState$ is a peak: If s_n is detected as a valley while the $prevState$ is a peak, we consider the current valley as a new peak and update the peak parameters a_v, n_v to s_n and n respectively. Now, we measure the delay between the corresponding peak and valley ($n_p - n_v$), and if it does not exceed the $maxDelay$, it can be classified as a true step and the $steps$ can be incremented. If the delay between the corresponding peak and valley exceeds the $maxDelay$, it can be regarded as a false step.

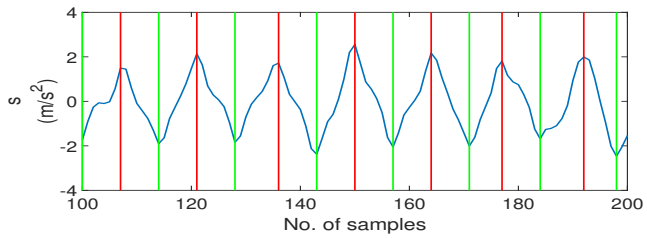


Fig. 5: Detection of steps when the user walking is continuous

Algorithm 2 Stance or Swing Detection

```
function stepDetector(s_n)
    if s_n = peak then
        if prevState = init then
            a_p = s_n
            n_p = n
            prevState = peak
        else if prevState = peak then
            if s_n > a_p then
                a_p = s_n
                n_p = n
                prevState = peak
            else if prevState = valley then
                a_p = s_n
                n_p = n
                prevState = peak
            end if
        end if
    else if s_n = valley then
        if prevState = valley then
            if s_n < a_v then
                a_v = s_n
                n_v = n
                prevState = valley
            else if prevState = peak then
                a_v = s_n
                n_v = n
                prevState = valley
                if n_v - n_p < maxDelay then
                    steps = steps + 1
                end if
            end if
        end if
    end if
end function
```

Fig. 5 and 6 plots the detected steps using the proposed architecture in two different scenarios where the user motion is continuous and discontinuous respectively. The red marker indicates the swing phase of the user and the green marker indicates the stance phase of the user and also a successful step was taken by the user. It can be observed that the proposed algorithm can detect the user steps accurately in both the cases.

III. Performance Analysis

For analyzing the performance of the proposed algorithm, we have developed a real-time testbed using Android platform. The developed application runs on a smartphone or tablet and counts the number of steps taken by the user using the proposed algorithm. Fig. 7a and 7b shows the developed application and the user experimenting using a smartphone respectively. For the experimentation, we have considered 12 subjects with varying demographics and each user is asked

TABLE I: Performance analysis of the proposed system architecture in real-time using a smartphone under two different scenarios with 12 demographically different subjects

User No.	Sex	Age	Height (cm)	T1 - Ground Truth	T1 - Proposed Method	T1 - Accuracy (%)	T2 - Ground Truth	T2 - Proposed Method	T2 - Accuracy (%)
1	F	26	173	274	273	99.63	271	270	99.63
2	M	30	170	236	238	99.15	236	237	99.57
3	M	33	178	204	204	100	214	213	99.53
4	M	45	180	245	245	100	222	222	100
5	M	24	175	245	245	100	244	245	99.59
6	M	29	165	251	250	99.60	248	247	99.59677
7	M	35	175	240	240	100	237	237	100
8	M	27	168	236	236	100	243	242	99.58
9	M	41	178	231	230	99.56	230	230	100
10	M	25	165	234	234	100	226	226	100
11	F	30	163	260	260	100	272	271	99.63
12	F	25	152	216	216	100	283	282	99.64

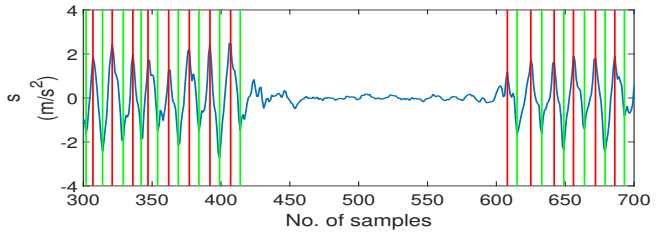


Fig. 6: Detection of the steps when the user walking is discontinuous

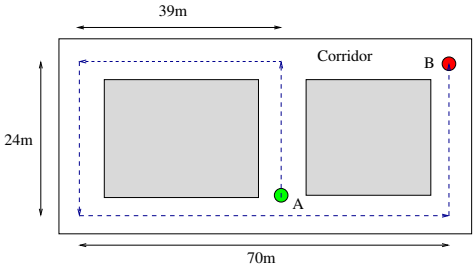
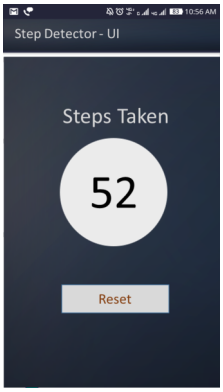


Fig. 8: Path trajectory adhered for experimental analysis



(a)

(b)

Fig. 7: (a) Developed Android application for analyzing the real-time performance (b) Experimental trials at Academic Block A, IIT Hyderabad

to walk across the hallway of the Academic Block A, IIT Hyderabad. The path trajectory adhered by every user during the experimentation is shown in Fig. 8 with A indicating the starting point and the ending point is represented using B. On an average, the user walks 176m for every trial considered for the real-time performance analysis.

We have considered two scenarios for the user walking along the defined trajectory namely continuous motion (T1) and discontinuous motion (T2). In the continuous motion scenario, the user walks continuously from start to end of the trajectory, while in the latter scenario the user stops walking for irregular intervals randomly. Table I describes the accuracy of the proposed system architecture under the two scenarios (T1 and T2). The column *T1 - Ground Truth* indicates the actual user steps taken to reach from start to the end of the trajectory with user motion being continuous and the column *T1 - Proposed Method* provides the total number of steps calculated by the smartphone in real-time. Similarly, the columns *T2 - Ground Truth* and *T2-Proposed Method* provides the actual steps taken and the steps detected by the smartphone under scenario T2. One can observe that the proposed method can accurately determine the total number of steps taken by the user accurately by achieving a negligible error. On an average over 12 subjects with varying demographics, the proposed architecture achieves an accuracy of 99.82% and 99.73% in both the scenarios T1 and T2 respectively.

Table II compares the performance of the proposed model with popular existing models. It can be observed that the proposed method achieves better accuracy compared to the existing models. Although the authors in [17] claim the accuracy of the step detection to be 100%, the analysis is performed under a constrained environment where the artifacts created due to random hand movements, false steps, etc. are not considered. Regarding the complexity of the proposed system architecture, as given in Table II, most of the techniques offer good accuracy using complex preprocessing techniques such as continuous wavelet transforms, Fourier transform and periodicity estimation, etc. Although these techniques improve the estimation accuracy, it significantly increases the power consumption of the smartphone and also introduces significant latency. The proposed architecture in this paper makes use of very low complex signal conditioning blocks such as Haar DWT, low complex HPF, etc. which are recursive and a new sample can be processed immediately as soon as it arrives

TABLE II: Performance analysis of the proposed architecture when compared to existing popular models

No.	Model	Accuracy (%)	Drawback
1	[15]	99.5	Results in inaccurate step detection if the user movement is discontinuous
2	[16]	99	Involves complex techniques such as Fourier, periodicity and similarity estimation
3	[17]	100	Introduces false steps due to random hand movements
4	[18]	99	Introduces false steps due to random hand movements
5	[19]	99	Uses complex Continuous Wavelet Transform
6	Proposed Method	99.73	Robust to false steps, but smartphone orientation dependent

thereby avoiding additional latency. From the experimental analysis, it is observed that the proposed system architecture detects the steps accurately in real-time without any notable latency. Hence, from the performance analysis, we are convinced that the proposed system architecture can aid in developing low complex, robust and real-time indoor positioning systems.

IV. CONCLUSION

In this paper, we proposed a novel low-complex, robust and real-time system architecture for step detection using a smartphone. The developed system architecture uses low-complex signal preprocessing techniques for noise removal, and the developed peak or valley estimator can accurately estimate the steps by avoiding the false steps or artifacts generated due to involuntary hand movements. Performance of the proposed system architecture is analyzed by developing an Android application. Twelve subjects with varying demographics are considered for real-time experimentation in the hallway of Academic Block A, IIT Hyderabad. Two scenarios which include both continuous user walking (T1) and discontinuous user walking (T2) are considered for the performance analysis, and in each of the scenario, the user walks an average of 176m. Performance analysis shows that the proposed system architecture achieves an accuracy of 99.82% and 99.73% in both the scenarios T1 and T2 respectively. Hence, we are convinced that the proposed system architecture can significantly aid in the development of a low complex an real-time indoor positioning system. The future scope of this work is to develop an accurate, robust, low-complex and real-time indoor positioning system.

ACKNOWLEDGMENT

This work is supported by Visvesvaraya PhD Scheme, Ministry of Electronics and Information Technology (MEITY, Govt. of India) and Indian Institute of Technology Hyderabad, India.

REFERENCES

[1] Z. Chen, Q. Zhu and Y. C. Soh, "Smartphone Inertial Sensor-Based Indoor Localization and Tracking With iBeacon Corrections," in *IEEE Transactions on Industrial Informatics*, vol. 12, no. 4, pp. 1540-1549, Aug. 2016. doi: 10.1109/TII.2016.2579265

[2] H. Xie, T. Gu, X. Tao, H. Ye and J. Lu, "A Reliability-Augmented Particle Filter for Magnetic Fingerprinting Based Indoor Localization on Smartphone," in *IEEE Transactions on Mobile Computing*, vol. 15, no. 8, pp. 1877-1892, Aug. 1 2016. doi: 10.1109/TMC.2015.2480064

[3] Z. Yin, C. Wu, Z. Yang and Y. Liu, "Peer-to-Peer Indoor Navigation Using Smartphones," in *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1141-1153, May 2017. doi: 10.1109/JSAC.2017.2680844

[4] N. T. Thuong, H. T. Phong, D. T. Do, P. Van Hieu and D. T. Loc, "Android application for WiFi based indoor position: System design and performance analysis," in *2016 International Conference on Information Networking (ICOIN)*, Kota Kinabalu, 2016, pp. 416-419. doi: 10.1109/ICOIN.2016.7427147

[5] S. He and S. H. G. Chan, "Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons," in *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 466-490, Firstquarter 2016. doi: 10.1109/COMST.2015.2464084

[6] Q. D. Vo and P. De, "A Survey of Fingerprint-Based Outdoor Localization," in *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 491-506, Firstquarter 2016. doi: 10.1109/COMST.2015.2448632

[7] Yuqi Li, Zhe He and J. Nielsen, "Enhancing Wi-Fi based indoor Pedestrian Dead Reckoning with security cameras," in *2016 Fourth International Conference on Ubiquitous Positioning, Indoor Navigation and Location Based Services (UPINLBS)*, Shanghai, 2016, pp. 107-112. doi: 10.1109/UPINLBS.2016.7809958

[8] J. Neburka et al., "Study of the performance of RSSI based Bluetooth Smart indoor positioning," in *2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA)*, Kosice, 2016, pp. 121-125. doi: 10.1109/RADIOELEK.2016.7477344

[9] S. A. Cheraghi, V. Namboodiri and L. Walker, "GuideBeacon: Beacon-based indoor wayfinding for the blind, visually impaired, and disoriented," in *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Kona, HI, 2017, pp. 121-130.

[10] Y. Zhao, K. Liu, Y. Ma, Z. Gao, Y. Zang and J. Teng, "Similarity Analysis-Based Indoor Localization Algorithm With Backscatter Information of Passive UHF RFID Tags," in *IEEE Sensors Journal*, vol. 17, no. 1, pp. 185-193, Jan.1, 1 2017. doi: 10.1109/JSEN.2016.2624314

[11] O. Friedewald and M. Lange, "Localization methods by using phase of arrival in systems with passive RFID Tag's," in *Smart SysTech 2016; European Conference on Smart Objects, Systems and Technologies*, Duisburg, Germany, 2016, pp. 1-6.

[12] T. R. Bennett, J. Wu, N. Kehtarnavaz and R. Jafari, "Inertial Measurement Unit-Based Wearable Computers for Assisted Living Applications: A signal processing perspective," in *IEEE Signal Processing Magazine*, vol. 33, no. 2, pp. 28-35, March 2016. doi: 10.1109/MSP.2015.2499314

[13] J. Windau and L. Itti, "Walking compass with head-mounted IMU sensor," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, 2016, pp. 5542-5547. doi: 10.1109/ICRA.2016.7487770

[14] H. Abdelnasser et al., "SemanticSLAM: Using Environment Landmarks for Unsupervised Indoor Localization," in *IEEE Transactions on Mobile Computing,* vol. 15, no. 7, pp. 1770-1782, July 1 2016. doi: 10.1109/TMC.2015.2478451

[15] H.-H. Lee, S. Choi, and M.-J. Lee, Step detection robust against the dynamics of smartphones, *Sensors,* vol. 15, no. 10, pp. 2723027250, 2015

[16] F. Gu, K. Khoshelham, J. Shang, F. Yu and Z. Wei, "Robust and Accurate Smartphone-Based Step Counting for Indoor Localization," in *IEEE Sensors Journal,* vol. 17, no. 11, pp. 3453-3460, June1, 1 2017. doi: 10.1109/JSEN.2017.2685999

[17] Q. Tian, Z. Salcic, K. I. K. Wang and Y. Pan, "A Multi-Mode Dead Reckoning System for Pedestrian Tracking Using Smartphones," in *IEEE Sensors Journal*, vol. 16, no. 7, pp. 2079-2093, April1, 2016. doi: 10.1109/JSEN.2015.2510364

[18] W. Kang and Y. Han, "SmartPDR: Smartphone-Based Pedestrian Dead Reckoning for Indoor Localization," in *IEEE Sensors Journal*, vol. 15, no. 5, pp. 2906-2916, May 2015. doi: 10.1109/JSEN.2014.2382568

[19] M. A. Chattha and I. H. Naqvi, "PiLoT: A Precise IMU Based Localization Technique for Smart Phone Users," in *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, Montreal, QC, 2016, pp. 1-5. doi: 10.1109/VTCFall.2016.7881166

[20] M. Alzantot and M. Youssef, "UPTIME: Ubiquitous pedestrian tracking using mobile phones," in *2012 IEEE Wireless Communications and Networking Conference (WCNC)*, Shanghai, 2012, pp. 3204-3209. doi: 10.1109/WCNC.2012.6214359