Featured Trajectory Generation Module for TrackPuzzle

Wanting Li Student Member, IEEE, Yongcai Wang Member, IEEE, Xiongfei Geng, Deying Li Member, IEEE

Abstract—Indoor route graph learning is critically important for autonomous indoor navigation. A key problem for crowd-sourcing indoor route graph learning is featured trajectory generation. In this paper, a system is provided to generate featured trajectories by crowd-sourcing smartphone data. Firstly, we propose a more accurate PDR algorithm for the generation of trajectory motion data. This algorithm uses ADAPTIV as the step counting method and uses the step estimation algorithm to make the trajectory more accurate in length. Next, the barometer is used to segment the tracks of different floors, and the track floors are obtained by WiFi feature clustering. Finally, by finding the turning point as the feature point of the trajectory, the vertices and edges of the trajectory are extracted to reduce the noise of the long straight trajectory.

Index Terms—Crowd-sourcing, PDR Algorithm, Floor Classification, Turning points, Trajectory segment

I. INTRODUCTION

Global Positioning System (GPS) provides satisfied outdoor positioning services but does not work indoor. WiFi is currently the dominant local wireless network standard for short-range communication and is the leading technology for indoor positioning. WiFi fingerprinting based on Received Signal Strength Indicator (RSSI) has been widely used in commercial indoor positioning systems because of its high positioning accuracy and ubiquitousness, such as Google and Apple. Traditional fingerprinting systems require a radio map comprised of a large quantity of WiFi fingerprints and each WiFi fingerprint contains a WiFi RSSI list and its ground truth location label. To build up radio maps, professionals consume huge amount of time to conduct a site survey for the ground truth location labels. To avoid the intensive training efforts in fingerprinting, crowdsourcing approaches have become increasingly interesting for researchers. And then, we use crowdsourcing smartphone data to build indoor map.

In this work, we propose a novel system for automatic construction of featured crowdsourcing trajectory generation, which requires neither prior knowledge of a floor plan nor manual initializations of PDR traces. Our contributions are summarized as follows.

Firstly, adaptive jerk pace step detection algorithm is exploited for step detection; hierarchical support vector machine method is exploited for displacement estimation; and magnetometer aided heading estimation is exploited in PDR. They work together to infer the relative motions more accurately from the raw IMU data sequences.

Then, a floor classification algorithm based on jointly DBSCAN clustering of barometer readings of PDR trajectories and WiFi MAC address clustering is proposed. It can divide the trajectories into correct floors rather reliably.

Finally, reliable turning point detection method is proposed to detect reliable turnings as the motion feature points. Small turnings are filtered out and frequent turning segments are deleted to exclude the impacts of irregular motions. So each PDR trajectory is converted to a chain-type graph, which uses the end points and the extracted turning points as vertices and uses the motion vectors between vertices as edge.

II. SYSTEM OVERVIEW

Trackpuzzle design is based on a participatory sensing approach, where phone sensors measurements collected from users moving naturally inside the building are processed to generate featured motion traces which in turn can provide an adequate description of the indoor route graph. In this paper, we only provide the method of featured trajectory generation, and the part of indoor route graph generation is discussed in another paper. The system consists of two main modules: (a) the Data Collection Module is responsible for collecting measurements from users’ devices, and (b) the Featured Trajectory Generation Module is responsible for building accurate motion traces based on Motion Trajectory Generation, Floor Classification and Vertex & Edge Extraction.

The data collection module collects measurements from phone embedded sensors including: accelerometer, magnetometer, gyroscope, barometer and the received cellular and WiFi signal strengths from the available cell towers and WiFi access points. In this system, we use Getsensordata app to collect these data as shown in Figure 1 and then generate files and send them back to the server for processing.

III. FEATURED TRAJECTORY GENERATION MODULE

In this section, we will firstly introduce (1) how a PDR algorithm is carefully designed to recover the motion trajectories as accurate as possible; (2) how the motion trajectories are segmented into different floors; and (3) how the motion trajectories are filtered and processed to extract reliable turning points as the feature points, so that each motion trajectory is converted to a chain-type graph using the feature points as vertices and motion vectors between feature points as edges.

A. Motion Trajectory Generation

The first step is to recover the motion trajectory from the raw IMU data, which relies on a dead-reckoning based
The user’s current location \( (X_k, Y_k) \) is estimated from the previous location \( (X_{k-1}, Y_{k-1}) \), the estimated step length is denoted by \( S \), and the estimated motion direction is denoted by \( \theta \). Then:

\[
\begin{align*}
X_k &= X_{k-1} + S \cos(\theta) \\
Y_k &= Y_{k-1} + S \sin(\theta)
\end{align*}
\]  

(1)

where the displacement \( S \) is mainly obtained from the accelerometer data, while \( \theta \) is estimated from the magnetometer and the gyroscope data. The PDR algorithm has been extensively studied in the literature. We conduct extensive experiments and select the following methods to try to recover the motion trajectories accurately.

1) Step Detection: The accuracy of step detection affects the accuracy of PDR greatly. Step detection is based on the detection of peak and valley oscillation on the magnitude signal of the accelerometer data. Because the motion trajectories may be collected by phones using different brands of sensors and the users may take the phones in different places and poses, the magnitudes of the accelerometer data are different in different trajectories, or even different in different segments of one trajectory.

To detect steps correctly in different application scenarios, we use ADAPTIV [4], an adaptive jerk pace buffer based step detection algorithm. Jerk is the acceleration change from peak to valley in a step oscillation. Pace is the time duration of a step. ADAPTIV proposes adaptive jerk and adaptive pace thresholds, and proposes a buffer to update the jerk and pace thresholds. It shows reliable step detection accuracy for accelerometer data collected by different phone poses, which provides better accuracy than the step detection algorithm implemented in Android system.

2) Stride Length Estimation: Stride length estimation is to estimate the displacement \( S \) in each step. [5] shows that the stride length changes significantly for the same person when the gait type changes. Therefore, this paper exploits the variable stride length estimation method proposed in UPTIME [5] to improve the accuracy of stride length estimation.

The method uses a hierarchical support vector machine [5] to design a gait classifier. The signal characteristics in a step are extracted to construct features, including the stride duration, accelerometer signal variance, acceleration peak, RMS acceleration etc. Then the user gait is classified by the hierarchical support vector machine using these features as input, and the stride length is estimated by looking up table using the estimated gait. The method can much improve the accuracy and adaptivity of stride length estimation.

3) Direction Estimation: Direction estimation is to estimate the heading angle \( \theta \). We exploit HUMAINE [6], a sensor fusion method to accurately estimate user orientation relative to the Earth coordinate frame.

In HUMAINE, firstly, the gravity vector is tracked by the opportunities when the acceleration vector approximately equals \( g (9.80665 m/s^2) \) and by fusion of gyroscope and magnetometer data. Whenever the trend of gyroscope and magnetometer reading is correlated, the magnetometer reading is believed accurate [7] and is selected to estimate the phone’s direction.

By tracking gravity vector and the correct magnetometer readings, equations for estimating roll and pitch are set up by relating phone pose with the gravity vector. Then the equation to estimate yaw angle (phone’s orientation) is set up based on the magnetometer readings. But the phone’s orientation maybe different from the user’s motion orientation. So a principal component analysis (PCA) method is applied on the extracted linear acceleration data of the phone to find the direction with the largest variance. The first principle direction with acute angle with the phone orientation is then detected as the user moving orientation.

B. Floor Classification

The first step to register the PDR trajectories is to find the trajectories that are collected from the same floor. One
trajectory maybe collected when a user walks through multiple floors. Such trajectories should be cut into segments of different floors for registering the trajectory segments of the same floor to recover the floor routes.

The barometer reading is an important clue to distinguish floor index. However, barometer sensors have good accuracy in measuring relative pressure changes, but have unsatisfied accuracy in measuring absolute pressure. This is because the biases of barometer readings are different in different phones. We therefore can use barometer readings to confidently separate a trajectory into segments of different floors but the barometer readings cannot tell exactly the floor index of the segments.

We find that the trajectories on the same floor share more common MAC addresses of the WiFi APs, and the trajectories on different floors share much less common MAC addresses. We therefore propose to use the Jaccard coefficient of MAC address appearing in trajectory segments to further classify the trajectory segments to classify their floor indices.

Firstly, the PDR points in a trajectory is clustered by DBSCAN method regarding the barometer readings of the points, which automatically cluster the trajectory data into multiple clusters if the trajectory travels multiple floors. Each cluster is a trajectory segment in one floor.

Then all the trajectory segments separated from different trajectories are put together. For each trajectory segment (say trajectory segment $s_i$), the set of AP MAC address appearing in $s_i$ is extracted and is denoted by $f_i$. Then for any two trajectory segments $s_i$ and $s_j$, we have the following two MAC address sets:

\[
\begin{align*}
    f_i &= \{mac_{i1}, mac_{i2}, ..., mac_{im}\} \\
    f_j &= \{mac_{j1}, mac_{j2}, ..., mac_{jn}\}
\end{align*}
\]  

where $mac$ is the MAC address of an AP. We then obtain the intersection and union of $f_i$ and $f_j$.

\[
\begin{align*}
    MAC_{\text{int}} &= f_i \cap f_j \\
    MAC_{\text{uni}} &= f_i \cup f_j
\end{align*}
\]  

The Jaccard coefficient of $f_i$ and $f_j$ is calculated using the following equation:

\[
Jac_{ij} = \frac{|MAC_{\text{int}}|}{|MAC_{\text{uni}}|} \tag{4}
\]

where $|X|$ is the carnality of the set $X$. Finally, using jaccard coefficient $Jac_{ij}$ as a distance, a hierarchical clustering [8] is performed on the trajectory segments to obtain specific floor indices. The final cluster of trajectory segments with the highest average pressure is the trajectories of the first floor. An example is shown in Fig[3] The three trajectories are segmented and are clustered into trajectory segments of four floors.

C. Vertex and Edge Extraction On Each Trajectory

Then we consider feature extraction on each trajectory segment. Traditional RSS features are noisy. But different PDR trajectories on a same floor show rather stable relative turning patterns at common turning points. We therefore propose to extract the reliable turning points (TPs) on the trajectories. The extracted turning points become vertices. The RSS at the turning point is used as vertex feature and the motion vectors among the turning points will become edges. This will turn each trajectory segment into a chain-type graph.

1) Turning Points Detection: The ideal turning point detection is shown in Fig[5] where reliable turning points are detected as feature points. Because user motions and the sensor data have noises, there are heading direction noises in each step, as shown in Fig[6]. To detect reliable turning points, we need on one hand to avoid false alarming caused by heading noises, and on the other hand to reliably detect turning when the accumulated heading direction has changed enough.

We therefore design a sliding window based turning point detection algorithm. The algorithm starts by using the first point as the origin, $V$ is the detected set of turning points which is initialized by $V = 1$. Starting from $(x_1, y_1)^T$, whenever a turning point (denoted as point $i$) is detected, this turning point will become the starting point of a new window, and is set as the origin of the new window. For each point following this point, i.e., $i + 1, \cdots, i + t$, turning angle $\{\alpha_{i+1}, \cdots, \alpha_{i+t}\}$ are calculated by

\[
\alpha_{i+t} = \left| \frac{y_{i+t}}{x_{i+t}} \right| - \left| \frac{y_{i}}{x_{i}} \right|, \quad t = 1, 2, \cdots \tag{5}
\]

When $\alpha_{i+t} \leq \varepsilon$, $i + t$ is added to the current window. Only when $\alpha_{i+t} > \varepsilon$, $i + t - 1$ will be detected as a new turning point. $i + t - 1$ is added into the turning point set. The window length is reset to 1 and the point $i + t - 1$ becomes the new origin. By this method, the sliding window filters out the turning noises and detects a turning point when enough turning has been accumulated. For a trajectory segment $T_i$, all detected turning points will be extracted as vertices, denoted by set $V_i$. The motion vector between each pair of vertices is a motion edge. The edges form a edge set $E_i$. So the trajectory $T_i$ is represented by a chain-type graph $G_i = (V_i, E_i)$.

D. Delete Frequent Turning Segments

Turning points will be detected frequently if a user turns frequently during data collection. Such kinds of trajectories cannot accurately reveal the indoor path features, because the turning are mainly caused by the user’s irregular motions. Since there are redundant trajectories, we don’t need these “bad” trajectories. We therefore propose to delete trajectory segments with continuous turning points. If some turning points appear very closely on a trajectory, such sub-trajectories will be deleted to avoid the frequent turning’s impacts to the trajectory registration. We propose to:

1) Divide a trajectory into two trajectories if two turning points are detected successively.
2) Delete a sub-trajectory if the length of a sub-trajectory is too short.

The algorithm checks the vertex set sequentially. If two neighboring vertices $v_i, v_{i+1}$ are not successive in the original trajectory, the vertices are added into current trajectory graph $G_i$. If two neighboring vertices are successive, a new sub-trajectory graph $G_{i+1}$ will be generated and using $v_{i+1}$ as the starting point of this new sub-trajectory graph. If $G_i$ is too short, $G_i$ will be deleted, since the short sub-trajectories can
not well reveal the path structure. An example is shown in Fig. 7.

E. The Motion Vector and the RSS Vertex Feature

After forming each chain-type graph $G_i = (V_i, E_i)$, the motion vector between each pair of vertices is calculated. Note that $v_1$ stores the $i$th vertex’s origin index in the trajectory. Let $w_{j-1,j}$ be the motion vector between $v_{j-1}$ and $v_j$. Then

$$w_{j-1,j} = \vec{u}_{v_{j-1}} + \vec{u}_{v_{j-1}+1} + \ldots + \vec{u}_{v_j-1}$$

where $\vec{u}_{v_j-1}$ is the motion vector in the original trajectory from node $v_{j-1}$ to node $v_j + 1$. Note the summation of motion vectors between successive turning points smooths the step-by-step motion noises. The motion vector gives the motion estimation between each pair of vertices. We then use $Z_i = \{z_{v_1}, z_{v_2}, \ldots, z_{v_{n_i}}\}$ to represent the RSS signature of vertices. So a trajectory segment $T_i$ is converted to a chain-type graph $G_i = (V_i, E_i, W_i, Z_i)$, where $V_i$ are reliable turning points; $W_i$ represent the motion vectors of edges; and $Z_i$ represents the RSS features of vertices. Note that each trajectory graph consists only one path, which likes a chain.

IV. EXPERIMENT RESULTS FOR TRACK GENERATION

A. Performance for Floor Classification

First, we analyze the performance of DBSCAN algorithm for separating the trajectories to different floors. We set the maximum number of clusters in DBSCAN as 20. For each dataset, we put all trajectories together for floor clustering. By comparing with the ground truth, the floor classification accuracy is shown 100% for all the trajectories in each dataset. The results are shown in Table 1.
| Data-set         | Floor number | Accuracy |
|------------------|--------------|----------|
| Dangdai Mall     | 3            | 100%     |
| Atlantis le Centre | 3        | 100%     |

B. Optimizing the Parameters for Turning Point Detection

The turning detection threshold $\varepsilon$ and sliding window length $t$ of the turning point detection algorithm are optimized using the experimental data. For evaluating turning point detection accuracy, we use precision, recall and F-measure as the evaluation indices for turning point detection.

Using the dataset of Dangdai Shopping Mall, we evaluated the precision, recall and $F$ of turning point detection when $\varepsilon$ varies in the range $[0.6, 1.4]$. The results are shown in Fig(8(a)). It can be seen that $\varepsilon = 1$ obtains the highest $F$-measure. Then, by fixing $\varepsilon = 1$, we calculated the precision, recall and $F$ of as a function of the window length $t$ when $t$ varies in $[1, 5]$. The result is shown in Fig(8(b)). It can be seen that when $t = 4$, the precision reaches 100%. So turning point detection parameters are set as $t = 4, \varepsilon = 1$ in the final experiments.

REFERENCES

[1] Z. Li and T. Braun, “Passively track wifi users with an enhanced particle filter using power-based ranging,” IEEE Transactions on Wireless Communications, vol. 16, no. 11, pp. 7305–7318, 2017.
[2] Z. Li, T. Braun, and D. C. Dimitrova, “A time-based passive source localization system for narrow-band signal,” in 2015 IEEE International Conference on Communications (ICC). IEEE, 2015, pp. 4599–4605.
[3] A. R. Jiménez, F. Seco, and J. Torres-Sospedra, “Tools for smartphone multi-sensor data registration and gt mapping for positioning applications,” in 2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN). IEEE, 2019, pp. 1–8.
[4] D. Murray and R. Bonick, “Adaptiv: An adaptive jerk pace buffer step detection algorithm,” 2018.
[5] M. Alzantot and M. Youssef, “Uptime: Ubiquitous pedestrian tracking using mobile phones,” in 2012 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2012, pp. 3204–3209.
[6] N. Mohssen, R. Mottaz, H. Aly, and M. Youssef, “It’s the human that matters: Accurate user orientation estimation for mobile computing applications,” arXiv preprint arXiv:1411.2156, 2014.
[7] P. Zhou, M. Li, and G. Shen, “Use it free: Instantly knowing your phone attitude,” in Proceedings of the 20th annual international conference on Mobile computing and networking, 2014, pp. 605–616.
[8] S. C. Johnson, “Hierarchical clustering schemes,” Psychometrika, vol. 32, no. 3, pp. 241–254, 1967.

V. Biography Section

Wanting Li is a PhD in the Department of Computer Science, Renmin University of China. She received the BS degree in Computer science and technology from China Agricultural University (2019). She won Honorable Mentions in 2018 mathematical modeling contest of American College Students. Her research is unsupervised learning pervasive computing.

Yongcai Wang received his BS and PhD degrees from department of automation sciences and engineering, Tsinghua University in 2001 and 2006 respectively. He worked as associated researcher at NEC Labs. China from 2007-2009. He was an research scientist in Institute for Interdisciplinary Information Sciences (IIS), Tsinghua University from 2009-2015. He was a visiting scholar at Cornell University in 2015. He is currently associate professor at Department of Computer Sciences, Renmin University of China. His research interests include network localization, and combinatorial optimization and applications.

Deying Li received the MS degree in Mathematics from Huazhong Normal University (1988) and PhD degree in Computer Science from City University of Hong Kong (2004). She is currently a Professor in the Department of Computer Science, Renmin University of China. Her research includes wireless networks, mobile computing, social network and algorithm design and analysis.