Robust Inertial Motion Tracking Through Deep Sensor Fusion Across Smart Earbuds And Smartphone

JIAN GONG [*], School of Computer Science and Engineering, Central South University, ChangSha, China; Department of Electrical and Computer Engineering, University of California San Diego, San Diego, United States
XINYU ZHANG, Department of Electrical and Computer Engineering, University of California San Diego, United States
YUANJUN HUANG, Department of Electrical and Computer Engineering, University of California San Diego, United States
JU REN[†], School of Computer Science and Engineering, Central South University, China
YAOXUE ZHANG, Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China; School of Computer Science and Engineering, Central South University, ChangSha, China

ABSTRACT

IMU based inertial tracking plays an indispensable role in many mobility centric tasks, such as robotic control, indoor navigation and virtual reality gaming. Despite its mature application in rigid machine mobility (*e.g.*, robot and aircraft), tracking human users via mobile devices remains a fundamental challenge due to the intractable gait/posture patterns. Recent data-driven models have tackled sensor drifting, one key issue that plagues inertial tracking. However, these systems still assume the devices are held or attached to the user body with a relatively fixed posture. In practice, natural body activities may rotate/translate the device which may be mistaken as whole body movement. Such motion artifacts remain as the dominating factor that fails existing inertial tracing systems in practical uncontrolled settings.

Inspired by the observation that human heads induces far less intensive movement relative to the body during walking, compared to other parts, we propose a novel multi-stage sensor fusion pipeline called DeepIT , which realizes inertial tracking by synthesizing the IMU measurements from a smartphone and an associated earbud. DeepIT introduces a data-driven reliability aware attention model, which assesses the reliability of each IMU and opportunistically synthesizes their data to mitigate the impacts of motion noise. Furthermore, DeepIT uses a reliability aware magnetometer compensation scheme to combat the angular drifting problem caused by unrestricted motion artifacts. We validate DeepIT on the first large-scale inertial navigation dataset involving both smartphone and earbud IMUs. The evaluation results show that DeepIT achieves multiple folds of accuracy improvement on the challenging uncontrolled natural walking scenarios, compared with state-of-the-art closed-form and data-driven models.

1 INTRODUCTION

Inertial sensors (IMUs), including gyroscope, accelerometer and magnetometer, have been a standard module in modern mobile devices, and played a fundamental role in many ubiquitous computing applications. Example use cases include tracking users' trajectory in VR gaming [37], navigating a user in GPS denied indoor environment [14], serving as a low-power fitness tracker [40], complementing GPS in last-mile scenarios [34], *etc.* The underlying algorithms, often referred to as inertial odometry [18] or dead reckoning [17], aim to estimate the moving direction and distance of the user, relative to the earth's coordinate, *i.e.*, the global reference frame (GRF).

Although inertial motion tracking has seen wide adoption in air, ocean and robotic navigation, it remains a fundamental open problem for daily mobility scenarios involving human users carrying IMU-equipped mobile/wearable devices. IMUs can only sample the linear acceleration, angular velocity, and magnetic flux density within the devices' local coordinate, *i.e.*, local reference frame (LRF). Ideally, an integration of the gyroscope leads to estimation of angular rotation, and double integration of accelerometer reading estimates radial distance change, within the LRF [32]. However, the LRF motion must be projected to the GRF, in order to estimate the user's true location/direction change. The projection essentially offsets a rotation matrix, which represents the device's 3D orientation (pose) relative to the global coordinate. The gravity (*i.e.*, acceleration towards the ground) and magnetic

[*]This work was conducted when the author was a visiting scholar at the University of California San Diego.
[†]The corresponding author.

Authors' addresses: Jian Gong, School of Computer Science and Engineering, Central South University, ChangSha, China;, Department of Electrical and Computer Engineering, University of California San Diego, San Diego, United States, gongjian@csu.edu.cn; Xinyu Zhang, Department of Electrical and Computer Engineering, University of California San Diego, San Diego, United States, xyzhang@ucsd.edu; Yuanjun Huang, Department of Electrical and Computer Engineering, University of California San Diego, San Diego, United States, yuh036@eng.ucsd.edu; Ju Ren, School of Computer Science and Engineering, Central South University, ChangSha, China, renju@csu.edu.cn; Yaoxue Zhang, Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China;, School of Computer Science and Engineering, Central South University, ChangSha, China, zyx@csu.edu.cn.

north can be used as references for the earth's vertical direction and horizontal plane, respectively. For objects that are static or undergo smooth/rigid mobility (*e.g.*, ground robots or aircrafts), the gravity involves a constant vector with known magnitude ($9.8\text{m}/s^2$), which can be easily extrapolated.

Unfortunately, to track human mobility with on-body IMUs, the sensor readings are intrinsically noisy. The motion noises are generated unconsciously and inevitably, either by limb swinging/shaking during normal walking or by ordinary maneuvers such as phone rotation or moving into pocket. Such motion artifacts are indistinguishable from walking-induced IMU dynamics, and the resulting error will be further magnified by gyroscope or accelerometer integration, resulting in exponential error propagation over a walking trajectory. In other words, it is the irregular motion artifact itself that plagues inertial motion tracking in practice. Recent research has investigated various mechanisms to periodically calibrate the orientation tracking, *e.g.*, leveraging the zero-acceleration moments for foot-mounted IMUs when the foot touches the ground [16], opportunistically recalibrating when the IMUs become static [2], or reducing the motion degree of freedom by attaching the IMUs to forearms [33]. However, the underlying assumptions tend to break in daily ambulant conditions, where the mobile device can arbitrarily change its orientation and position relative to the user body while walking.

The statistics of IMUs' intrinsic noise vary drastically across different usage scenarios. Obviously, it is infeasible to derive a general closed-form framework that deals with the elusive noise. The past two years have seen a trend towards data-driven approaches [3, 43, 44]. At a high level, machine learning algorithms can establish an implicit representation of the noise distribution along with the relationship between IMU measurements and location/angle offset. These algorithms often take variants of recurrent neural network (RNN) as the backbone structure, which captures the temporal continuity of moving trajectory given the input sensor sequence [39]. However, these approaches do not explicitly model the device orientation. So the motion noise can still contaminate the location estimation, causing severe drift over time. In fact, the training/testing data of these systems are collected assuming the user's smartphone has a stable orientation (held steady facing upwards or in-pocket), which are incommensurate with many real-world use cases.

In this paper, we propose deep inertial tracking (DeepIT), a deep learning based sensor fusion framework to enable truly robust inertial motion tracking in the wild. Our key insight is simple: To combat the motion noise, we need to incorporate additional IMU sensors whose measurements are not likely to be contaminated, at least not simultaneously with the smartphone's IMU measurements. We realize such sensing diversity using smart earbuds, which are emerging as a new generation of wearable devices. In many of the desired use cases of inertial tracking, such as indoor navigation and outdoor run tracker, the earbuds are accompanied by the user's smartphone[1]. Smart earbuds are far less susceptible to noise pollution than hand-held smartphones, because head rotation occurs much less frequently and at much smaller scale compared with the body limbs [27].

Our DeepIT model segments the input IMU sequences into small windows, and outputs the polar coordinate representation of the user's location. To effectively harness the complementary capability of the smartphone and earbuds, we design a *reliability network* which gauges the confidence of sensing from each IMU, *i.e.*, which IMU is suffering less from motion artifacts momentarily. The reliability network then forwards its output to an attention module, which fuses the two IMUs' readings based on their reliability. Observing that the IMU data is a continuous stream of samples; and motion artifacts, despite burstiness, do exhibit short-term correlation. Therefore, we cast the reliability network design as a sequence learning problem and construct it based on the LSTM model.

Even with sensor fusion, the angular and radial components of localization are still susceptible to cumulative error caused by the summation of each window's offset. Whereas the drift of radial component cannot be compensated by additional measurement, the angular counterpart can be calibrated by magnetometer reading which is not susceptible to accumulative error. The noise level of both IMUs, represented by reliability vector, is adopted as a metric to determine how much the magnetometer reading of the phone's IMU can be relied on. Based on the value of reliability vector, the magnetometer reading is then fused with angular component determined by the neural network for a more accurate angular offset estimation.

Aside from the dynamic motion noise, we found that existing state-of-the-art DL based trajectory estimation systems [3, 43] also face an *angle distribution inconsistency* problem in practical usage scenarios. When the IMU devices are placed with largely different angles during training and testing, the model accuracy drops significantly. This is because the angle distribution rules learned in the training data do not match the testing cases. A straightforward solution is to collect training data to cover all possible angular distributions, but this is infeasible due to the massive dimensions of angles in the 3D space. We address this problem by preprocessing the original IMU data and transforming them into *virtual IMU (VIMU)* samples, which eliminates the impacts of angle distribution by projecting the IMU data in a unified space anchored by the gravity.

We have implemented the DeepIT deep learning pipeline based on PyTorch. Our IMU sensing data are collected from real users

[1]While it is unsafe to wear earbuds on a street with heavy traffic, most smart earbuds have an ambient hear-through mode to make the user aware of the surroundings.

carrying Android phones while wearing eSense [11], a commercial smart earbuds device that exposes a gyroscope/accelerometer interface. The data set covers a wide range of scenarios including indoor/outdoor, different walking gaits/postures, various trajectory lengths, smartphone hardware models and ways of device attachment (handheld, in-backpack), *etc.* We have also implemented two state-of-the-art baseline systems (*i.e.*, MUSE [33] and IONet [3]). Our sanity check confirmed that these systems work well in simple walking scenarios where the smartphone/earbuds device is attached to the user body with a fixed posture, which is consistent with the original publications. However, in more natural and practical scenarios with occasional phone/head rotation, their performance drops dramatically due to motion noise. In contrast, DeepIT remains robust across across all the testing scenarios, achieving meter-level accuracy when normalized over time/distance. For the unnormalized cumulative tracking performance, DeepIT keeps the error to a few meters for trajectory areas covering a few thousand m^2, and 10 to 20 m even for larger trajectories covering more than 10k m^2. In contrast, IONet and MUSE show 50 to 200 m of cumulative error in the same scenarios.

In addition, DeepIT demonstrates strong generalization capabilities across all the settings. Even though the training is conducted on completely different trajectories/environment/users/attachments, it still maintains a consistent level of accuracy during testing.

To our knowledge, DeepIT represents the first data-driven framework to explicitly address the challenging motion noise problem in inertial tracking. Our main contributions can be summarized as follows:

- We design the first IMU sensor fusion framework that harnesses the complementary sensing capabilities of smart earbuds and smartphones. Although recent work employed earbuds for step counting [27], DeepIT is the first to use them for distance and direction estimation.
- We design a reliability estimation network which tracks down the motion noise contamination on each IMU. This design enables DeepIT 's inertial tracking to be applied in the wild, which involves relative motion between the phone/earbuds and human body during walking. Furthermore, we propose a reliability aware magnetometer calibration scheme that enables DeepIT to sense the absolute rather than relative direction, even in challenging scenarios with ferromagnetic disturbances. In addition, we propose a VIMU mechanism to resolve the angular distribution inconsistency problem which has plagued existing data-driven inertial tracking systems.
- We design and implement a data collection pipeline which synchronizes the samples from the phone, earbuds and ground truth. Our data set covers around 214k ft of trajectories and spans a wide range of practical settings which is amenable for verifying model generalization. Our experiments have validated the superior performance of DeepIT compared with state-of-the-art baselines.

The source code and dataset generated from DeepIT are available online to promote further research[2].

2 A PRIMER ON INERTIAL TRACKING: HEURISTIC PRIOR VS. DATA-DRIVEN PRIOR

Before delving into the DeepIT design, we first introduce the two mainstream solutions in inertial tracking, which build on data-driven prior knowledge and heuristic observations of motion sensor properties, respectively.

IONet: data-driven inertial tracking. IONet (AAAI'18) [3] represents the state-of-the-art inertial localization solution with data-driven priors. The key idea is to break the accelerometer/gyroscope sampling sequence into small windows and process them separately to mitigate error propagation. The change in location displacement is computed over each independent window of n time samples: $\Delta L = nv(0)dt + [(n-1)s_1 + (n-2)s_2 + \cdot + s_{n-1}]dt^2$, where $v(0)$ represents the initial velocity. s_i is the acceleration sample i within the GRF, which can be expressed as: $s_i = \mathrm{C}_b^n(t-1)\mathrm{a}_i - \mathrm{g}$. Here a is the acceleration measured in the LRF. C_b^n denotes the rotation matrix from the LRF to the GRF. g is the gravity vector.

IONet uses an LSTM model to capture $L(\cdot)$. The input are the linear and angular acceleration samples from a smartphone IMU, and the states of velocity and gravity are learnt by the LSTM implicitly. This method improves on the problem of inertial drift under non-periodic motion, *e.g.*, when the IMUs stay in a shopping trolley and baby-stroller, which cannot be easily handled by closed-form models. However, the underlying model works well *only when no rotational motion noise exists.* Its performance deteriorates rapidly when applied under more complicated scenarios involving random rotations, *e.g.*, when the IMUs are handheld, swinging, and occasionally move close to chest/face positions. These complications also lead to the aforementioned angular distribution inconsistency between training and testing data. The resulting estimation error will grow quickly towards the end of each window as more rotational vectors are multiplied together, leading to large cumulative drift over a long moving trajectory. In effect, although IONet demonstrated reasonable tracking accuracy, the validation experiments are conducted mostly in controlled scenarios, where the smartphones are keeping steadily in the subjects' hand or on trolley. When relative motion occurs between smartphones and subjects, even occasionally, IONet would show large angle deviations, which will become more evident in our experiments (Sec. 7).

[2]Project: *https://github.com/jamesdeep/DeepIT*

MUSE: A closed form model. MUSE (MobiCom'18) [33] is a representative closed-form inertial odometry algorithm. Its key idea lies in two parts: an improved 3D orientation estimation, and jointly tracking orientation and location using a particle filter. MUSE first determines the initial vertical tilt of the device using unpolluted (static) gyroscope reading. The 3D magnetic vector in LRF can then be projected to the GRF using the initial 3D orientation matrix ($O(t_0)$): $N^G = O(t_0)N^L(t_0)$, where N^L and N^G are the 3D magnetic North vectors in LRF and GRF, respectively; $O(\cdot)$ is a 3×3 transformation matrix that rotates the object's GRF to LRF. The model then combines the orientation obtained from gyroscope and magnetometer using a complementary filter. The intuition here is that gyroscope integration tends to be accurate over short-term, whereas magnetometer is stable over a long-term despite occasional ferromagnetic disturbances.

MUSE jointly estimates the location and orientation using a particle filter, particle is defined as a vector composed of three recent location updates with an angle that models the orientation drift (δ) around magnetic North vector at the previous time. On the other hand, the acceleration can also be approximated using the orientation estimated by magnetometer and gyroscope.

Ideally, the global accelerations calculated from the 3 consecutive locations and from the particle's orientation projection should have little difference. The particle filter is then resampled, and its weight is modeled as a zero-mean Gaussian probability density function on the difference. In order to constrain the state estimation on valid predefined location space, MUSE introduced an arm motion model (AMM) that reduces the LRF's spatial degree of freedom (DoF) from 6 to 5. However, *when arbitrary rotation occurs between the phone and the user body, the AMM model's assumption breaks.* The particle filter architecture cannot precisely model such rotation noise as it may not even follow a tractable distribution. Consequently, MUSE suffers from severe angle errors under rotational noise. This limitation will be verified in our field tests (Sec. 7).

3 OVERVIEW OF DEEPIT

The diagram in Fig. 1 illustrates the architecture and workflow of DeepIT , which consists of 4 elements. **(i) Estimating translation and angle increment feature.** The model estimates the translation increment feature and angle increment feature from the IMUs (accelerometers and gyroscopes) on the smartphone and earbuds respectively, based on 4 independent trajectory LSTMs (2 l-LSTMs and 2 d-LSTMs). Although earbuds always come in pairs, the left and right ones are highly correlated. We thus only use one of them to reduce the processing load on the learning model. **(ii) Sensor fusion.** We design a deep sensor fusion model to overcome the motion noise that plagues the smartphone and earbuds. The fusion model assesses the reliability of the sensors based on a customized reliability LSTM (r-LSTM), and fuses the translation and angle increment features using two attention models. **(iii) Angle drifting compensation.** Since motion noise incapacitates the existing orientation sensing methods, we design a reliability based drifting compensation scheme to stabilize the angle estimation over long trajectories. **(iv) Reconstructing the complete trajectory.** By concatenating the refined trajectory increments of each window, we can derive the complete trajectory estimation with high accuracy.

4 ATTENTION BASED FUSION MODEL

4.1 VIMU: Transforming the IMU Samples to Overcome the Angular Distribution Inconsistency

The accelerometer measurement is a 3-element vector that reflects the walking acceleration along the x, y and z direction of the device's body frame (*i.e.*, LRF), and can estimate the translation by quadratic integral. The gyroscope measurement reflects the angular velocity around the x, y and z axis of the LRF, which can estimate the angular change of device after a first order integration. Ideally, if the device has a fixed pose relative to the user body (*i.e.*, the LRF and GRF has a stable offset), then by combining the translation and angular offset, we can infer the walking distance and direction and hence the complete trajectory. At a high level, DeepIT follows this principle, using an LSTM model (l-LSTM in Fig. 1) to estimate the translation from the accelerometer, and another LSTM model (d-LSTM in Fig. 1) to estimate the moving direction from gyroscope.

State-of-the-art DL based trajectory estimation models [3, 43, 44] usually divide the input data sequences to independent windows and estimate a trajectory increment for each. This method forces the model to focus on learning the correlation within a small time window (*e.g.*, 2 seconds). Specifically, it uses LSTM model to map the input window data with 6 dimensions (3 axes of gyroscope and 3 axes of accelerometer) to the estimated translation and angle *increment*. One obvious *drawback* is that it suffers large deviation when the training data and testing data have different angle distributions. For example, when the smartphone is facing upwards during training and facing left with 90° during testing, the model would not work properly (as shown in Sec. 7.2). One may consider collecting data exhaustively in all possible angles to address this problem. However, the 3D angle distribution is a huge space which is impractical to cover fully. Therefore, existing implementation [3] only covers a narrow set of angular distributions in their training/test cases (*e.g.*, phone always facing upwards).

We propose a *virtual IMU (VIMU)* mechanism to overcome this limitation. We leverage the fact that people typically make turns in the horizontal plane (*i.e.*, x-y plane), which is perpendicular to the direction of gravity. Therefore, to estimate the turning angles,

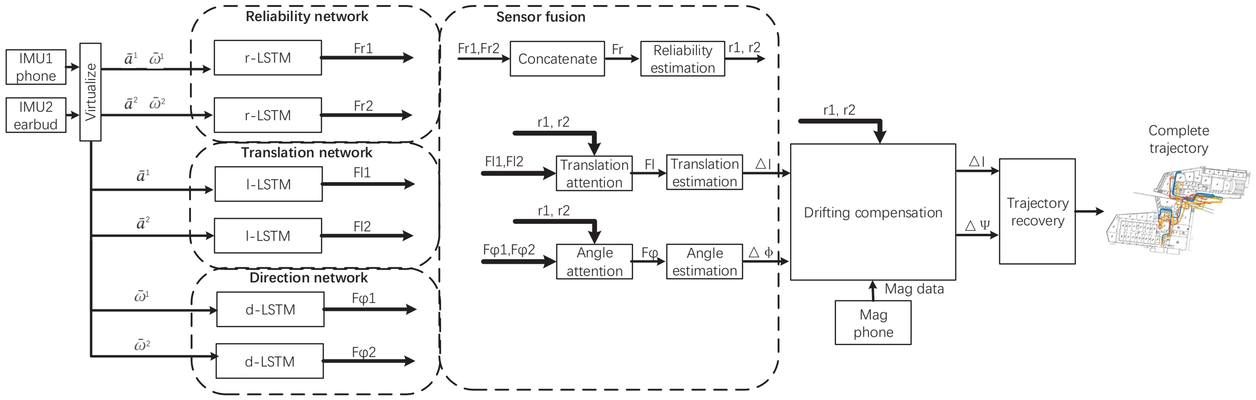

Fig. 1. Overall architecture of the DeepIT . The modules wrapped by dotted boxes are implemented by DL models.

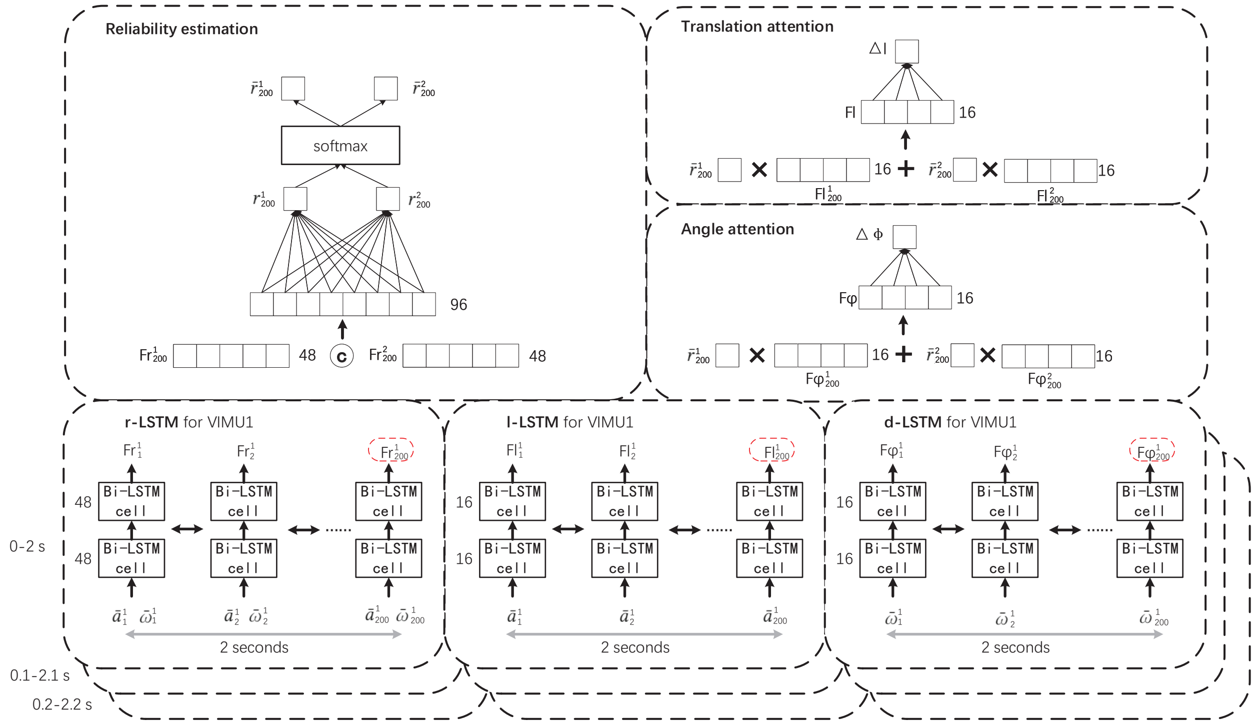

Fig. 2. Detailed structure of the DL parts of the DeepIT .

we only care about the angular speed within the x-y plane, which we denote as virtual angular speed (VAS) $\bar{\omega}$. The $\bar{\omega}$ can be derived projecting the gyroscope data on the gravity direction: $\bar{\omega} = \frac{\omega \cdot a}{\|a\|}$, where a and ω represent the original 3D accelerometer and gyroscope samples, respectively.

Similarly, we convert the acceleration samples to virtual acceleration (VAC) $\bar{a}$ to reduce the impacts of IMU angle. It is well known that the accelerometer measures the summary of gravity and IMU motion. Additionally, existing works have shown that human walking results in a dominating sinusoidal pattern in acceleration [43]. Therefore, the 2-norm of acceleration contains the necessary information which can be used to infer the walking speed. We thus define the VAC as: $\bar{a} = \|a\|_2$. Since the accelerometer measures acceleration on 3 orthometric axes, the summary of gravity and IMU motion has different distributions on the 3 axes measurements when the IMU is placed in different angles. The 2-norm of acceleration eliminates such inconsistency and results in a unified distribution on arbitrary angles.

The VIMU data is comprised of VAS and VAC. Compared to the raw IMU data, the VIMU data have consistent angle distribution in any pose. This is because the VIMU data leverages the gravity as an anchor to calculate the reference coordinate, which is immune to the pose variation. Therefore, it can bridge the gap between the angular distributions of training and testing data. We note that VIMU assumes the accelerometer can faithfully record the gravity direction during walking. However, this assumption breaks when large motion noise occurs. The inaccuracy of gravity estimation appears mostly when the sensor is enduring motion noise. Therefore, it is natural for the proposed reliability attention to lower the fusion weights of the noisy sensor, and deriving gravity primarily from the stable sensor.

The smartphone and earbuds have independent trajectory estimation from their own VIMUs. For each input window, the l-LSTM takes a VAC sequence of length T as input. The output is translation feature Fl, rather than the translation value itself, to prepare of the feature fusion later. The d-LSTM learns to map the input VAS sequence of length T to angle increment feature $F\varphi$. The d-LSTM and l-LSTM produce results for each input time stamp within a window, but we only use the last output which represents the cumulative result across the entire window.

4.2 Reliability Aware Attention Fusion Model

DeepIT 's sensor fusion model aims to synthesize the estimations from multiple sensors and automatically adjust their fusion weights according to their motion noise levels, in order to mitigate the impacts of the corrupted sensor. For example, when the smartphone's IMU data is polluted by rotational/translational noise, the model automatically resorts to the stable earbud as the main data source for estimation. Although occasional head rotation incurs motion noise as well, this rarely happens simultaneously with phone rotation. Hence a proper fusion model should be able to harness such complementary sensing properties to improve reliability.

Reliability network. The reliability network in DeepIT aims to distinguish the regular body moving patterns and the motion-noise contaminated patterns. As shown in existing studies [33], regular human walking leads to periodic patterns in gyroscope/accelerometer measurements when these sensors are rigidly attached to the body (*e.g.*, on arm or foot). The accelerometer data shows sinusoidal variation while walking, and gyroscope shows tractable patterns while the pedestrian is making a turn.

On the other hand, the subject may engage in unexpected daily activities that bring additional motion on the IMU sensor, such as large-scale motion (swinging smartphone in hand), aperiodic motion (taking up smartphone to make phone calls), or the combination of them, while walking, thus inducing motion noise. In such practical usage scenarios, the accelerometer and gyroscope measurement can be modeled as $a = a^* + e_a$ and $\omega = \omega^* + e_w$, where e_a and e_ω represents the motion interference, whereas a^* and ω^* are the components contributed by walking itself. If the a^* is close to a, it means that a^* is more likely to faithfully reflect the walking trajectory. Otherwise, it is unreliable. We use the Euclidean norm to represent the similarity reciprocal between a and a^*. Then, the reliability metric r is defined as the combination of similarities between a^* and a, and between ω^* and ω:

$$r = 1/\big(\|a - a^*\|_2 + \|\omega - \omega^*\|_2\big) \tag{1}$$

Since the motion artifacts normally last for a period of time, estimation of the IMU reliability can be cast to a sequence mapping problem, which is solved by the reliability LSTM (r-LSTM) in DeepIT :

$$Fr_t^i = R(x_t^i), x_t^i = (\bar{a}_t^i, \bar{\omega}_t^i), \tag{2}$$

where $R(\cdot)$ represents the r-LSTM function; Fr_t^i denotes the reliability feature of IMU sensor i. All the reliability feature are then concatenated as Fr and inputted to a fully connected layer to predict the reliability r_t^i for each sensor i.

Sensor fusion. DeepIT 's fusion scheme builds on a customized attention model. Classical attention mechanism [38] has been well recognized in natural language processing, and demonstrated its ability to jointly process different data segments (*e.g.*, different phrases in a sentence), or process the intermediate outputs from different neural layers. It uses a score function to calculate the weighted coefficients, which is implemented by a feed-forward network. However, the feed-forward network is not suitable for inertial tracking because it is unable to model the temporal properties of the motion sequence. In DeepIT , we redesign the score function using the r-LSTM in Equation (2). Since the original reliability value r_t^i is unnormalized, we can use the softmax layer in attention model to enforce normalization:

$$\bar{r}_t^i = \frac{exp(r_t^i)}{\sum_{i=1}^{M} \exp(r_t^i)}. \tag{3}$$

where i represents the sensor index; $\bar{r}_t^i$ is the fusion coefficient of sensor i. Then, the translation increment feature Fl_t^i and direction increment feature $F\varphi_t^i$ are fused according to sensor reliability:

$$Fl_t = \sum_{i=1}^{M} \bar{r}_t^i * Fl_t^i,\ \ F\varphi_t = \sum_{i=1}^{M} \bar{r}_t^i * F\varphi_t^i, \tag{4}$$

where M represents the total number of IMU sensors. Finally, the fused translation increment feature Fl_t and fused direction increment feature $F\varphi_t$ are inputted to two fully connected layers to derive translation increment Δl_t and direction increment $\Delta\phi_t$, respectively.

The value of reliability r of the IMU which endures motion noise is less than another IMU, so the noisy IMU data has less impacts on the fused feature, thus reduces the influence of motion artifacts. One may ask how can the r-LSTM exactly learn to adjust the reliability of a sensor corresponding noise level. The answer is that the noisy sensor induces noisy data, which makes the training of corresponding l-LSTM and d-LSTM not able to converge. In order to make the whole model able to converge, the only way is to to decrease the corresponding reliability r to lower the impacts of the noisy data on the results, which is achieved by the back propagation scheme itself.

Loss function. The optimal parameters p^* of the proposed fusion model can be derived by minimizing a loss function on the training data set $\mathbf{D} = (\mathbf{X}, \mathbf{Y})$: $p^* = \arg\min\{\mathcal{L}(f(\mathbf{X}), \mathbf{Y})\}$. We define the loss function $\mathcal{L}$ as the sum of Euclidean distances between the ground truth $(\Delta\bar{l}_t, \Delta\bar{\phi}_t)$ and estimated value $(\Delta l_t, \Delta\phi_t)$:

$$\mathcal{L} = \sum \|\Delta l_t - \Delta\bar{l}_t\|_2^2 + \|\Delta\phi_t - \Delta\bar{\phi}_t\|_2^2. \tag{5}$$

5 RELIABILITY AWARE ANGLE DRIFT COMPENSATION

Angle drifting due to IMU sensor noise is well recognized as the key challenge for all inertial odometry systems [25, 33]. Existing methods have attempted to address this problem by using magnetometer as an angle anchor, facilitated by adaptive filters [24] to compensate the drifting. However, when disturbed by motion noise (*e.g.* user arbitrarily rotating the phone by hand), the mag anchor points to non-walking directions, which misleads the trajectory estimation, worsens the angle drifting and hence the localization accuracy.

To meet this challenge, we design a reliability aware angle compensation method. One caveat with this design choice is that commercial smart earbuds are not equipped with any magnetometer [11]. The reason lies in the small package of earbuds. If it is ever needed, the magnetometer has to be placed very close to the earbud's speaker, which itself has a magnet and causes severe interference to the magnetometer [12]. Therefore, we have to handle the motion noise-induced angular drifting using the magnetometer on the smartphone. Besides motion noise, the magnetometer may suffer from low reliability due to occasional ferromagnetic interference from ambient environment, which must be accounted for in the angular compensation mechanism.

5.1 A Primer on Magnetometer Compensation

Consider the case when the magnetometer is parallel to the horizontal plane of the LRF. The yaw angle ψ can be represented by the magnetic field strength B:

$$\psi = \arctan \frac{B_y}{B_x}, \tag{6}$$

where B_x and B_y are components of the earth magnetic field vector, which can be read from the magnetometer.

In reality, the magnetometer is not guaranteed to be parallel to the local horizontal plane. Due to the angular deviation, we need to use the accelerometer to get another 2 angles: roll and pitch, *i.e.*, rotation around the x-axis and y-axis of the LRF. By definition, the roll θ and the pitch γ can be calculated as:

$$\theta = \arctan \frac{a_y}{a_z}, \quad \gamma = \arctan - \frac{a_x}{\sqrt{a_y^2 + a_z^2}}, \tag{7}$$

where the a_x, a_y and a_z are components corresponding to the acceleration of gravity along the three axes. Thus, the yaw angle ψ can be formulated as:

$$\psi = \frac{B_x \sin(\theta)\sin(\gamma) + B_y \cos(\theta) - B_z \sin(\theta)\cos(\gamma)}{B_x \cos(\gamma) + B_z \sin(\gamma)}. \tag{8}$$

To apply angular compensation to the accelerometer's estimation, complementary filter based methods are often used [33]. Given the yaw angle increment $\Delta\phi$, we can calculate the absolute angle at time t as:

$$\phi_t = \phi_{t-1} + \Delta\phi_t, \tag{9}$$

And then we calculate the angle of magnetometer anchor ψ_t following Equation (8). Finally, the compensated angle $\hat{\psi}_t$ should be:

$$\hat{\psi}_t = k\phi_t + (1-k)\psi_t, \tag{10}$$

where k is the compensation coefficient, which can be deemed as a boundary weight assigned between the sensor fusion model and the magnetometer anchor. k should be set close to 1 to prioritize latest samples. We fix it to $0.95 - 0.98$ in our evaluation.

5.2 Reliability Aware Drifting Compensation

The aforementioned classical magnetometer compensation method bears one major limitation: It does not consider the situation when the phone's angle may vary relative to the user body, which usually happens due to motion noise.

In our experiments, we find that the reliability output from r-LSTM varies according to the duration of the noise in the input data window. The longer the motion artifacts last, the lower the reliability. Thus, for a fixed time window, we can use the reliability network (*i.e.*, r-LSTM) to identify whether the smartphone is enduring motion noise or relatively stationary. The same mechanism can identify reliability drop due to temporary ferromagnetic interference. We introduce a new parameter called *interference ratio*

p_n, which represents the degree of interference in the current input data window for each device. It can be formulated as:

$$p_n = \frac{r_{stable} - r_t}{r_{stable}}. \tag{11}$$

where a_{stable} represents the reliability output without motion noise. Both motion noise and magnetic interference usually occur in a bursty manner. So we can approximate the stable value r_{stable} by averaging the reliability outputs. A large p_n implies that the phone is enduring severe noise/interference, and thus mag compensation should be suspended. We use an empirical threshold p_{Th} to identify such cases. Finally, we can update the compensated angle $\hat{\psi}_t$ as:

$$\phi_t = \hat{\psi}_{t-1} + \Delta\phi_t, \tag{12}$$

$$\hat{\psi}_t = \begin{cases} k\phi_t + (1-k)\psi_t & p_n > p_{Th} \\ \phi_t & p_n \le p_{Th}. \end{cases} \tag{13}$$

6 SYSTEM IMPLEMENTATION

Dataset. There is no public dataset for inertial tracking using both smartphone and earbuds simultaneously. Therefore, we collect our own dataset with pedestrian carrying Android phones and wearing smart earbuds. We use the eSense [11], an experimental smart earbuds device developed by Nokia, which can stream gyroscope and accelerometer readings through BLE to an associated smartphone. Our dataset covers a comprehensive set of real-world usage settings. To examine the performance of DeepIT and benchmark solutions under different levels of motion noise, the dataset incorporates 3 types of walking patterns: normal walking (NW), walking while swinging phone occasionally (RP), walking while swinging phones and moving heads alternatively (RPE), walking while moving heads occasionally (HM). The RPE represents practical scenarios where the user browses the smartphone while walking on the street. Additionally, 5 different users and 4 different smartphone models are involved. The data are collected over 10 different places covering both indoor and outdoor. By default, the users are holding the smartphone in their hands, but the dataset also contains a scenario with phone in a backpack. The total length of dataset is over 30 hours (around 214k ft in length) with each single trajectory lasting 5-10 minutes. The ground truth data (location and walking direction) is collected by a Google Tango phone [13]. Tango uses visual inertial odometry techniques to achieve cm precision in motion tracking. Since its accuracy may be impacted by lighting condition and diversity of visual features, we collect all the data during daytime and in places with sufficient object diversity. In addition, we tie the Tango in front of the chest to keep its pose relatively stable. The Tango device, smartphone, and earbuds are synchronized using NTP time stamps. The app running on the smartphone can provide sufficient precision in synchronization with the earbuds. The app periodically sends data requests to the earbuds, which immediately returns the sensing data. Based on our tests, the average duration of the process starting from the request to the data capturing is less than 20 ms. We also manually observed the data features, e.g., plotting the gyroscope and accelerometer curves of earbud and smartphone, and found average error (distance between data features) is indeed within 20 ms, which is far less than the algorithm running window time (2s). Therefore, the synchronization should be sufficient and is unlikely to impact the model performance.

Evaluation metrics. We test the localization performance using 5 standard metrics widely adopted by inertial tracking and visual SLAM systems [35]. In addition, we propose 2 metrics to quantify the microbenchmark performance of DeepIT in comparison with baselines.

(i) Mean Trajectory Error (MTE). MTE calculates the RMSE of estimated location and ground truth location trajectories, *i.e.*,

$$MTE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|e_i\|_2^2}, \tag{14}$$

where e_i is the $i-th$ location error between the estimation and ground truth and n is the total number of location estimations. This metric reflects the absolute localization error of estimation and is prone to inflate as route length increases.

(ii) Time-normalized Mean Trajectory Error (T-MTE). The T-MTE is used to amortize the impacts of time during of trajectory on MTE. Additionally, early location errors make larger impacts on T-MTE, so by contrasting MTE and T-MTE we can evaluate the distribution of the errors.

$$T\text{-}MTE = 60 * \sqrt{\frac{1}{n-1}\sum_{i=2}^{n} \frac{\|e_i\|_2^2}{t_i}}, \tag{15}$$

where t_i is the time of $i-th$ location started from 0. Since the original numeric value of T-MTE is too small, we multiply it with the coefficient of 60 to make it have similar range to MTE.

(iii) Distance-normalized Mean Trajectory Error (D-MTE). D-MTE is used to counteract the impacts of the area covered by the trajectory on MTE. Compared with T-MTE, D-MTE is less sensitive to long trajectories because it is armortized by the trajectory

size, *i.e.*,

$$D\text{-}MTE = 100 * \frac{MTE}{x_{max} - x_{min} + y_{max} - y_{min}}, \tag{16}$$

where x_{max} and x_{min} are the maximum and minimum x-axis values of ground truth trajectory, and y_{max} and y_{min} are the maximum and minimum y-axis values of ground truth trajectory. Since the original numeric value of D-MTE is too small, we multiply it with a coefficient of 100 to make its range similar to MTE.

Besides the above metrics, we need to separately determine the angle errors and translation errors, so as to track down the reasons for inaccurate location estimations. To this end, we propose 2 extra metrics as follows.

(iv) Total mean translation error e_l, computed by averaging the difference between estimated trajectory length and ground truth trajectory length.

$$e_l = \frac{1}{n}\sum_{i=1}^{n}(\bar{l}_i - l_i), \tag{17}$$

where $\bar{l}_i$ and l_i are the $i-th$ cumulative trajectory length from ground truth and estimation, respectively.

(v) Total mean angle error e_a in radian, which is the average difference between estimated angle and ground truth angle.

$$e_a = \frac{1}{n}\sum_{i=1}^{n}|\bar{\psi}_i - \hat{\psi}_i|, \tag{18}$$

where $\bar{\psi}_i$ and $\hat{\psi}_i$ are the $i-th$ angles from ground truth and estimation respectively.

Model implementation. Our sensor fusion model is comprised of the DL parts and non-DL parts. The DL parts include the reliability network, translation network, direction network and attention model. The detail structure of the DL parts are shown in Fig.2. The reliability networks include 2 r-LSTMs, each of which is comprised of 2 stacked bidirectional LSTM cells. These 2 LSTM cells both contain 48 hidden units. Each r-LSTM accepts the VAC and VAS sequence with length of 200 (corresponds to 2 seconds) as input, and outputs the reliability feature Fr_t^i with size of 48 (the last element from the output sequence). The l-LSTMs and d-LSTMs share the same structure, which is comprised of 2 stacked bidirectional LSTM cells. These 2 LSTM cells both contain 16 hidden units. The l-LSTM accepts the VAC sequence with length of 200 as input, and outputs the translation feature Fl_t^i with size of 16 (the last element from the output sequence). The d-LSTM accepts the VAS sequence with length of 200 as input, and outputs the angle feature $F\varphi_t^i$ with size of 16 (the last element from the output sequence). The attention model consists of a reliability estimation module, a translation attention module and an angle attention module. The reliability estimation module first concatenates the two reliability feature Fr_t^1 and Fr_t^2 and use a layer of fully connected layer and a softmax layer to estimate and normalize the reliability outputs $\bar{r}_t^1$ and $\bar{r}_t^2$. The two attention modules fuses the translation feature Fl_t^i and angle feature $F\varphi_t^i$ following Section 4.2.

We use Pytorch [28] to implement our fusion model and train it on a server with a cluster of Nvidia RTX 2080Ti GPUs. The whole DL parts of DeepIT are trained in an end-to-end manner. The training uses the Adam optimizer with learning rate of 0.0005, and inserts a dropout layer with rate 0.25 between each two LSTM cells to prevent over-fitting. The batch size is setting to 16 and the total training epoch is 100.

During the testing, we recover the complete trajectory by accumulating the translation increments Δl_t together with fused angle $\hat{\psi}_t$. The fusion model predicts the results for every 2-seconds time window with stride of 0.1 second. In other words, the trajectory is updated every 0.1 second.

As an ablation study, we implement 2 versions of DeepIT with and without using drifting compensation, which are referred to as *DeepIT w/ mag* and *DeepIT w/o mag*, respectively. The above 2 models use VIMU data as input. For VIMU ablation study, we also implement a special version which uses raw IMU data as input, denoted as *DeepIT raw-IMU w/ mag*. For single sensor ablation study, we implement DeepIT -earbud and DeepIT -phone, which only accept single sensor data as input. Compared with DeepIT , their r-LSTMs are removed.

Baseline methods. For comparison, we implement two baseline inertial odometry systems: IONet [3] and MUSE [33], which represent state-of-the-art data-driven and closed-form algorithms, respectively. We also implement DeepFusion [42], a representative data-driven multi-sensor fusion model.

IONet. IONet is a deep learning based inertial odometry system. The IONet accepts 6-D IMU data sequence (accelerometer and gyroscope) as input, which is normalized to $[-1, 1]$. It outputs translation increment and angle increment for every 2-second window. IONet uses bidirectional LSTM as basic structure, which is comprised of 2 bidirectional LSTM cells and an MLP layer. All the LSTM cells have the same hidden layer size of 96 and the MLP layer has an output dimensions of 2. Since the IONet code is not publicly available, we implement it following the description in the original paper [3, 4]. An Adam optimizer and learning rate of 0.0005 is used for training. We use Dropout layer in each LSTM cell with 25% dropping rate to prevent overfitting. The training/testing is applied to the smartphone IMU data and earbuds IMU data, respectively. To ensure our implementation matches the original IONet, we have tested our implementation on the published IONet dataset, and the results are almost identical to [3, 4]. However, note that our dataset contains more challenging trajectories — containing sharp turnings (180 degrees), covering larger areas (maximized to 18000 m^2), and involving uncontrolled user mobility (and hence motion noise) which is not handled by the IONet design. Therefore, it is expected that IONet's performance will degrade when trained/tested on our dataset. In order to make fair comparison, we also extend the IONet for fusing both smartphone and earbud, named IONet-fuse. Specifically, we fuse the outputs(translation increment and angle increment) from IONet-phone and IONet-earbud by weighted sum with equal weights for each window.

MUSE. MUSE is a filter based motion tracking pipeline originally designed to track the position of a subject with restricted movement (*e.g.*, human-arm motion restricted by elbow and shoulder joints). MUSE takes advantages of the constraint between IMU reading and object's location to reduce its valid search space in particle filter, resulting in high localization accuracy. We have implemented the core pipeline of MUSE, made of a complimentary filter and a particle filter. Since the constraint between localization and IMU reading no longer exists in general inertial tracking scenarios, we implemented additional constraints on object's moving speed (limited between 1m/s and 2m/s).

DeepFusion [42]. We use DeepFusion as a benchmark multi-sensor fusion model, to verify the effectiveness of our reliability-based fusion scheme. DeepFusion comprises several core fusion schemes such as *Weighted-Combination Module* and *Cross-Sensor Module*. We implement the *Sensor-Representation Module* using two IONet LSTMs with the final FC layers removed, for smartphone and earbud, respectively. The features output by the two LSTMs are connected in the form of the weighted-combination and cross-sensor combination, consistent with the original DeepFusion design [42]. Finally, the output features in *Weighted-Combination Module* and *Cross-Sensor Module* are fed into fully-connected layers and added up to generate the angle estimation and translation estimation.

We note that recently there emerged other DL based inertial navigation, such as RoNIN [43]. Similar to IONet, RoNIN assumes the smartphone is attached to the body without relative motion (such as swinging for a few seconds), and hence negligible motion noise. From this perspective, there is no fundamental difference from IONet.

7 EVALUATION

We first present the experiment results on the overall performance of our model involving different levels of motion noise. Then, we evaluate the model's generalization ability across a wide range of practical usage scenarios, *e.g.*, for different users, mobile devices, indoor/outdoor and ways of attachment. Finally, we also study the model's performance in other aspects, e.g., varying hyper-parameters of magnetometer compensation.

7.1 Performance Under Daily Mobility Scenarios

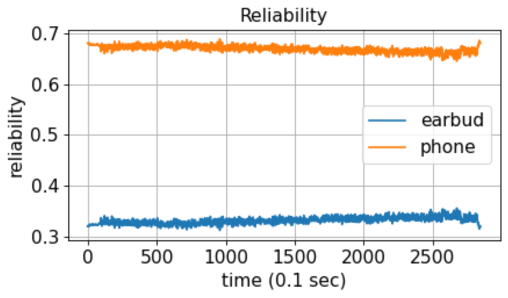

Fig. 3. Reliability output in NW scenario

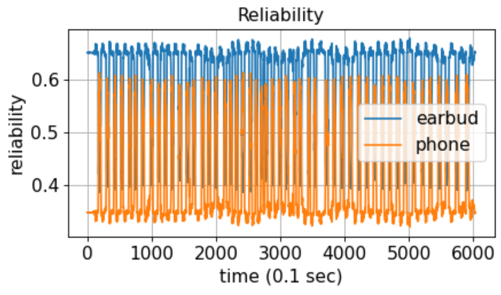

Fig. 4. Reliability output in RPE scenario

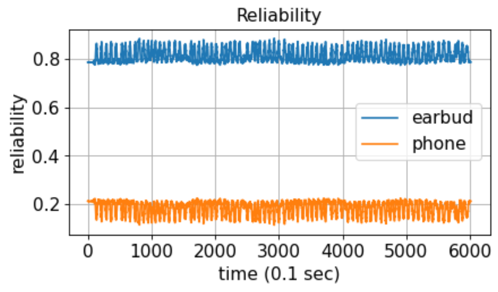

Fig. 5. Reliability output in RP scenario

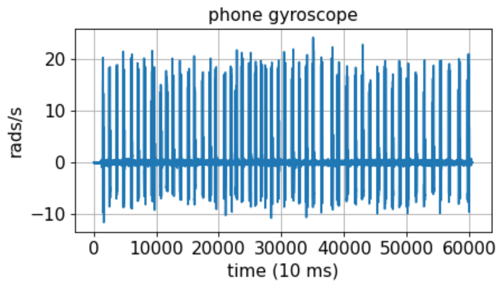

Fig. 6. Phone gyroscope output in RPE scenario

We first test DeepIT ' performance in 3 scenarios: normally walking (NW), walking with swinging phone casually (RP) and

Table 1. Performance comparison in NW scenario

model	Route1 (32 m^2)					Route2 (3000 m^2)					Route3 (12000 m^2)				
	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a
MUSE	33.327	11.436	2.885	\	\	43.343	9.755	0.381	\	\	56.370	20.697	0.154	\	\
IONet-earbud	1.532	0.790	0.097	0.990	0.389	25.207	5.337	0.238	5.054	1.649	50.257	17.084	0.162	2.214	0.546
IONet-phone	1.945	0.801	0.124	0.736	0.639	35.205	6.319	4.411	1.522	1.735	41.997	14.111	0.135	3.168	0.377
IONet-fuse	1.834	0.794	0.116	0.630	0.482	31.699	5.821	0.299	1.470	1.750	47.675	16.153	0.153	2.594	0.485
DeepFusion	1.441	0.708	0.087	0.782	0.430	25.694	4.682	0.242	13.185	1.356	29.337	11.316	0.094	3.691	0.168
DeepIT -earbud	1.550	0.792	0.101	0.895	0.412	24.197	4.983	0.215	4.188	1.357	34.412	11.659	0.118	3.867	0.479
DeepIT -phone	1.867	0.759	0.117	0.815	0.612	34.459	8.308	0.325	2.550	1.655	30.031	9.682	0.096	9.222	0.306
DeepIT w/o mag	0.855	0.354	0.054	0.468	0.293	21.416	4.263	0.202	2.376	1.525	16.319	6.534	0.053	1.956	0.118
DeepIT w/ mag	0.848	0.413	0.054	0.468	0.256	20.616	4.139	0.195	2.376	1.444	15.800	6.364	0.051	1.956	0.089

Table 2. Performance comparison in RP scenario

model	Route1 (32 m^2)					Route2 (3000 m^2)					Route3 (12000 m^2)				
	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a
MUSE	41.070	12.922	3.139	\	\	36.913	16.710	0.319	\	\	40.752	20.527	0.124	\	\
IONet-earbud	3.362	1.356	0.203	3.491	1.076	10.542	5.697	0.097	4.428	0.267	26.978	8.714	0.100	1.385	0.225
IONet-phone	5.658	2.822	0.342	8.007	1.462	29.859	10.234	0.274	11.169	0.919	58.128	19.570	0.215	1.831	0.455
IONet-fuse	3.786	1.691	0.229	9.839	0.943	18.669	8.512	0.166	13.518	0.303	42.251	13.971	0.156	1.751	0.333
DeepFusion	3.144	1.602	0.193	1.398	0.876	19.996	11.152	0.230	4.229	0.824	34.865	13.180	0.122	0.744	0.275
DeepIT -earbud	2.396	0.967	0.165	3.982	0.906	11.691	6.644	0.114	4.345	0.284	31.512	11.331	0.131	0.923	0.653
DeepIT -phone	5.980	2.937	0.373	3.088	1.467	27.585	9.808	0.264	13.464	0.843	53.087	18.244	0.196	1.418	0.399
DeepIT w/o mag	1.410	0.733	0.085	4.372	0.275	13.089	4.617	0.120	3.714	0.661	26.723	8.875	0.099	0.653	0.166
DeepIT w/ mag	0.734	0.428	0.044	4.372	0.146	5.763	2.420	0.053	3.714	0.143	15.763	5.946	0.058	0.653	0.132

Table 3. Performance comparison in RPE scenario

model	Route1 (32 m^2)					Route2 (3000 m^2)					Route3 (18000 m^2)				
	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a
MUSE	38.243	27.384	3.006	\	\	81.563	32.367	0.697	\	\	154.418	46.579	0.428	\	\
IONet-earbud	4.273	2.464	0.333	6.187	1.704	21.638	7.654	0.208	1.464	1.022	237.368	71.051	0.803	3.652	1.791
IONet-phone	12.886	9.534	1.005	29.856	1.594	22.275	11.214	0.215	1.074	0.542	96.019	26.942	0.325	7.401	0.903
IONet-fuse	8.748	6.158	0.737	20.847	1.646	21.738	8.382	0.214	1.384	0.837	175.482	51.320	0.594	6.532	1.472
DeepFusion	7.069	3.767	0.551	2.755	1.758	27.799	9.863	0.260	1.017	1.169	64.925	20.368	0.219	5.182	1.051
DeepIT -earbud	7.503	3.582	0.585	2.814	1.451	28.420	12.975	0.273	1.165	0.684	116.469	35.568	0.393	6.007	1.396
DeepIT -phone	5.232	3.157	0.408	2.275	1.762	23.774	11.482	0.234	1.451	1.157	158.757	50.406	0.536	15.412	1.862
DeepIT w/o mag	2.754	1.524	0.215	4.291	0.469	15.215	5.900	0.147	0.275	0.531	34.750	11.442	0.117	3.315	0.323
DeepIT w/ mag	2.699	1.697	0.211	4.291	0.242	8.554	3.802	0.082	0.275	0.104	20.464	5.984	0.069	3.315	0.159

Table 4. Performance comparison in HM scenario

model	Route1 (32 m^2)					Route2 (3000 m^2)					Route3 (18000 m^2)				
	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a
MUSE	48.543	37.585	4.158	\	\	76.278	29.254	0.685	\	\	135.547	40.547	0.354	\	\
IONet-earbud	6.575	3.478	0.458	9.187	1.684	61.573	19.067	0.507	4.464	1.322	107.348	45.584	0.518	5.278	1.278
IONet-phone	3.894	1.820	0.251	4.573	1.857	28.478	10.482	0.257	2.277	1.673	31.275	10.964	0.118	3.478	0.428
IONet-fuse	5.278	2.167	0.398	6.365	1.781	45.548	14.562	0.363	3.982	1.518	64.731	25.280	0.335	4.548	0.808
DeepFusion	5.038	1.885	0.335	8.860	1.259	28.319	8.858	0.242	0.899	0.527	92.787	37.068	0.363	1.030	1.290
DeepIT -earbud	6.967	3.742	0.464	8.252	1.359	55.696	17.704	0.477	5.855	1.138	78.242	34.856	0.306	7.010	1.450
DeepIT -phone	3.677	1.778	0.245	5.135	1.770	30.608	11.515	0.262	3.518	1.718	26.818	9.723	0.104	4.411	0.665
DeepIT w/o mag	3.197	1.322	0.213	8.037	0.372	20.482	7.023	0.173	1.543	0.590	23.040	6.779	0.098	1.891	0.463
DeepIT w/ mag	2.578	1.157	0.186	8.037	0.257	11.576	4.158	0.086	1.543	0.285	15.675	4.687	0.079	1.891	0.169

walking while swinging both phone and earbud casually (RPE). The motion interference intensity increases from NW to RP and RPE. For each case, we evaluate 3 different routes, *i.e.*, route 1 to route 3, with increasing trajectory length and area coverage. For a fair comparison, we always train DeepIT and IONet using the same datasets, and similarly for testing. All the training and testing data in this experiment are collected by the same subject and same device. We will evaluate the generalizability in the next section.

The NW scenario is the baseline scenario used in existing inertial odometry methods [3, 33, 43], in which the subjects were asked to handhold their phones with a fixed pose while walking. The experimental results for NW are summarized in Table 1. We find that DeepIT w/ mag achieves the lowest D-MTE amongst all DL models, *i.e.*, 0.054, 0.195, and 0.051m, along route 1, 2, and 3 respectively. It is 1.5× lower than IONet-phone, 2× lower than IONet-earbud, 1.7× lower than IONet-fuse and 11× lower than MUSE on average. The reason behind its performance advantage is that both the translation and angle drifting are largely decreased by the dual-sensor feature fusion. Although IONet-fuse also leverages sensor fusion, the results show that the simple weighted sum based sensor fusion adopted by IONet-fuse performs much worse than DL based sensor fusion. As shown in the table, both the e_l and e_a of DeepIT are lower than the IONet models, indicating DeepIT 's ability to combine the merits of multiple sensors.

Notably, IONet achieves a reasonable MTE of 1 to 2m along route 1, which is consistent with the original paper [3]. However, it exhibits much worse performance of 40 to 50m along route 2 and 3. The reason lies in the larger turning angles along these routes. The turning angles in the IONet dataset [3] are usually within 90°, whereas routes 2 and 3 here contain close to 180° turning angles. The larger turning angles are easier to produce large motion noise and angle drifts, and thus worsens the localization accuracy.

We further examine the reliability network output of the smartphone and earbud and plot the results along one representative route in Figure 3. We can see that the reliability of the phone is always higher than the earbud in NW. This is because the phone endures clearer periodic walking patterns than the earbud, so its translation estimation is more accurate.

In the RP scenario, the subjects may swing the phone casually just as in practical daily walking, which incurs continuous relative motion between the phone and user body. This breaks the assumption of stable phone attachment in existing inertial odometry systems [3, 33, 43]. From the results in Table 2, we see that DeepIT w/ mag achieves the lowest D-MTE, amongst all DL models, *i.e.*, 0.044, 0.053, and 0.058m, along route 1, 2, and 3 respectively. This is around 5× lower than IONet-phone, 2× lower than IONet-earbud, 3.4× lower than IONet-fuse and 20× lower than MUSE on average. Additionally, we observe that the D-MTE improvement of DeepIT over IONet is larger than in the NW scenario. This is because DeepIT effectively assigns proper fusion weights according to the sensors' reliability and thus reduces the impacts of motion noise. From the corresponding reliability network output in Figure 5, we see a completely reverse pattern compared to the NW scenario. Here the earbud's reliability is always higher than the phone. This matches our expectation because the phone endures more motion noise in this scenario.

The subjects were asked to look forward while walking in the above two scenarios, so their heads (earbuds) remain in a relative stable posture relative to the body. The third scenario, RPE, is more challenging, as the subjects can swing both their phones and heads casually. The corresponding experimental results are shown in Table 3. We can find that DeepIT w/ mag consistently demonstrates superior performance, with almost 5× lower D-MTE than IONet-phone, 4× lower D-MTE than IONet-earbud, 4.5× lower D-MTE than IONet-fuse and up to 12× lower than MUSE, across all test routes. Although the earbud also endures motion noise in this scenario, the relative performance gain of DeepIT is almost unaffected. It indicates that DeepIT effectively adjusts the fusion weights according to the sensors' reliability.

Compare to the RP scenario, DeepIT 's own performance does degrade noticeably, which is due to the increase of motion noise levels on both sensors. The corresponding reliability network output (Figure 4) is more complicated than the previous 2 scenarios. Interestingly, the two sensors' reliability level change alternately. Comparing with the original IMU data within the same time frame in Fig. 6, we find that the lower reliability period precisely corresponds to the moments when the sensor endures motion noise. This indicates our reliability aware attention scheme faithfully tracks the motion noise intensity and adjusts the fusion weights accurately.

In realistic usage scenarios, the head motion is very common. We thus examine the model performance under the HM setup. The corresponding experimental results are shown in Table 4. We observe that DeepIT w/ mag still performs the best among all the models, with almost 1.5× lower D-MTE than IONet-phone, 2.5× lower D-MTE than IONet-earbud, 2× lower D-MTE than IONet-fuse and up to 22× lower than MUSE, across all test routes. In addition, the models with only earbud as input (DeepIT -earbud and IONet-earbud) show worse results than their corresponding smartphone competitors (DeepIT -phone and IONet-phone), which indicates that the complementary sensing capability of the smartphone is necessary to improve the overall model robustness.

In order to verify that the superior performance of DeepIT mainly comes from the reliability based fusion scheme rather than the additional capacity brought by multiple LSTMs, we implement and study the performance of DeepIT with a single sensor. The results are shown in Table 1-4. We draw three main conclusions. *(i)* The average D-MTEs of the DeepIT -earbud and DeepIT -phone are worse than DeepIT in all four scenarios, with 2.2× and 2.3× higher. It indicates that the reliability based fusion scheme is better than single-sensor models. *(ii)* In HM scenario, the average D-MTE of DeepIT -phone is 1.9× lower than DeepIT -earbud. In RP scenario, the average D-MTE of DeepIT -earbud is 2.3× lower than DeepIT -phone. It indicates that different sensors have their own advantages in corresponding scenarios, thus the fusion is a necessary choice. *(iii)* The overall performance of DeepIT -earbud and DeepIT -smartphone are similar to the corresponding IONet competitors. But the advantage of DeepIT single sensor version is distinct when there exists inconsistent data distribution, as shown in Section 7.2.

Another data-driven multi-sensor fusion model DeepFusion [42] is also compared in all four scenarios, with 1.5×, 1.4×, 1.9× and 2.7× higher D-MTE in scenario NW, RP, RPE and HM respectively. It indicates that the reliability based fusion scheme is more effective than the general sensor fusion scheme proposed in DeepFusion.

7.2 Generalization Capability

We now conduct a series of experiments where new scenarios, users and devices are involved in the testing stage to verify the generalizability of the DeepIT model.

Table 5. Performance of different models in indoor places, RP scenario

model	Indoor 1		Indoor 2		Indoor 3		Indoor 4 (strong mag interference)		Indoor 5		Indoor 6	
	D-MTE	e_a	D-MTE	e_a	D-MTE	e_a	D-MTE	e_a	D-MTE	e_a	D-MTE	e_a
MUSE	0.163	\	0.230	\	0.193	\	0.410	\	0.454	\	0.242	\
IONet-fuse	0.339	1.018	0.438	1.267	0.445	1.026	0.366	1.112	0.968	1.116	0.289	1.362
DeepIT w/o mag	0.135	0.769	0.333	1.256	0.198	0.821	0.108	0.385	0.301	1.517	0.192	0.675
DeepIT w/ mag	0.107	0.303	0.202	0.465	0.162	0.242	0.163	0.555	0.092	0.216	0.108	0.105

Table 6. Performance comparison under different attachments, NW scenario

model	In-Backpack					Handheld				
	MTE	T-MTE	D-MTE	e_l	e_a	MTE	T-MTE	D-MTE	e_l	e_a
MUSE	20.147	15.207	0.871	\	\	10.546	7.590	0.661	\	\
IONet-fuse	13.951	6.231	0.467	12.824	1.695	11.983	4.641	0.65	27.542	1.198
DeepIT w/ mag	9.037	6.877	0.303	12.633	1.488	7.575	4.39	0.411	1.397	0.757

Table 7. Performance comparison of different models with a variety of devices, RP scenario

model	Black shark			Xiaomi Max2			Google Pixel 2			Xiaomi Mi5		
	D-MTE	e_l	e_a	D-MTE	e_l	e_a	D-MTE	e_l	e_a	D-MTE	e_l	e_a
MUSE	0.276	\	\	0.176	\	\	1.584	\	\	0.594	\	\
IONet-fuse	0.471	3.584	1.484	0.159	4.898	0.763	1.078	3.476	1.449	0.567	17.327	1.635
DeepIT w/ mag	0.068	7.706	0.181	0.109	4.982	0.305	0.441	7.556	0.704	0.445	6.141	0.612

Table 8. Performance comparison of different models with a variety of users, RP scenario

model	User 1			User 2			User 3			User 4			User 5		
	D-MTE	e_l	e_a	D-MTE	e_l	e_a	D-MTE	e_l	e_a	D-MTE	e_l	e_a	D-MTE	e_l	e_a
MUSE	0.401	\	\	0.333	\	\	0.250	\	\	1.586	\	\	0.599	\	\
IONet-fuse	0.337	12.26	1.284	0.344	12.303	1.713	0.43	3.763	2.249	1.078	3.476	1.449	0.65	27.542	1.198
DeepIT w/ mag	0.09	1.815	0.213	0.165	8.567	0.544	0.092	5.532	0.261	0.441	7.556	0.704	0.411	1.397	0.757

Table 9. Ablation study for VIMU. Training and testing have different angle distributions. Training: both smartphone and earbud face upward. Testing: both smartphone and earbud face left by 90 degrees.

model	Different angle distribution				
	MTE	T-MTE	D-MTE	e_l	e_a
IONet-fuse	167.765	78.575	1.486	8.576	1.765
DeepFusion	138.542	65.276	1.228	11.576	1.537
DeepIT raw-IMU w/ mag	158.849	71.171	1.406	13.871	1.429
DeepIT w/ mag	21.661	9.798	0.191	8.999	0.794

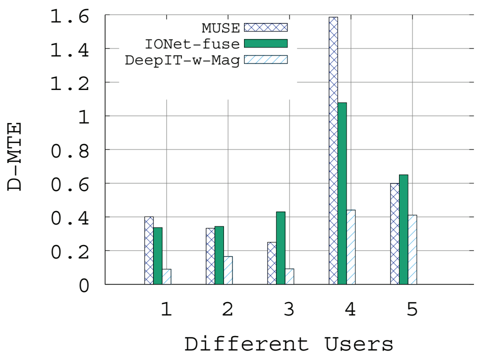

Fig. 19. Performance comparison of different models with a variety of users, RP scenario.

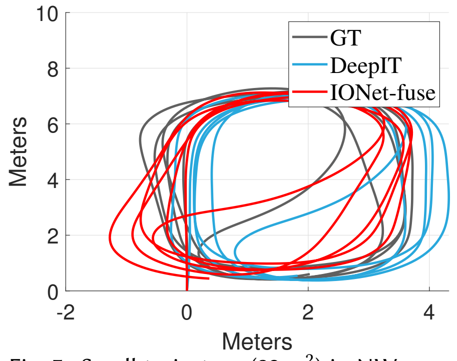

Fig. 7. Small trajectory (32 m^2) in NW scenario, outdoor.

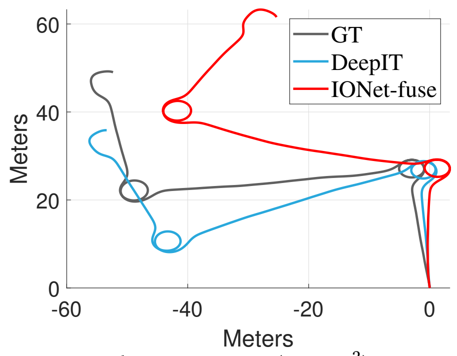

Fig. 8. Medium trajectory (3000 m^2) in NW scenario, outdoor.

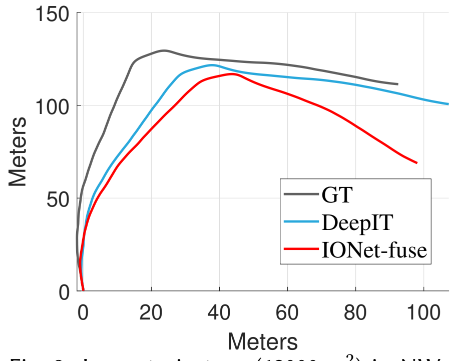

Fig. 9. Large trajectory (12000 m^2) in NW scenario, outdoor.

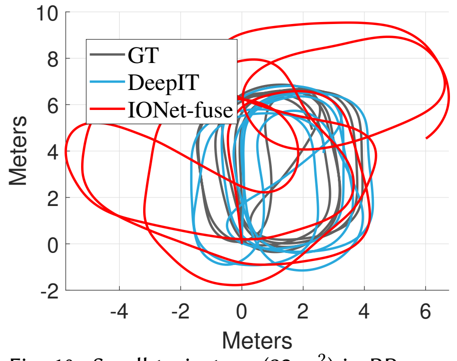

Fig. 10. Small trajectory (32 m^2) in RP scenario, outdoor.

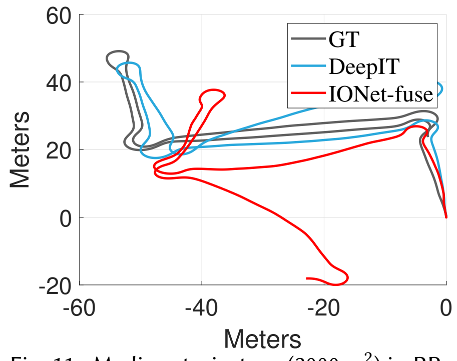

Fig. 11. Medium trajectory (3000 m^2) in RP scenario, outdoor.

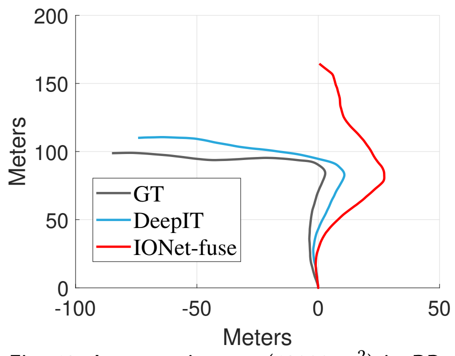

Fig. 12. Large trajectory (12000 m^2) in RP scenario, outdoor.

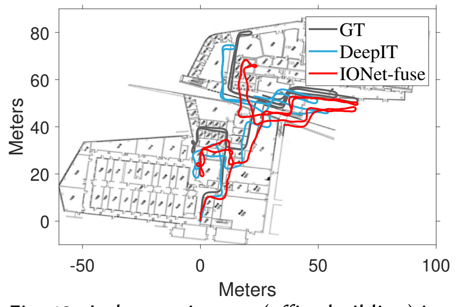

Fig. 13. Indoor trajectory (office building) in RP scenario.

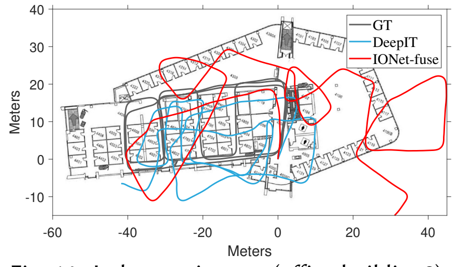

Fig. 14. Indoor trajectory (office building2) in RP scenario.

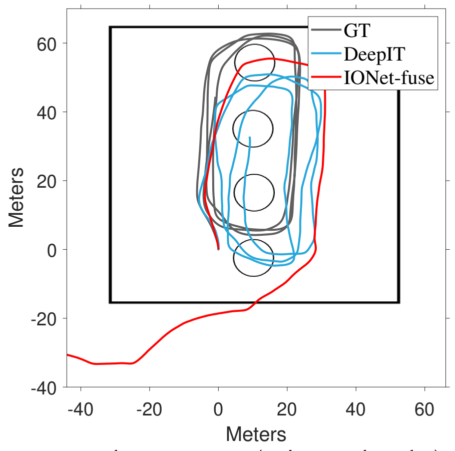

Fig. 15. Indoor trajectory (indoor parking lot) in RP scenario.

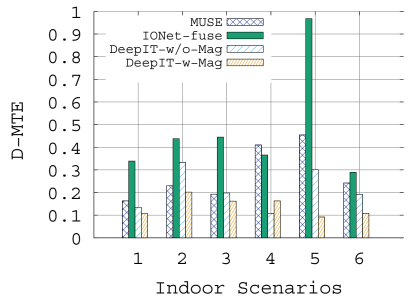

Fig. 16. Performance of different models in indoor places, RP scenario

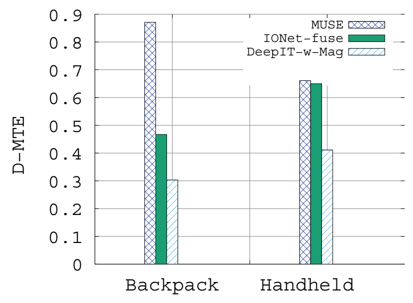

Fig. 17. Performance comparison under different attachments, NW scenario

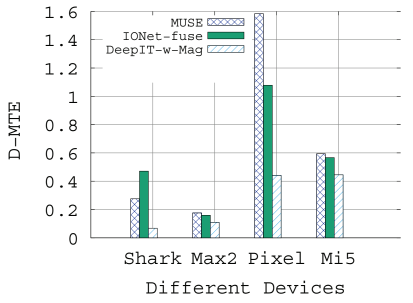

Fig. 18. Performance comparison of different models with a variety of devices, RP scenario

Indoor/outdoor scenarios. Due to the impacts of ferromagnetic materials in indoor environment (*e.g.*, metal frames in walls and furnitures), the magnetometer is easier to be interfered than outdoor. To verify whether DeepIT is robust under such disturbances, we evaluate 6 indoor sites, covering student center building, electronics laboratory, indoor parking lot filled with vehicles, along with 3 outdoor sites near residential buildings. In order to guarantee the generalization test, we ensure that *none of the tested routes exists in the training data set.* Since we focus on evaluating the generalization ability in this and following experiments, we use D-MTE as the main metric to normalize the impacts of the trajectory length. The numerical results of indoor experiments are shown in Table 5 and Fig. 16, and outdoor results are summarized in Table 2.

From the outdoor results, we find that the DeepIT fusion model with mag compensation shows best localization accuracy on all routes. The average D-MTE improvements compared with IONet-fuse, fusion w/o mag and MUSE are over 2×, 1.5×, and 20× respectively. Our fusion model effectively combats the interferences from motion artifacts, and lowers the average angle error e_a by 0.84 rad and 0.23 rad compared to IONet-fuse and fusion w/o mag respectively. Among all the DL models, the IONet-fuse model shows worst results because the handheld smartphone is affected most by motion noise.

The outdoor sites demonstrate a similar trend. The average improvements of D-MTE, compared to IONet-fuse and MUSE, are almost 3× and 2× respectively. Overall, the results indicate that the DeepIT fusion model is still effective indoor. However, we find an exception for indoor site 4, where the D-MTE of DeepIT w/ mag increases compared with the w/o mag case. The reason is that indoor site 4 contains many metal handrails which severely distort the magnetometer readings.

In summary, the inertial tracking systems' behaviors in indoor environments are usually more complicated than outdoors. They contain more turnings, which worsen the impacts of angle drifting, and hence reduce the localization accuracy. Additionally, more magnetic field interference is induced by metal sundries, which may disturb the angle compensation. However, our angle compensation method still achieves the best performance compared with the state-of-the-art baselines. Fig. 7-15 visualizes a few example trajectories in our experiments. It is clear that DeepIT suffers from much less drift compared with IONet, and its trajectory is much closer to the ground truth (GT).

Different ways of carrying the mobile devices. We include an additional challenging experimental scenarios, namely in-backpack, to evaluate our DeepIT 's generalization across different ways of attachments. The in-backpack scenario mixes the periodic pattern caused by both human steps and sway of backpack. While rich features on human step are contained in the measurement, a higher degree of noise is also introduced as a by-product of reciprocating motions, potentially jeopardising the performance of existing inertial tracking methods. We emphasize that the training data are all from the handheld attachments scenario, so the additional testing attachment do not appear in the training data. The experimental results are shown in Table 6 and Fig. 17.

In the in-backpack attachment, the DeepIT model has the lowest D-MTE which is 0.303m. It is almost 1.5× lower than IONet-fuse and 3× lower than MUSE. The backpack endures more fierce walking shakes than the handheld attachments. However, it does not cause large deviations on the localization results because the walking patterns can still be discriminated by the inertial tracking algorithms, be it closed-form or data-driven. Therefore, the performance gain of DeepIT is relatively lower compared with other scenarios.

In the handheld attachment, DeepIT achieves lower MTE (7.575m) among these two attachments, which is 1.46m lower than 'in-backpack'. This is expected since the model is trained under the handheld attachment (although with orthogonal data samples).

Generalization across different mobile devices. We further verify the effectiveness of DeepIT on different smartphone (and hence IMU) hardware. We adopt 4 different devices (Xiaomi Mi5, Google Pixel 2, Blackshark V1 and Xiaomi Max2), and ask the same tester to collect the IMU data in the RP scenario. The training data is collected from one devices (Blackshark) and the testing data from others. The results are shown in Table 7 and Fig.18. On average, DeepIT shows the lowest D-MTE (0.27m) on all devices, 2.5× lower than IONet and 3× lower than MUSE. This implies the performance gain of DeepIT is consistent across different devices. Additionally, the D-MTE on different devices varies slightly. This is caused by the difference of test routes. The test routes collected by Google Pixel 2 and Xiaomi Mi5 have more turnings than the other 2 routes, leading to larger tracking error.

Generalization across users. We recruit 5 users with varying height and gait to collect data in the RP scenario. Users 1-3 run data collection in one site, while users 4-5 in another. To guarantee the rigor of evaluation, *no subject in the test has participated in the training data collection.* The results are shown in Table 8 and Fig. 19. The DeepIT fusion model shows obvious improvement on D-MTE on all users, which is almost 2.5× better than IONet-fuse and 3× better than MUSE on average. The experiment implies that DeepIT generalizes well across users. The major difference among different users lies in their walking speed. Therefore, we inspect their walking speed by observing the total length of route in unit time. We find that the walking speed difference impacts the translation estimation error e_l. The larger difference between the training set and the validation set leads to larger e_l. DeepIT shows lower e_l on average than IONet and MUSE, indicating that it generalizes better than the baseline models, and is less sensitive to walking speed variation.

Ablation study to validate VIMU. We verify the effect of VIMU by intentionally choosing a testing usage pattern that is not covered in the training data's angular distribution. For training, both the smartphone and earbuds face upwards; For testing, both are placed vertically facing the user's left. The test trajectory is similar to those in Fig. 11. We create two test cases, using original IMU data and VIMU data as input, labeled as *DeepIT raw-IMU* and *DeepIT* , respectively. The results in Table 9 show that the improvement of VIMU compared with IMU is larger than 7.5×, which indicates that *the VIMU scheme effectively eliminates the impacts of angle distribution and thus enables more robust trajectory estimation.*

Reliability output in a mixture of motion modes. We test DeepIT when different motion modes are mixed, *i.e.*, the user randomly switches between different motion modes, including standing still (ST), normal walking (NW), swinging smartphone in hands (RP), moving heads to see around (HM), swinging smartphone and moving heads alternatively (RPE), running with both smartphone and earbud interfered (RS). Figure. 20 shows the reliability outputs for both sensors. Regardless of the motion patterns, we see that the higher-reliability sensor tends to toggle between the earbud and smartphone, implying that the proposed reliability model indeed can adapt to different motion patterns by attending to the most reliable sensor.

Hyper-parameters for the reliability based magnetometer compensation.

We now evaluate the influence of the hyper-parameter p_{Th} of the magnetometer compensation. We compare 4 model settings: 1*DeepIT w/o mag and 3*DeepIT w/ mag with different p_{Th}. The average MTEs in 4 scenarios: NW, RP, RPE and HM, as shown in Table 10. We can see that the p_{Th} of -0.1 shows best average performance in all four scenarios, although different values only have subtle influence. In addition, all the models using magnetometer compensation show better results than the DeepIT w/o mag, which show that the system performance is not very sensitive to the selection of p_{Th}.

Table 10. Ablation study for different value of p_{Th}. The w/o refers to DeepIT w/o mag.

p_{Th}	Average MTE			
	NW	RP	RPE	HM
w/o	15.56	17.57	13.74	12.85
0	12.78	10.48	11.167	10.164
-0.1	12.42	7.42	10.57	9.94
-0.2	12.13	9.49	12.16	9.85

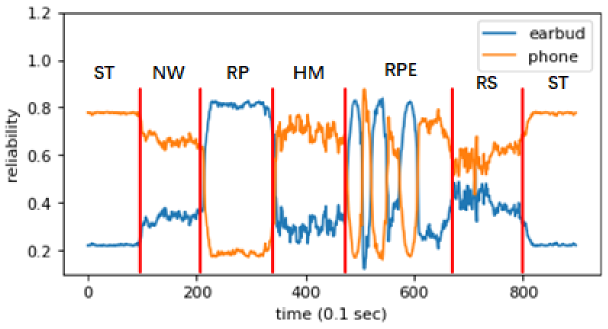

Fig. 20. Reliability variation under combined motion modes.

8 RELATED WORK

We summarize existing research in inertial motion sensing in 4 categories according to their methodology and design principles.

Closed-form Newtonian motion models. In early inertial navigation system design, the IMU is typically rigidly fixed on some parts of the user body, *e.g.*, foot or arm. The relative position and orientation deviation from an initial state can then be obtained by double integrating the accelerometer reading and integrating the gyroscope reading [16]. However, many real-world experiments have demonstrated that the integration of velocity is easily diverging due to the accumulation of error and noise of IMU sensors. More advanced step counting system design leveraged the unique human walking constraints to alleviate the drifting problem [1, 22, 36]. These systems accurately model the walking speed range, step size and other prior knowledge to restrict the estimation space. However, this method fails to generalize because the constraints they build inevitably vary across different people.

Filtering and calibration. Various kinds of adaptive filter methods have been merged into inertial tracking algorithms to improve the localization performance, such as Kalman filter [7, 15, 46], particle filter [33] and complementary filter [20, 21]. Kalman filter models the sensor data processing as a continuous time linear system, and elaborately handles the sensor noise by assuming the error model as Gaussian distribution. However, such an assumption only approximately works under slow rotational motion. It often fails in real scenarios involving sophisticated body parts motion along with walking. Particle filter can handle any arbitrary error distribution by representing the system as a set of samples (particles). However, prior physical constraints have to be accurately modeled in this method to restrict the computation space, which hampers its practical effectiveness, because accurately modeling motion constraints for daily mobility scenarios with high degree of freedom is infeasible. Complementary filter can mitigate the long-term angle drifting problem by fusing the magnetometer anchor with the integrated gyroscope angle estimation. However, for trajectory estimation, it still heavily relies on double integration which is prone to error accumulation.

Machine learning models. More recently, there have been attempts to use sequential deep learning models to realize data-driven inertial navigation systems [9]. IONet [3, 6] is an end-to-end deep learning model that predicts location transform in

Polar coordinate from raw IMU data. It aims to solve the inertial system drifting problem by breaking integration into separate windows and achieve meter-level accuracy, under the condition of simple trajectories and fixed phone posture (relative to user body). Robust Neural Inertial Navigation (RoNIN) [43] is another DL based model that takes a ResNet backbone combined with LSTM to reduce velocity error and increases the accuracy of prediction. Unlike such prior work, DeepIT represents the first to tackle the problem of motion noise due to varying IMU postures relative to the user body. DeepIT also represents the first to leverage a smart earbuds device, *i.e.*, eSense [3], to improve inertial tracking accuracy. DeepIT incorporates a few novel mechanisms in its DL model design, *e.g.*, utilizing the attention based sensor fusion scheme to synthesize the smartphone and earbuds sensors, combined with a reliability estimation network to further push the accuracy to its limit. With this measure, it can operate under more noisy and complex settings and handle different walking and running postures that are typical in practice.

Sensor fusion algorithms. Sensor fusion has been widely used to integrate multiple modalities and improve a system's robustness in real life applications. One early application of DL based sensor fusion is object detection, *e.g.*, combining RGB and depth image through convolutional neural networks to enhance the ability of detection [10]. Similar approach has also been used to fuse data from camera, LiDAR, IMU and even radar sensors [26, 29, 30], particularly for self-driving applications. Sensor fusion can also be applied on IMU and other mobile sensors to enhance the system performance [8, 23]. Muzner *et al.* used a CNN-based sensor fusion techniques for multi-modal human activity recognition [23]. More recent works focused on time-related sensor fusion architecture. Ming *et al.* [47] proposed an LSTM based multi-sensor fusion architecture to recognize human actions from continuous data observations. Yao *et al.* proposed an innovative self-attention based DL model fuse heterogeneous Internet-of-Things devices to a unified framework [45]. Recently, deep sensor fusion has been successfully applied to processing heterogeneous data modalities, such as millimeter-wave radar and IMU [19], heterogeneous mobile device sensors [42], visual sensors and IMU [5], RGB images and thermal cameras [31], WiFi and acoustic devices [41]. Compared with existing sensor fusion methods, the unique aspect in our DeepIT design lies in the reliability estimation network which assesses the motion noise level of heterogeneous IMU sensors and fuse the estimation via an attention model. This model incorporates physical meanings and shows remarkable advantage under daily ambulant scenarios.

9 DISCUSSION

9.1 Stability of earbuds and smartphones

Previous IMU based navigation systems [33] have shown that the IMU data on earbud, especially the accelerometer data, usually shows more stable patterns than smartphones. This raises an interesting question: would the earbud sensor data be enough for inertial tracking? Through the foregoing experiments, we observe that the earbud mainly benefits from the combination with smartphone sensors in three aspects: *(i)* The walking patterns sensed by the accelerometer on earbuds are too weak to infer walking speed. Due to the stability of the earbuds, the accelerometer data on the earbuds are filtered by nature, making it difficult to distinguish walking speed, thus causing errors on translation estimation. The smartphone does not have such issues and can thus complement the earbuds' limitation. *(ii)* Compensation of head motion noise. Head motion is very common in realistic cases. For instance, the head moves up/down when the user is browsing/texting on smartphone while inspecting the road situation ahead. Another common example is the head moving left/right when the user is distracted by ambient environment, or when the user is talking to people nearby. In these cases, the smartphone is necessary to provide more stable sensing output that reflects the actual movement of the user's body. *(iii)* Drifting compensation by smartphone magnetometer. Commercial earbuds usually avoid magnetometers as they may be impacted by the magnetic field in the loud speakers. Therefore, the drifting compensation of earbuds has to rely on the magnetometer on the smartphones using the proposed *reliability aware angle drift compensation* method described in Sec.5.

9.2 Real-time performance

DeepIT aims to run on smartphones and ideally the entire inference model needs to be processed in real-time. Although the 6 LSTMs seem to have a large computational burden, we find that the overall parameter size of DeepIT (75716) is only a fraction (1/4) compared with the IONet model (302978). This is due to the VIMU scheme in DeepIT , which not only decreases the overall input dimension from 6 (3 gyro and 3 accel) to 2 (1 virtual gyro and 1 virtual accel), but also decouples the gyroscope and accelerometer. By this means, the required parameter size of each single LSTM is largely decreased.

To test the real-time performance, we implement DeepIT along with IONet on Android using Pytorch. We run the inference model and feed the sensor data collected from the smartphone and earbuds in real-time. Then we repetitively run the inference model and calculate the average frame rate. We use a relatively low-end smartphone, XiaoMi BlackShark (Qualcomm Snapdragon

[3]To the best of our knowledge, eSense is the only publicly-available smart earbud to date that exposes the IMU sensor interface for data collection.

845) in the tests. The average frame rate of DeepIT is around 8Hz and the IONet is about 15Hz. The results indicate that, DeepIT can still meet the requirements of real-time inertial tracking. Even though it needs to process a higher dimension of sensor data, its processing speed is not fundamentally degraded owing to the smaller model size than IONet.

10 CONCLUSION

In this paper, we argue that motion noise remains as a major obstacle for IMU based location tracking in the wild. Without an explicit attack at this problem, existing methods can only work under controlled mobility, *e.g.*, holding a smartphone rigidly level on the hand or attaching it on legs while walking. Our DeepIT model fills the gap and marks a major step towards more usable inertial navigation applications. Through a reliability based sensor fusion scheme and drifting compensation algorithm, DeepIT achieves a large performance margin compared with state-of-the-art systems across diverse scenarios. Nonetheless, the performance of DeepIT is still not ideal for extremely long trajectories. We believe further refinement of its deep learning model is needed and can easily builds on our smartphone/earbuds dataset.

ACKNOWLEDGEMENT

This research was supported in part by the National Key R&D Program of China under Grant No. 2019YFA0706403, National Natural Science Foundation of China under Grant No. 62072472, 61702562 and U19A2067, Natural Science Foundation of Hunan Province, China under Grant No. 2020JJ2050, 111 Project under Grant No. B18059, the Young Elite Scientists Sponsorship Program by CAST under Grant No. 2018QNRC001 and the Young Talents Plan of Hunan Province of China under Grant No. 2019RS2001.

REFERENCES

[1] Agata Brajdic and Robert Harle. 2013. Walk detection and step counting on unconstrained smartphones. In *Proceedings of ACM international joint conference on Pervasive and ubiquitous computing*.

[2] MA Brodie, Alan Walmsley, and Wyatt Page. 2008. The static accuracy and calibration of inertial measurement units for 3D orientation. (2008).

[3] Changhao Chen, Xiaoxuan Lu, Andrew Markham, and Niki Trigoni. 2018. IONet: Learning to Cure the Curse of Drift in Inertial Odometry.

[4] Changhao Chen, Xiaoxuan Lu, Johan Wahlstrom, Andrew Markham, and Niki Trigoni. 2019. Deep neural network based inertial odometry using low-cost inertial measurement units. *IEEE Transactions on Mobile Computing* (2019).

[5] Changhao Chen, Stefano Rosa, Yishu Miao, Chris Xiaoxuan Lu, Wei Wu, Andrew Markham, and Niki Trigoni. 2019. Selective sensor fusion for neural visual-inertial odometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[6] Changhao Chen, Peijun Zhao, Chris Xiaoxuan Lu, Wei Wang, Andrew Markham, and Niki Trigoni. 2020. Deep-Learning-Based Pedestrian Inertial Navigation: Methods, Data Set, and On-Device Inference. *IEEE Internet of Things Journal* 7, 5 (2020).

[7] Zhenghua Chen, Han Zou, Hao Jiang, Qingchang Zhu, Yeng Chai Soh, and Lihua Xie. 2015. Fusion of WiFi, smartphone sensors and landmarks using the Kalman filter for indoor localization. *Sensors* 15, 1 (2015), 715–732.

[8] Seungeun Chung, Jiyoun Lim, Kyoung Ju Noh, Gague Kim, and Hyuntae Jeong. 2019. Sensor Data Acquisition and Multimodal Sensor Fusion for Human Activity Recognition Using Deep Learning. *Sensors* 19, 7 (2019). https://www.mdpi.com/1424-8220/19/7/1716

[9] Santiago Cortés, Arno Solin, and Juho Kannala. 2018. Deep learning based speed estimation for constraining strapdown inertial navigation on smartphones. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*.

[10] Andreas Eitel, Jost Springenberg, Luciano Spinello, Martin Riedmiller, and Wolfram Burgard. 2015. Multimodal Deep Learning for Robust RGB-D Object Recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[11] eSense. 2020. *eSense website*. https://www.esense.io/

[12] Andrea Ferlini, Alessandro Montanari, Cecilia Mascolo, and Robert Harle. 2019. Head Motion Tracking Through in-Ear Wearables. *Proceedings of the 1st International Workshop on Earable Computing* (2019).

[13] Google. 2020. *tango phone*. https://support.google.com/faqs/faq/6029402?hl=en

[14] Guoquan Huang. 2019. Visual-inertial navigation: A concise review. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 9572–9582.

[15] Benoit Huyghe, Jan Doutreloigne, and Jan Vanfleteren. 2009. 3D orientation tracking based on unscented Kalman filtering of accelerometer and magnetometer data. In *IEEE Sensors Applications Symposium*.

[16] Antonio R Jimenez, Fernando Seco, Carlos Prieto, and Jorge Guevara. 2009. A comparison of pedestrian dead-reckoning algorithms using a low-cost MEMS IMU. In *2009 IEEE International Symposium on Intelligent Signal Processing*. IEEE, 37–42.

[17] Wonho Kang and Youngnam Han. 2014. SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization. *IEEE Sensors journal* 15, 5 (2014), 2906–2916.

[18] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. 2015. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research* 34, 3 (2015), 314–334.

[19] Chris Xiaoxuan Lu, Muhamad Risqi U Saputra, Peijun Zhao, Yasin Almalioglu, Pedro PB de Gusmao, Changhao Chen, Ke Sun, Niki Trigoni, and Andrew Markham. 2020. milliEgo: single-chip mmWave radar aided egomotion estimation via deep sensor fusion. In *Proceedings of ACM SenSys*.

[20] Sebastian Madgwick. 2010. An efficient orientation filter for inertial and inertial/magnetic sensor arrays. *Report x-io and University of Bristol (UK)* 25 (2010).

[21] Robert Mahony, Tarek Hamel, and Jean-Michel Pflimlin. 2008. Nonlinear complementary filters on the special orthogonal group. *IEEE Transactions on automatic control* 53, 5 (2008), 1203–1218.

[22] Thibaud Michel, Hassen Fourati, Pierre Geneves, and Nabil Layaïda. 2015. A comparative analysis of attitude estimation for pedestrian navigation with smartphones. In *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, 1–10.

[23] Sebastian Münzner, Philip Schmidt, Attila Reiss, Michael Hanselmann, Rainer Stiefelhagen, and Robert Dürichen. 2017. CNN-based sensor fusion techniques for multimodal human activity recognition.

[24] Armando Neto, Douglas Macharet, Víctor Campos, and Mario Campos. 2009. Adaptive complementary filtering algorithm for mobile robot localization. *J. Braz. Comp. Soc.* (2009).

[25] Nonnarit O.-Larnnithipong and Armando Barreto. 2016. Gyroscope drift correction algorithm for inertial measurement unit used in hand motion tracking. *2016 IEEE SENSORS* (2016), 1–3.

[26] N. Patel, A. Choromanska, P. Krishnamurthy, and F. Khorrami. 2017. Sensor modality fusion with CNNs for UGV autonomous driving in indoor environments. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 1531–1536.

[27] Jay Prakash, Zhijian Yang, Yu-Lin Wei, and Romit Choudhury. 2019. STEAR: Robust Step Counting from Earables. https://doi.org/10.1145/3345615.3361133

[28] Pytorch. 2020. *Pytorch website*. https://pytorch.org/

[29] Kun Qian, Zhaoyuan He, and Xinyu Zhang. 2020. 3D Point Cloud Generation with Millimeter-Wave Radar. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4 (2020).

[30] Kun Qian, Shilin Zhu, Xinyu Zhang, and Li Erran Li. 2020. Robust Multimodal Vehicle Detection in Adverse Weather using Complementary Lidar and Radar Signals. In *Proceedings of IEEE CVPR*.

[31] Muhamad Risqi U Saputra, Pedro PB de Gusmao, Chris Xiaoxuan Lu, Yasin Almalioglu, Stefano Rosa, Changhao Chen, Johan Wahlström, Wei Wang, Andrew Markham, and Niki Trigoni. 2020. Deeptio: A deep thermal-inertial odometry with visual hallucination. *IEEE Robotics and Automation Letters* 5, 2 (2020).

[32] Paul G. Savage. 1998. Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms. *Journal of Guidance, Control, and Dynamics* 21, 1 (1998).

[33] Sheng Shen, Mahanth Gowda, and Romit Roy Choudhury. 2018. Closing the gaps in inertial motion tracking. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. 429–444.

[34] Yuanchao Shu, Kang G Shin, Tian He, and Jiming Chen. 2015. Last-mile navigation using smartphones. In *Proceedings of ACM MobiSys*.

[35] Jürgen Sturm, Stéphane Magnenat, Nikolas Engelhard, François Pomerleau, Francis Colas, Daniel Cremers, Roland Siegwart, and Wolfram Burgard. 2011. Towards a benchmark for RGB-D SLAM evaluation.

[36] Qinglin Tian, Zoran Salcic, I Kevin, Kai Wang, and Yun Pan. 2015. An enhanced pedestrian dead reckoning approach for pedestrian tracking using smartphones. In *2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*. IEEE, 1–6.

[37] Sam Tregillus and Eelke Folmer. 2016. Vr-step: Walking-in-place using inertial sensing for hands free navigation in mobile vr environments. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 1250–1255.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Proceedings of the International Conference on Neural Information Processing Systems*.

[39] B. Wagstaff and J. Kelly. 2018. LSTM-Based Zero-Velocity Detection for Robust Inertial Navigation. In *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*.

[40] Gerard Wilkinson, Ahmed Kharrufa, Jonathan Hook, Bradley Pursglove, Gavin Wood, Hendrik Haeuser, Nils Y Hammerla, Steve Hodges, and Patrick Olivier. 2016. Expressy: Using a wrist-worn inertial measurement unit to add expressiveness to touch-based interactions. In *Proceedings of the ACM CHI*.

[41] Hongfei Xue, Wenjun Jiang, Chenglin Miao, Fenglong Ma, Shiyang Wang, Ye Yuan, Shuochao Yao, Aidong Zhang, and Lu Su. 2020. DeepMV: Multi-view deep learning for device-free human activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–26.

[42] Hongfei Xue, Wenjun Jiang, Chenglin Miao, Ye Yuan, Fenglong Ma, Xin Ma, Yijiang Wang, Shuochao Yao, Wenyao Xu, Aidong Zhang, et al. 2019. Deepfusion: A deep learning framework for the fusion of heterogeneous sensory data. In *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 151–160.

[43] Hang Yan, Sachini Herath, and Yasutaka Furukawa. 2019. RoNIN: Robust Neural Inertial Navigation in the Wild: Benchmark, Evaluations, and New Methods. *arXiv preprint arXiv:1905.12853* (2019).

[44] Hang Yan, Qi Shan, and Yasutaka Furukawa. 2018. RIDI: Robust IMU double integration. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[45] S. Yao, Y. Zhao, H. Shao, D. Liu, S. Liu, Y. Hao, A. Piao, S. Hu, S. Lu, and T. F. Abdelzaher. 2019. SADeepSense: Self-Attention Deep Learning Framework for Heterogeneous On-Device Sensors in Internet of Things Applications. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. 1243–1251.

[46] Seanglidet Yean, Bu Sung Lee, Chai Kiat Yeo, and Chan Hua Vun. 2016. Algorithm for 3D orientation estimation based on Kalman filter and gradient descent. In *2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE, 1–6.

[47] Ming Zeng, Haoxiang Gao, Tong Yu, Ole J Mengshoel, Helge Langseth, Ian Lane, and Xiaobing Liu. 2018. Understanding and improving recurrent networks for human activity recognition by continuous attention. In *Proceedings of the 2018 ACM International Symposium on Wearable Computers*. 56–63.