Track Your Foot Step: Anchor-free Indoor Localization based on Sensing Users' Foot Steps

Chang Liu†, Lei Xie†, Chuyu Wang†, Jie Wu‡, Sanglu Lu†
†State Key Laboratory for Novel Software Technology, Nanjing University, P.R. China
‡Department of Computer Information and Sciences, Temple University, USA
Email: †liuchang@dislab.nju.edu.cn, †lxie@nju.edu.cn, †wangcyu217@126.com, ‡jiewu@temple.edu, †sanglu@nju.edu.cn

***Abstract*—Currently, conventional indoor localization schemes mainly leverage WiFi-based or Bluetooth-based schemes to locate the users in the indoor environment. These schemes require to deploy the infrastructures such as the WiFi APs and Bluetooth beacons in advance to assist indoor localization. This property hinders the indoor localization schemes in that they are not scalable to any other situations without these infrastructures. In this paper, we propose FootStep-Tracker, an anchor-free indoor localization scheme purely based on sensing the user's footsteps. By embedding the tiny SensorTag into the user's shoes, FootStep-Tracker is able to accurately perceive the user's moving trace, including the moving direction and distance, by leveraging the accelerometers and gyroscopes. Furthermore, by detecting the user's activities such as ascending/descending the stairs and taking an elevator, FootStep-Tracker can effectively correlate with the specified positions such as stairs and elevators, and further determine the exacted moving traces in the indoor map by leveraging the space constraints in the map. Realistic experiment results show that, FootStep-Tracker is able to achieve an average localization accuracy of 1m for indoor localization, without any infrastructures having been deployed in advance.**

I. Introduction

Recently, indoor localization schemes have been widely used to support various applications such as context-aware or location-based services. Conventional localization schemes mainly leverage WiFi-based or Bluetooth-based schemes to locate the users in the indoor environment. These schemes primarily require the deployment of the infrastructures such as WiFi APs and Bluetooth beacons in advance to assist indoor localization. However, for a number of indoor environments, it is impossible (or rather expensive) to deploy such a large number of devices as the localization infrastructures. This property hinders the indoor localization schemes in that there are not scalable to any other situations without these infrastructures. Therefore, it is essential to design a brand new approach for indoor localization without any requirement for the infrastructure.

Recently, a few researchers have sought to leverage the devices with embedded sensors, such as smart phones [1–3] and wearable bracelets, to position and track the indoor environment users. However, the previous work in positioning and tracking the users have had the following common limitations: First, they usually put devices like smart phones into the user's pant pocket and perceive the user's movements via the embedded sensors. They cannot accurately capture the user's movements, including the moving directions and distances, due to the inappropriate placement of sensors. Second, they conventionally estimate the moving distance by counting the foot steps, while assuming the user's step length remains to be a constant value. This approach is not adaptive to the variation of user's moving activities, since the user may sometimes walk with small steps, and sometimes jog with large steps. Third, they still need to leverage the anchor nodes like the WiFi APs to help determine the exact position in the map. This increases their dependence on the surrounding infrastructure.

Fig. 1. The SensorTag used in FootStep-Tracker. We embed two tags into the insoles and use an Android phone to collect and process the sensor data.

In this paper, we propose FootStep-Tracker, an anchor-free indoor localization scheme purely based on sensing the user's footsteps. Our novel solution is based on the observation that the user's moving activities can be effectively inferred from his/her footsteps by leveraging the tiny sensors embedded in shoes, such as accelerometers and gyroscopes. As is shown in Fig. 1 (a), by embedding the tiny sensor like the SensorTag [4] into the user's shoes, FootStep-Tracker is able to accurately perceive the user's moving traces, including the moving direction and distance, by leveraging the accelerometers and gyroscopes. Fig. 1 (b) shows the FootStep-Tracker Android app. Furthermore, by detecting the user's activities such as ascending/descending the stairs and taking an elevator, FootStep-Tracker can effectively correlate with the specified positions such as the stairs and elevators, and further determine the exact moving traces in the indoor map, by leveraging the space constraints in the map.

There are several challenges building the indoor localization scheme purely based on sensing the user's footsteps. First, it is difficult to accurately estimate the user's horizontal step movements. Since the sensors are embedded in the shoes, they actually capture the feet's movement in the air while the user is moving, and thus the user's horizontal movement cannot

be directly derived from the collected sensor data. To address this challenge, we leverage the gyroscope to measure the angle between the foot's direction of movement and the ground, and leverage the accelerometer to measure the actual movement of the foot. We then build a geometric model to estimate the horizontal movement. Second, it is difficult to accurately estimate the user's moving direction during the movement. While tracking the user's foot steps, the angle variation of the foot steps cannot be directly correlated to the user's moving direction. To address this challenge, we build a geometric model to depict the relationship between the angle variation of the foot steps and the moving direction, and further derive the user's moving direction from the measurements from the embedded sensors. Third, to realize the indoor localization, it is essential to determine the exact moving traces in the indoor map. To address this challenge, we use activity sensing to effectively figure out the reference positions, such as the elevators and stairs, and further leverage the space constraints in the indoor map to filter out those infeasible candidate traces, so as to fix the moving traces in the indoor map.

We advance the state of the art on positioning and tracking the users from three perspectives. First, we propose an anchor-free indoor localization purely based on sensing the user's footsteps, without the support of any infrastructure. Second, we propose efficient solutions to accurately estimate the moving direction and distance, by only leveraging the low-cost inertial sensors like accelerometer and gyroscope. Third, we leverage activity sensing to effectively figure out the reference positions during the process of tracking the user, so as to further determine the exact moving traces in the indoor map.

II. Related Work

A. Infrastructure based Indoor Localization

Infrastructure based indoor localization schemes primarily use wireless signal, such as RF signal and WiFi signal, to locate the users or objects in the indoor environment. Several location algorithms such as *Fingerprint*[6] and LANDMARC[7] have been proposed and widely accepted in the academic area. Yang et al. [9] proposed *Tagoram*, an object localization system based on COTS RFID reader and tags. By proposed *Differential Augmented Hologram* (DAH), *Tagoram* can recover the tag's moving trajectories and achieves a milimeter location accuracy in tracking mobile RFID tags. Xiao et al. [10] proposed *Nomloc* which dynamically adjusts the WLAN network topology by nomadic WiFi AP to address the performance variance problem. By the proposed space partition based algorithm and fine-grained channel state information, *Nomloc* can effectively mitigate the multipath and NLOS effects.

B. Infrastructure-free based Indoor Localization

State-of-the-art infrastructure-free based indoor localization schemes, especially for pedestrian navigation work track the user by detecting the user's movement with the IMU sensors, and dead-reckoning is the most popular scheme which estimate the object's current position by it's previous determined position.[3, 11–17]. Leppäkoski et al. [11] proposed an IMU sensors, WLAN signals and indoor map combined localization system. By using extended Kalman filter to combine the sensor with WLAN signal and particle filter to combine the inertial data with map information, the diverse data are fused well to improve the pedestrian dead reckoning. Vidal et al. [12] present an indoor pedestrian tracking system with the sensor on the smart phone. Combined with the dead-reckoning and the gait detecting approach, and aided by the indoor signatures such as corners, the system have an acceptable location accuracy. Wang et al. [13] present *UnLoc*, which leverage the identifiable signal signatures of indoor environment which can be captured by the sensor or WiFi to improve the dead-reckoning method. With *UnLoc*, the localization system converge speed can be effectively improved. Fourati et al. [15] proposed an Complementary Filter algorithm to process the sensor data, and combined with *Zero Velocity Update* (ZVU), the system can locate the user with high accuracy. Rai el al. developed ZEE [3], which leverages the smart phone built-in sensors, tracking the user when he travels in an indoor environment, and scanning with WiFi signal simultaneously. By combining the sensors and WiFi, ZEE uses crowdsourcing to locate the user, achieving a meter-level location accuracy.

Different from the previous work, in this paper, we propose an anchor-free indoor localization system. By sensing the user's foot step and utilizing the reference position and constraint of the indoor map, FootStep-Tracker track the user's location without any deployment of anchor nodes.

III. System Overview

In our system, called FootStep-Tracker, we focus on how to track the user's position based on the low-cost inertial sensors embedded inside the shoes, according to a given indoor map. Fig.2 shows the framework of FootStep-Tracker. First, the *Activity Classifier* is designed to classify the user's activities into two activity groups, i.e., walking and reference activities such as ascending/descending the stairs, and the elevator ascending/descending, according to the raw sensor data of gyroscope and accelerometer. In regard to the walking activity, we measure the moving distance based on the *Step Segmentation* and *Step Length Estimator*, and measure the moving direction based on *Moving Direction Estimator*. According to the moving distance and moving direction, we reconstruct the user's moving trace relative to the starting point. Meanwhile, it is possible to derive the reference positions according to the activity sensing results from the *Activity Classifier*. For example, the reference positions can be the elevators if the activity of elevator ascending/descending is detected. Furthermore, by leveraging the space constraints in the indoor map to filter out those infeasible candidate traces, our solution could finally determine the user's trace in the indoor map.

The components of FootStep-Tracker are as follows:

1) **Activity Classifier.** It extracts corresponding features from the inertial sensor data of human movement, then it estimates the user's current activities via the classifica-

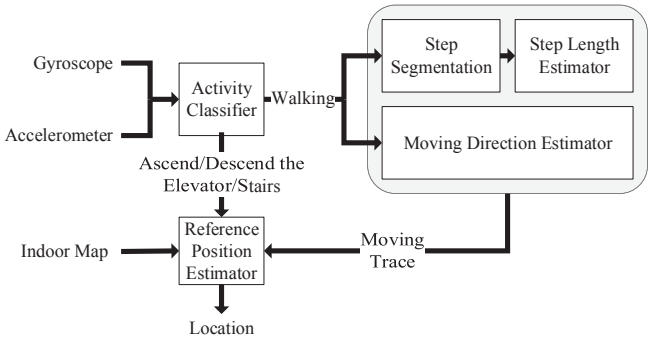


Fig. 2. Framework of FootStep-Tracker. By input the sensors' data and the indoor map, FootStep-Tracker outputs the user's location in time.

tion techniques such as decision tree and hidden Markov model.

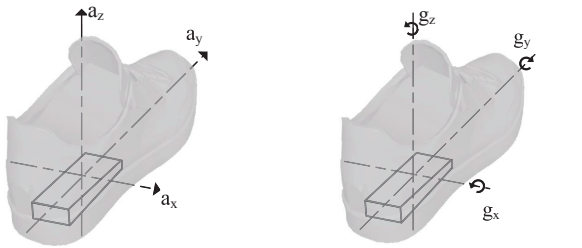
2) **Step Segmentation.** In regard to the activity of walking, it splits the sequential inertial sensor data into segments, each segment represents a complete process of footstep during walking.
3) **Step Length Estimator.** It estimates the distance of each step in the horizontal line. We use a geometric model to depict the footstep movement and rotation of the foot during one step, and then project the step length in the air to the horizontal line, by leveraging the accelerometer to estimate the step length in the air and the gyroscope to estimate the projection angle.
4) **Moving Direction Estimator.** It estimates the turning angle during the process of walking. We use a geometric model to depict the relationship between the angle variation of the foot steps and the moving direction, and further derive the user's moving direction from the measurements from the embedded sensors.
5) **Reference Position Estimator.** It estimates the reference positions in the given indoor map, such as elevators and stairs, according to the results in activity sensing. In this way, the moving trace can be fixed in the indoor map.

IV. System Design

System Deployment. FootStep-Tracker processes the data captured by the sensors which is embedded in the user's shoes. Without loss of generality, we use CC2541 SensorTag[4] which is produced by *TEXAS INSTRUMENTS*. We sample the accelerometer and gyroscope with 20Hz, analysing data and presenting the result of localization by an android smart phone carried by the user. For the convenience of further discussion, we present the axes on the SensorTag coordinate system in Fig. 3. We denote the three-axis acceleration as a_x, a_y, a_z, and the three-axis angular velocity as g_x, g_y, g_z.

A. Activity Classifier

Motivation. For the purpose of estimating the moving trace and reference position, we first need to know what the user is currently doing. In our scene, we need to classify the user's activity into two main classes: walking and reference activities. If the user is walking, we use the sensor data



(a) Axes of accelerometer. (b) Axes of gyroscope.

Fig. 3. Axes on SensorTag.

to estimate the user's moving trace. If the user is doing reference activities, including *ascending/descending the stairs and ascending/descending the elevator*, we use it to find the reference positions in the map. Besides, if the user is detected as standing still, we keep sensing the sensors.

Observation and Intuition. The acceleration a_z is strongly relative with the six activities. That is because when the user is standing still, the direction of z-axis is along the vertical direction which is the opposite direction of the gravity. Besides, the acceleration is constant, which differs from the periodicity fluctuant acceleration of walking and climbing stairs. And when the user is moving up or down, such as ascend/descend the stairs, the foot's movement is along the vertical direction which can be sensed well by a_z.

We first collect a_z for each activity. Fig.4 shows the acceleration of different activities. Fig.4 (a) shows that when the user is standing still, a_z almost stays constant, and the amplitude equals to the gravity. Fig.4 (b-d) show that when the user is walking or ascending/descending the stairs, a_z changes periodically. Fig.4 (e) shows the process of a user ascending the elevator. The red box in the figure shows that the acceleration first gets smaller than the gravity, then gets larger. While the elevator accelerates to have an upward speed, the user is under the hypergravity condition and the a_z is smaller than the gravity. Then the elevator rises in a constant speed, with the user's speed relative to the rest of the elevator. Meanwhile, the a_z is equal to the gravity. Finally, the elevator slows down, the user is under the weightlessness condition, and a_z has a negative, but bigger reading than the gravity. Fig.4 (f) shows the the process of an elevator descending which is a opposite the ascending process.

Solution. To classify the user's activities, we first segment the sequential data into windows, then classify the window by a hybrid method. Generally, the human step frequency is 1Hz to 3Hz. That is to say, the period of a step will last from 0.3s to 1s. The weightlessness and hypergravity process in the elevator will commonly last for about 2 seconds. We use a slide window with size 40, which equals to 2 seconds in time, to ensure that the window contains an entire step period during walking or a process of hypergravity and weightlessness.

We classify the window into eight classes, which are the description shown in the Fig.5. Table I shows the description of each abbreviation. Firstly, we note that UST, DST and WALK obviously have a higher variance than EHG, EOW and SS. To classify this two activities groups, we use a decision tree with

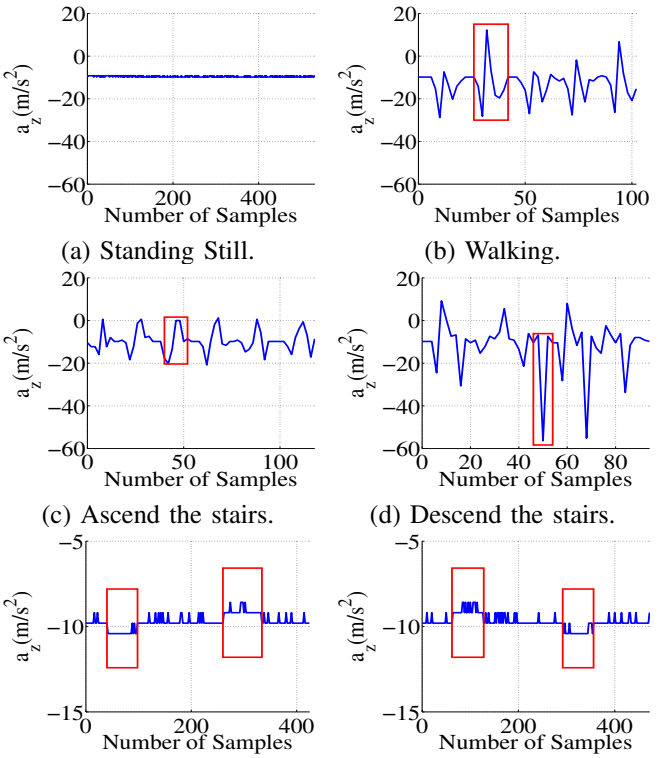


Fig. 4. Accelerometer data of vertical direction(z-axis, contains the gravity about -9.8) when the user is standing still, walking, ascending/descending stairs and taking an elevator.

a threshold on the variance of window.

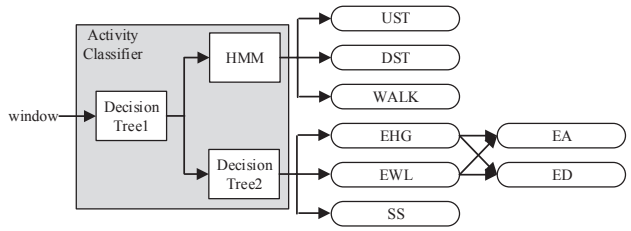


Fig. 5. Activity Classifier

TABLE I
LABEL DESCRIPTION

Abbrev	Description	Abbrev	Description
UST	ascend the stairs	EWL	weightlessness in elevator
DST	descend the stairs	EA	ascend the elevator
WALK	walking	ED	descend the elevator
EHG	hypergravity in elevator	SS	stand still

Fig.6 (a) shows the CDF (Cumulative Distribution Function) plot of the two groups' window variances of a_z, which contains about 700 windows collected among three different users.

The a_z almost stays constant when a user is standing still or is taking an elevator. Meanwhile, it has a larger variance while the user is walking or ascending/descending stairs. Moreover, there is an obvious bound between the two groups, which can be selected as the threshold. There is no such an obvious bound to classify the UST, DST and WALK. However note that, the fluctuation of UST, DST and WALK is different as we have

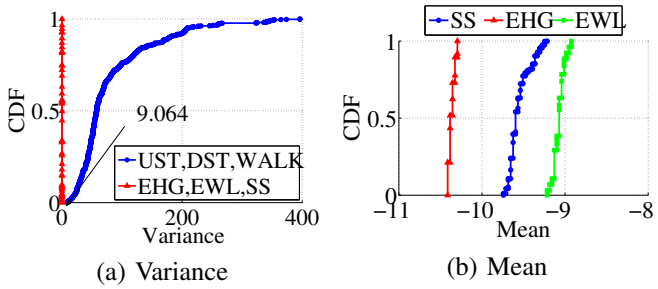


Fig. 6. CDF of variance and means for different activity.

mentioned before. We use Hidden Markov Model (HMM) for the classification. To classify EHG, EOW and SS, we also notice that the mean value of the window is different, caused by the hypergravity and weightlessness. Fig. 6 (b) shows the CDF plot of the three activities' mean values of a_z window, which contains about 200 windows collected among three five different users. Further more, if we estimate the user's activity as EHG, then we wait for an EOW, and we say the user is under EA. If the user is under EOW, we wait for an EHG, and we say the user is under ED.

B. Step Segmentation

To estimate the length of each step, we first need to split the raw sequential data into each step. Human walking is a periodical movement along the moving direction, which has a specific pattern in sensors' reading. The direction of y-axis is almost the same as the moving direction, we do step segmentation on a_y and assisted by a_x and a_z. Fig.7 shows the acceleration of three-axis while the user is walking. Note that, after the foot touches the floor, and before it lifts up, it is relative static to the ground and the accelerometer have a constant reading, which we called "static zone".

The red boxes in Fig.7 shows the "static zone" of accelerometer. To avoid the mistake segmentation caused by the activities which is similar to walking, such as swing the leg, we also detect "static zone" on a_x and a_z. If the current activity is walking, *Step Segmentation* takes the raw data as input, segmenting a_x, a_y, a_z by "static zones" which contains six consecutive samples which range from 0 ± 0.5 on a_x, a_y and -9.8 ± 0.5 on a_z. We extract the window between the "static zone" window for each axis. And get intersection elements of the three as our segmented data for the current step.

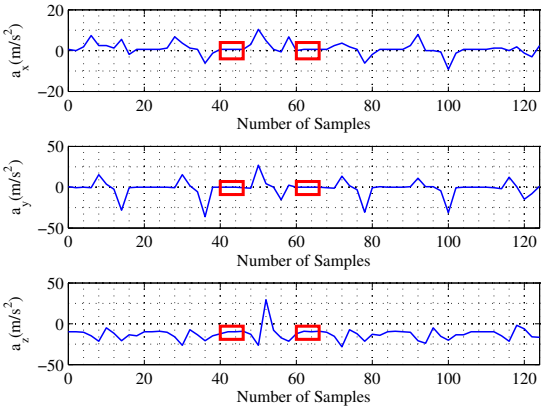


Fig. 7. the data of accelerometer of walking.

C. Step Length Estimator

Motivation. For the purpose of depicting the user's moving trace, we need the user's moving distance. Different users have different step length according to their figure. For a specific user, lot of existing step length estimation schemes are based on the assumption that the step length is invariable during a period of time. While we believe that the user's step length may change frequently in some cases, such as walking with small steps and jogging with large steps. *Step Length Estimator* estimates the length step by step, which can sensing the change of the user's stride in time.

Challenge. The step length is not exactly the length of the foot's moving trace in the air. Instead, as depicted by the red dotted line in Fig.8, it is the moving trace's projection on the ground. Therefore, we cannot directly derive the step length by the double integral on a_y.

Observation and Intuition. Fig.8 (a) depicts the moving process of the feet. As shown in the figure, the y-axis is not always horizontal, we project it on the horizontal plane and denote it as foot direction, and the angle between the y-axis direction and foot direction as θ.

Fig.8 (b) shows the sensor's data corresponding to (a). As shown in Fig.3, in the sensor's coordinate system, the forward direction is positive of a_y and the anticlockwise direction is positive of x-axis of g_x. At phase (1), the foot is relative static to the ground, corresponding to a few zero values on a_y and g_x. At phase (2), the foot actually does not have a forward acceleration. But the heel uplifts, leading to a negative reading in g_x. As the y-axis is no longer horizontal, a_y is slightly less than zero caused by the gravity. We denoted the time at begin of phase (2) as *uplift time, i.e.,* T_u. At phase (3), the foot starts to move forward. Instep moving upwarp, leading to a positive reading in g_x. The entire foot accelerates forward and causes a positive reading in a_y. We denoted the time at begin of phase (3) as *liftoff time, i.e.,* T_l. At phase (4), the foot decelerates to static, touch the land and the instep downwarps. We denoted the time at begin of phase (4) as *landing time, i.e.,* T_d. At phase (5), the heel touches down the land and rests again. We denote the time at begin of phase (5) as *rest time, i.e.,* T_r.

Besides, due to the toe in and toe out, the forward horizontal acceleration along the foot direction can not represent that along the moving direction. Fig.9 depicts the situation. a_x denotes the x-axis acceleration, a_m denotes the acceleration along the moving direction, and a_f is the acceleration along the horizontal foot direction. There is an angle φ between the moving direction and foot direction. The relationship of the three acceleration a_m, a_f, a_x can be represent as Eq.(1).

$$a_m = a_f cos(\varphi) + a_x sin(\varphi) \tag{1}$$

Given the segmented sensor data by *Step Segmentation*, we first extract the critical time, including uplift time, liftoff time, landing time and rest time. Then we estimate the step length by integral on the a_m from liftoff time to landing time. Lastly, as we embedded sensor in both shoes, we use double feet calibration to reduce the error further, getting the calibrated step length.

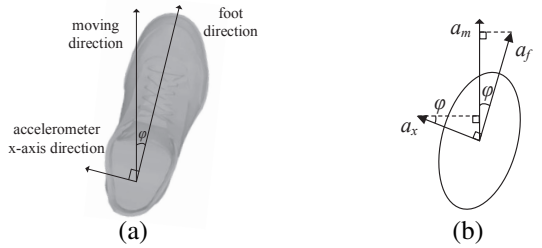


Fig. 9. The toe-in and toe-out situation.

Critical Time Extraction. As we mentioned above, only the foot's movement in phase (3) leads to the displacement which happens between the uplift time and liftoff time. Besides, the angle θ changes from uplift time to landing time. So we extract the critical time, which is uplift time, liftoff time, landing time and rest time. Given the segmented data by *Step Segmentation*, which only contains the phase (2)-(4) data, FootStep-Tracker extracts critical times in the data sequences. At the *uplift time*, the heel uplifts, the g_x starts to be negative, but a_y is slightly less than zero. We extract backward from the segment, taking the time when g_x starts to be negative as *uplift time*. At the *liftoff time*, the foot just starts to move forward. We extract among the segment, taking the time when $a_y(t) < 0$, and $a_y(t+1)$ is positive as *liftoff time*. At the *landing time*, the heel touch the ground, a_y declines to negative. At the *rest time*, g_x and a_y start to be zero again. We extract the first time when ay, gx become zero.

Algorithm 1: Critical Time Extraction.

Input: Sequential data a_y, g_x, Segmented data for current step D_s

Output: Uplift time T_u, liftoff time T_l, landing time T_d, rest time T_r

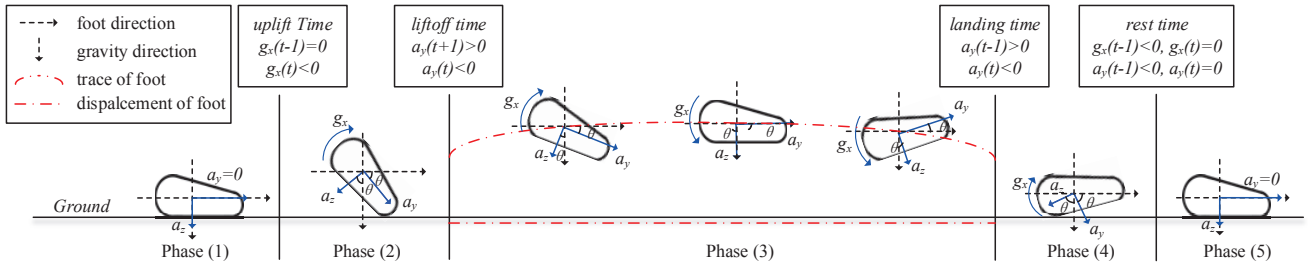
1 Find the T_u backward from the beginning of D_s until the data at time t satisfies that $g_x(t-1) = 0, g_x(t) < 0$;
2 Find the T_l backward from the beginning of D_s until the data at time t that satisfies that $a_y(t) < 0, a_y(t+1) > 0$;
3 Find the T_d forward from the end of D_s until the the data at time t satisfies that $a_y(t-1) > 0, a_y(t) < 0$;
4 Find the T_r forward from the end of D_s until the the data at time t satisfies that $g_x(t), a_y(t)$ is equal to zero;
5 return T_u, T_l, T_d, T_r;

Step Length Estimation. The red dotted line in Fig.8 shows that the step length is not the foot's moving tracing in the air, but it's projection on the ground. Eq.(2) shows that the forward acceleration along the foot direction a_f can be calculated by a_y ,a_z, and the angle θ at each time. We project a_y, a_z on the horizontal plane, and compound them as a_f.

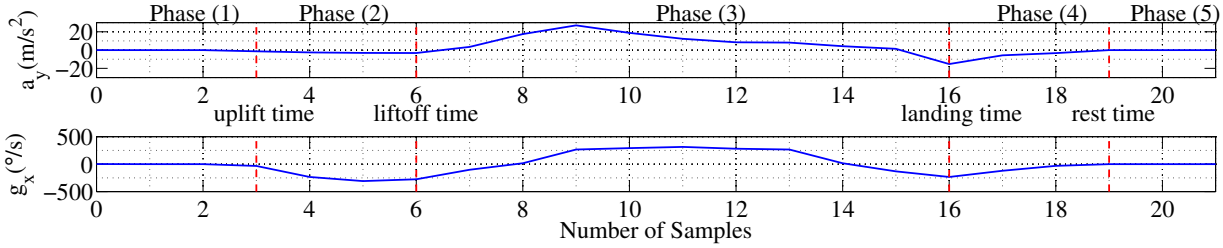
$$a_f(t) = a_y(t)cos(\theta(t)) + a_z(t)sin(\theta(t)),\ t \in [T_l, T_d] \tag{2}$$

Eq.(3) calculates the angle between y-axis direction and foot direction for each time. As the instep starts to roll at uplift time, we do the integral on x-axis gyroscope from uplift time, getting the angle θ at each time t.

$$\theta(t) = \int_{T_u}^{t} g_x(t)\ dt,\ t \in [T_u, T_r] \tag{3}$$



(a) Walking process



(b) Sensor data of the walking process

Fig. 8. The process of a complete foot step.

To calibrate the acceleration from toe-in and toe-out problem, Eq. (4) gives the way to get the acceleration along the moving direction.

$$a_m(t) = a_f(t)cos(\varphi) + a_x(t)sin(\varphi),\ t \in [T_l, T_d] \quad (4)$$

Having the acceleration along the moving direction, we can finally get the displacement of the current step by Eq.(5).

$$\begin{aligned} S &= \int_{T_l}^{T_d}\int_{T_l}^{t'} a_m(t)\ dtdt' \\ &= \int_{T_l}^{T_d}\int_{T_l}^{t'} (a_f(t)cos(\varphi) + a_x(t)sin(\varphi))\ dtdt' \\ &= S_x + S_y + S_z \end{aligned} \quad (5)$$

Here S_x, S_y, S_z is the real acceleration's projection on the accelerometer's three-axis, which are Eq.(6).

$$\begin{aligned} S_x &= \int_{T_l}^{T_d}\int_{T_l}^{t'} (a_x(t)sin(\varphi))\ dtdt' \\ S_y &= \int_{T_l}^{T_d}\int_{T_l}^{t'} (a_y(t)cos(\theta(t))cos(\varphi))\ dtdt' \\ S_z &= \int_{T_l}^{T_d}\int_{T_l}^{t'} (a_z(t)sin(\theta(t))cos(\varphi))\ dtdt' \end{aligned} \quad (6)$$

As depicted in Fig.10, to get the angle φ, we let the user to walking ahead for a constant distance s. Then we do double integral on the a_f to get the displacement along the foot direction s_f. As the angle between the Δs_f and Δs is φ, we can estimate the angle φ by $\varphi = arccos(\frac{\Delta s}{\Delta s_f})$. Besides, s_f is equal to the sum of Δs_f, and s is equal to the sum of Δs. Therefore, we can estimate the angle by $\varphi = arccos(\frac{s}{s_f})$.

Fig.11 (a) shows the raw data a_y and calibrated data a_m. Figure (b) shows the corresponding displacement. We do integral on raw data and on calibrated data. For raw data, we got an estimated step length of 1.86m. For the calibrated

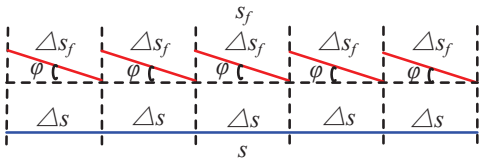
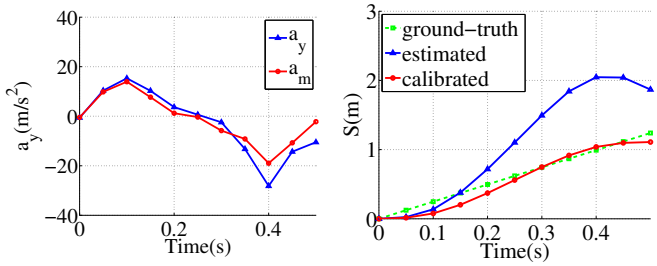


Fig. 10. Estimate the angle φ.

data, we got an estimated step length of 1.11m. Referring to the ground-truth 1.23m, our calibration reduces the error from 0.63m to 0.12m.



(a) Acceleration calibration. (b) Displacement calibration.

Fig. 11. Case study of the step length estimator.

Double feet based calibration. To further reduce the error accumulation, FootStep-Tracker embeds two sensors in both feet and respectively estimates step length. Having the intuitions that the distance between the two feet can not be too large at any time, if the difference of displacement for each foot is more than one meter, we chose the mean of them as the displacement, and restart the estimation process.

D. Moving Direction Estimation

Motivation and Challenge. To depict the user's moving trace, we also need to figure out the user's moving direction. As we embed the sensor in the shoes, we should estimate the relatively variety angle of foot when the user make turns according to the inertial sensor readings. And furthermore, due

to the different walking habits of different user, the relatively variety angle of moving direction is not exactly the angle of foot direction. So we need to estimate the moving direction by the measured foot direction.

Observation and Intuition. When the users are turning left/right, they always take the gravity direction as the axis. As depicted in Fig.1 (b), the direction of z-axis is opposite to the gravity direction. Thus we measure the g_z, which is strongly relative with the turning movement.

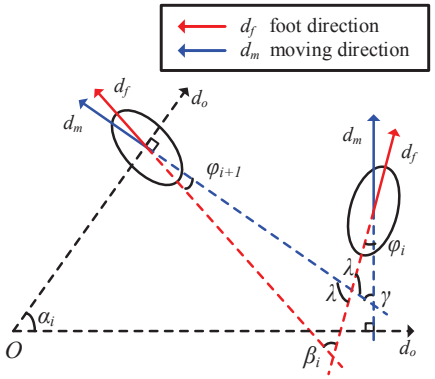


Fig. 12. Foot direction and moving direction

For a specific user, we assume the angle between the foot direction, i.e., d_f, and moving direction, i.e., d_w, is invariable during his walking process. The assumption is reasonable in our scene. That is because for a person, the degree of toe-in and toe-out is almost constant in a long time. For the propose of getting the turning degree, which is α_i, we rely on the following theorem,

Theorem 1. *Assume the angle φ between the foot direction, i.e., d_f, and moving direction, i.e., d_m, is invariable. Then the degree of the turning, i.e., α_i, is equal to the relatively variety angle of moving direction, i.e., β_i.*

Proof. Without loss of generality, as shown in Fig.12, the user makes a turning with angle of degree α around the point O. Let the direction from O to the foot as d_o, the foot direction as d_f, the moving direction as d_m, the angle between the foot direction and moving direction are φ_i and φ_{i+1}, the variety angle of foot direction is β_i, and the variety angle of moving direction is γ. Since d_w is orthogonal to d_o, then $\alpha_i = \gamma$. As the vertically opposite angles is equal, we can have that $\gamma + \varphi_i + \lambda = \beta_i + \varphi_{i+1} + \lambda$. According to the assumption, $\varphi_i = \varphi_{i+1}$, then $\gamma = \beta_i$. So we have $\alpha_i = \beta_i$. □

Moving Direction Estimation. FootStep-Tracker use low-pass filter to extract the turning steps from the steps of walking straight ahead. Besides, for a single turning step, we find that the actual time which makes the foot turn is during phase (3), from the liftoff time to the landing time. That is because at phase (1)-(2), the heel lift up, preparing the forward movement. And at the phase (4)-(5), the foot is under the landing process. At those time, the feet has no rotation around the z-axis. To divide each phase, we need to extract the critical times, which is already given by Algorithm 1.

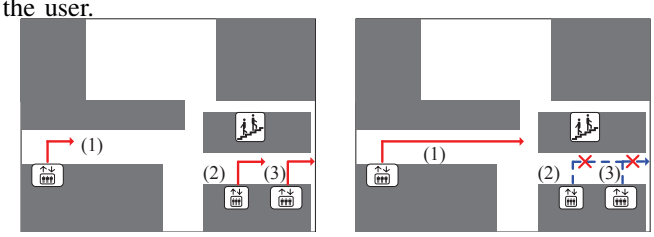
By getting the liftoff time and landing time by algorithm 1, we calculate the turning degree by Eq.(7). As the foot is swing around the z-axis of gyroscope, we integral on g_z from liftoff time to landing time, getting the turning degree of foot direction β_i of the current step. And according to Theorem 1, we have the turning degree of the moving direction for the current step α_i is equal to β_i. For a n-step turning process, we then sum the α_i up to get the turning degree α for one foot by equation $\alpha = \sum_{i=1}^{n} \alpha_i$. Then we use the mean of the two feet as the turning degree.

$$\alpha_i = \beta_i = \int_{T_l}^{T_d} g_z(t)\ dt \tag{7}$$

E. Reference Position Estimator

By *Step Length Estimator* and *Moving Direction Estimator*, we can accurately estimate the user's moving trace. However, we still need to fix the moving trace into the global indoor map. To determine the location of the user by the moving trace and the indoor map, we have two basic intuitions. First, the user's moving trace is constrained by the topological structure of indoor environment, which is to say that the user can not walking through the wall. Second, due to the reference position, such as elevators and stairs are fixed in the indoor map, we can accurately locate the user when he/she is doing the reference activities.

To locate the user, and further track the user in the indoor environment, we adopt *Snake Game*[18] strategy as depicted in Fig. 14. As shown in (a), the user firstly taking an elevator, then turning right after walking a short distance. At this time, there are three possible location according to the moving trace and the position of elevator. When it comes to (b), the user keep walking. Limited by the topological structure of indoor environment, the infeasible moving trace (2) and (3) are filtered out, and the trace (1) is the actual moving trace of the user.



(a) Feasible traces. (b) Filter out infeasible traces.

Fig. 14. Snake Game strategy.

As we have got the class of the user's activity by *Activity Classifier*, if the user is ascending/descending an elevator or ascending/descending the stairs, we find the elevator or stairs' location in the given indoor map as the reference position. After employing *Snake Game* strategy, we determine the actual user's moving trace and location. Then we keep tracking the user's location in time by the moving trace.

V. Performance Evaluation

A. Implementation

Hardware: As shown in Fig.1 (a), our system consists of a TEXAS-INSTRUMENTS CC2541 SensorTag[4]. and a SAMSUNG Galaxy S5 Android smart phone.

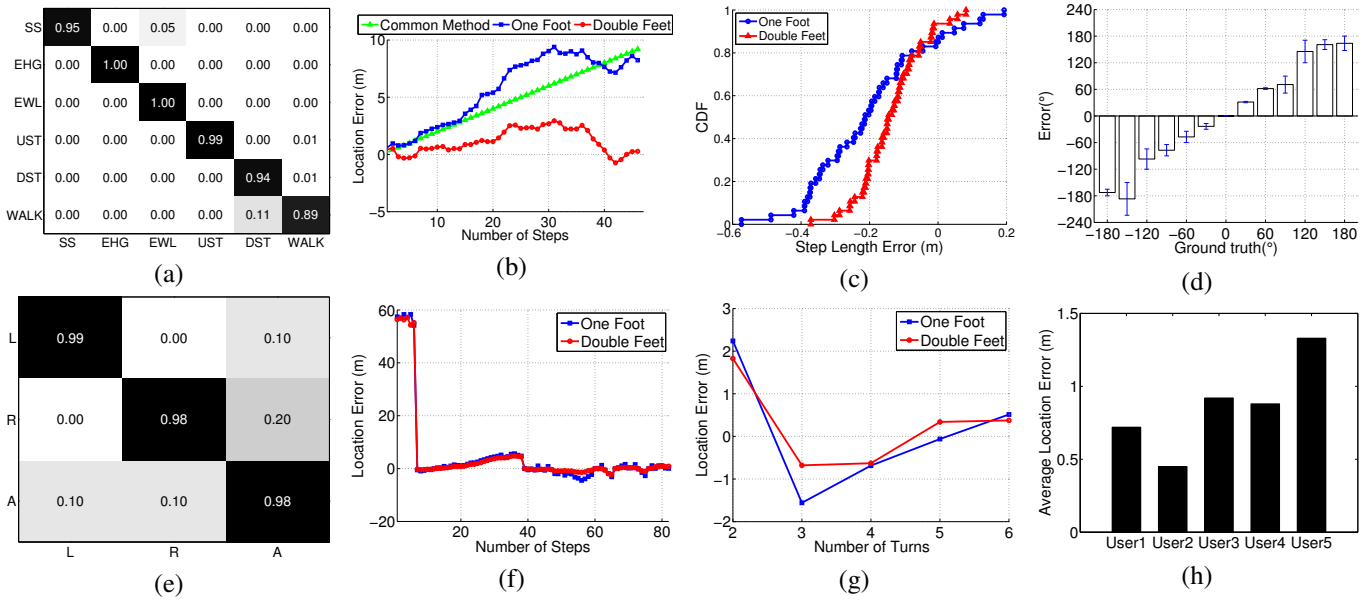


Fig. 13. Performance Evaluation. (a) Activity Classifier performance. (b) Location error when user walking along a long corridor. (c) CDF of Step Length Error (d) Moving Direction Estimator performance. (e) Classified Moving Direction Estimator performance. (f) Location error with number of steps. (g) Location error with number of turns. (h) Location error for each user.

Software: Fig.1 (b) shows the FootStep-Tracker app. We implement our system in Java on the Android platform. The sensor transmit data to the smart phone via Bluetooth. The smart phone locates and tracks the user by the given sensor data, and provides Graphical User Interface to the user.

B. Experiment Setting

We embed the FootStep-Tracker into user's shoes, collecting sensor data, and then analyzing the data by the smart phone carried by the user. The SensorTag's sample frequency is set as 20 Hz, and it is embedded in the insole as depicted in Fig.3. We chose our department building with 56m×63m as the indoor environment of the evaluation. We evaluate our scheme by the following two metrics-classification accuracy and location error. Classification accuracy calculated by the percentage of the number of window which is rightly classified of the wrongly classified. Location error evaluate the error in meter and degree from the ground-truth for *Step Length Estimator and Moving Direction Estimator*.

C. Evaluate the Activity Classifier

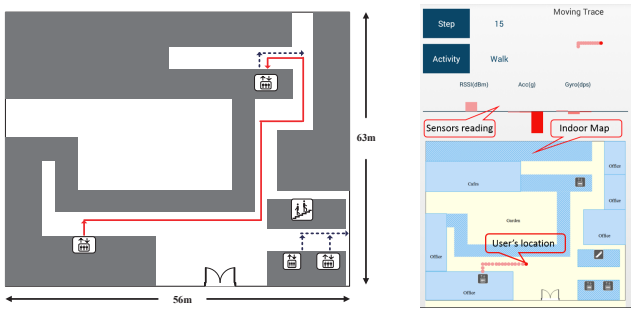
Activity Classifier could accurately classify the mentioned activities with an accuracy over 96.2%. To evaluate the accuracy of the *Activity Classifier*, we collect data for each of the six mentioned activities. For each activity, we collect about 500 windows of accelerometer and gyroscope data among three different users. Then we use them to train the HMM model and to determine the threshold for the decision tree offline. We perform the classifier on the previous three users and five new users online. They ascend/descend the stairs, walk and take elevator 10 times for each in the environment. Fig.13 (a) shows the accuracy of *Activity Classifier* on the collected data. For the part of decision tree, which is to classify SS, EHG and EWL, it is more accurate than the part of HMM. This is because SS, EHG and EWL have some essential difference among each other, such as variance and mean value. However, the UST, DST and WALK are much more similar. On average, we achieve a classification accuracy of 96.2% which is acceptable.

D. Evaluate the Step Length Estimator

Step Length Estimator could effectively reduce the accumulated error and estimate the user's moving distance. To evaluate the performance of *Step Length Estimator*, we track a user wearing the FootStep-Tracker, walking along a long corridor which is about 63m. Besides, as a comparison, we also use the common method which is multiplying the number of steps by a step length estimated by the user's height. Fig.13 (b) shows the location error. As can be seen from the curve labeled with *One Foot* and *Common Method*, the error estimated by only one foot and common method accumulates over time as the user's walking. The curve labeled with *Double Feet* shows the location error after we use our approach to estimate the step length by two feet. It reduces the error from 14.65%, i.e., 9.23m/63m and 14.60%, i.e., 9.2m/63m to 4.19%, i.e., 2.64m/63m. Fig. 13 (c) shows the CDF of step length error. It shows that the error of each step length distributes around zero, and the error when we use our double feet based calibration scheme is effectively reduced by about 0.2m.

E. Evaluate the Moving Direction Estimator

Moving Direction Estimator could estimate the user's moving direction with low error and can accurately classify the moving direction into turning left, turning right and turning around. To evaluate the performance of the *Moving Direction Estimator*, we invited eight different users, take turning from $-180°$ to $+180°$ in $30°$ increments(Left is the positive direction), 10 times for each. Fig.13 (d) shows the average error.



(a) (b)

Fig. 15. (a) Walking path in case study. (b) FootStep-Tracker Android APP.

To further improve the accuracy, we use the degree of both foot as features of the turn, and use SVM to classify the current turning into three classes: left turn (L), right turn (R) and turn around (A). We invite three users turning 30 times for left turn, right turn, turn around. Then we train the SVM model and test the performance on five new users. Fig. 13 (e) shows the accuracy of the *Classified Moving Direction Estimator*. The L/R can be classified accurately, because gz of left turn is positive and right turn is negative. There is some error that classify the L/R turn into A. That is because that the differen between a L/R turn and A is just the turning degree, which has some overlapping area. On average, we achieve an accuracy of 98.3%. Clearly, the performance of *Moving Direction Estimator* is reliable.

F. Case Study in a Real World Environment

FootStep-Tracker could accurately track the user in the indoor environment without any deployment of infrastructure. To evaluate the performance of FootStep-Tracker, we test it with 5 users. They wear FootStep-Tracker, walking along a specific path in our department building which is the full red line in Fig.15. The user takes the elevator (the left bottom one) down to this floorand finally he takes another elevator to leave the floor. At first, FootStep-Tracker cannot ensure which elevator is the user's location. The dotted line depicts the other feasible locations. But when the user keep moving, FootStep-Tracker filters out the imaginary three infeasible path by *Snake Game* strategy. Fig. 13 (f) shows the relationship between location error and the number of steps for a specific user. At the beginning, foot step tracker had a large location error caused by the undetermined location. As the user keep walking, the initial location is determined after six steps, and the error sharply reduces to about 1m for one feet scheme and 0.5m for double feet scheme. Fig. 13 (g) shows the relation between average location error of the path and number of turns. Since the error of first turn occurs when the initial location has not be determined, the error is too large that have little reference value. We evaluate the error started from the second turn. The error reduced along the number of turns obviously, and the Double Feet Scheme can further reduce the error. Fig.13 (h) shows the average location error for each user. On average, we have a location error in 1m.

VI. Conclusion

In this paper, we present a purely sensor-based scheme for indoor localization. We embed sensors into user's shoes, leveraging the accelerometer and gyroscope to estimate the user's step length and moving direction. Besides, we sense the user's activity of ascending/descending the stairs and taking an elevator, to get a reference position to help further localization. The realistic evaluation shows our scheme can achieve an average accuracy of about 1m for indoor localization.

Acknowledgments

This work is supported in part by National Natural Science Foundation of China under Grant Nos. 61472185, 61373129, 61321491, 91218302, 61502224; JiangSu Natural Science Foundation, No. BK20151390; EU FP7 IRSES MobileCloud Project under Grant No. 612212; CCF-Tencent Open Fund. This work is partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization. The work of Jie Wu was supported in part by NSF grants CNS 1449860, CNS 1461932, CNS 1460971, CNS 1439672, CNS 1301774, and ECCS 1231461. Lei Xie is the corresponding author.

References

[1] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A reliable and accurate indoor localization method using phone inertial sensors," in *Proc. of ACM UbiComp*, 2012.
[2] Y.-C. Tung and K. G. Shin, "Echotag: Accurate infrastructure-free indoor location tagging with smartphones," in *Porc. of ACM MobiCom*, 2015.
[3] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: zero-effort crowdsourcing for indoor localization," in *Proc. of ACM MobiCom*, 2012.
[4] "Sensortag," http://www.ti.com/lsds/ti/analog/sensors/overview.page.
[5] S. He, T. Hu, and S.-H. G. Chan, "Contour-based trilateration for indoor fingerprinting localization," in *Proc. of ACM SenSys*, 2015.
[6] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: indoor location sensing using active rfid," *Wireless networks*, 2004.
[7] L. Yang, Y. Chen, X.-Y. Li, C. Xiao, M. Li, and Y. Liu, "Tagoram: real-time tracking of mobile rfid tags to high precision using cots devices," in *Proc. of ACM MobiCom*, 2014.
[8] J. Xiao, Y. Yi, L. Wang, H. Li, Z. Zhou, K. Wu, and L. M. Ni, "Nomloc: Calibration-free indoor localization with nomadic access points," in *In Proc. of IEEE ICDCS*, 2014.
[9] H. Leppäkoski, J. Collin, and J. Takala, "Pedestrian navigation based on inertial sensors, indoor map, and wlan signals," *Journal of Signal Processing Systems*, 2013.
[10] A. Vidal, J. J. Marron, and M. A. Labrador, "Real-time pedestrian tracking in indoor environments," in *Proc. of IEEE LATINCOM*, 2014.
[11] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, "No need to war-drive: unsupervised indoor localization," in *Proc. of ACM Mobisys*, 2012.
[12] C. Fischer, P. Talkad Sukumar, and M. Hazas, "Tutorial: implementation of a pedestrian tracker using foot-mounted inertial sensors," *IEEE pervasive computing*, 2013.
[13] H. Fourati, "Heterogeneous data fusion algorithm for pedestrian navigation via foot-mounted inertial measurement unit and complementary filter," *Instrumentation and Measurement, IEEE Transactions on*, 2015.
[14] W. Li, Y. Hu, X. Fu, S. Lu, and D. Chen, "Cooperative positioning and tracking in disruption tolerant networks," *Transactions of IEEE Parallel and Distributed Systems*, 2015.
[15] F. Zampella, M. Khider, P. Robertson, and A. Jiménez, "Unscented kalman filter and magnetic angular rate update (maru) for an improved pedestrian dead-reckoning," in *Proc. of IEEE PLANS*, 2012.
[16] "Snake game," https://en.wikipedia.org/wiki/Snake_(video_game).