RAMT: Real-time Attitude and Motion Tracking for Mobile Devices in Moving Vehicle

CHONGGUANG BI, Department of Computer Science and Engineering, Michigan State University

GUOLIANG XING, Department of Information Engineering, The Chinese University of Hong Kong

Recently a class of new in-vehicle technologies based on off-the-shelf mobile devices have been developed to improve driving safety and experience. For instance, wearables like the smartwatches are utilized to monitor the action of the driver and detect possible secondary tasks. Moreover, wearables can allow a driver to use gesture for in-vehicle controls, reducing distractions to driving. The accuracy of these systems can be significantly improved by tracking the real-time attitude of mobile devices. This paper proposes a novel system called Real-time Attitude and Motion Tracking (RAMT) that can enable a mobile device to accurately learn the coordinate system of a moving vehicle, and hence track its attitude and motion in real time. RAMT consists of a series of lightweight algorithms to sense the vehicle's movement and calculate the device's attitude. It provides a solution for trajectory-based gesture recognition. We have implemented RAMT on a smartphone and a smartwatch and evaluated the performance in 10 real driving trips. Our results show that the overall error of the coordinate system alignment is around 5° for the smartphone and 10° for the smartwatch, and over 84% of customized hand gestures can be accurately recognized with the result of RAMT. A video demo of RAMT is available at <https://youtu.be/9rZp7HxyRts>.

CCS Concepts: • **Human-centered computing** → **Mobile devices**; *Empirical studies in ubiquitous and mobile computing*.

Additional Key Words and Phrases: mobile devices, driving, motion tracking

ACM Reference Format:

Chongguang Bi and Guoliang Xing. 2019. RAMT: Real-time Attitude and Motion Tracking for Mobile Devices in Moving Vehicle. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 2, Article 38 (June 2019), 21 pages. <https://doi.org/10.1145/3328909>

1 INTRODUCTION

Recent years have witnessed the emergence of a class of new in-vehicle technologies for improving driving safety and experience. For instance, the smart steering wheel proposed in [30] can detect dangerous driving behaviors and warn the driver. The systems proposed in [45] [39] employ gesture control and auditory-only displays to avoid visual distraction. By installing additional equipment, these systems allow the driver to focus on driving without being distracted or handling secondary tasks. Such technologies can effectively improve driving safety since the major causes of road accidents are the distraction and secondary tasks, such as texting or operating the in-vehicle infotainment [8] [29] [26]. However, the aforementioned systems are based on proprietary technologies, presenting barriers to wide adaptation. Recently several in-vehicle systems based on off-the-shelf mobile and wearable devices have been developed to improve the driver's awareness. Specifically, smartwatches can be used to detect the driver's action based on inertial measurement units (IMUs) and monitor the driving performance by recognizing possible second tasks [27] [4].

Authors' addresses: Chongguang Bi, Department of Computer Science and Engineering, Michigan State University; Guoliang Xing, Department of Information Engineering, The Chinese University of Hong Kong.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2019 Association for Computing Machinery.

2474-9567/2019/6-ART38 \$15.00

<https://doi.org/10.1145/3328909>

Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., Vol. 3, No. 2, Article 38. Publication date: June 2019.

Fig. 1. The problem caused by the error of the estimation of horizontal heading. The motion signals collected by the accelerometer and the gyroscope are similar for gesture A, B, or C because of their identical shapes.

However, these systems often have poor performance due to the significant dynamics of moving vehicles. In particular, the motion of the smartwatch worn by the driver is induced by the movement of both the driver's hand and the vehicle. As a result, it is challenging to use the motion sensors (accelerometer and gyroscope) of the smartwatch to accurately track the hand motion or recognize the gesture in real time. A typical difficulty of motion-based tracking is the estimation of horizontal heading [15]. As shown in Fig. 1, those gestures cannot be distinguished based on motion signals without prior knowledge of the horizontal heading of the smartwatch, i.e. an alignment between the smartwatch and the driver or the vehicle. A possible solution to address this issue is leveraging GPS and compass [18] to compute the vehicle's moving direction and the device's attitude relative to the earth, which can then be combined to derive the device's attitude in the vehicle. However, the GPS signal is not always available or reliable. Moreover, the low sampling rate of GPS (around 1Hz) makes it impractical to track hand gestures in real time. The compass often fails to achieve accurate measurement because of the magnetic interference from the vehicle or other electronic devices [24]. Instead, most existing solutions still employ additional equipment like cameras and infrared distance sensors to capture the image and calculate the gesture in a 3-D space [13] [3] [21] [31] [42] to capture the real-time position of the wearable.

In this work, we propose a novel system called Real-time Attitude and Motion Tracking (RAMT) that can enable a mobile device to accurately learn the coordinate system of the moving vehicle, and hence track the attitude and the motion of the device in real time. The primary goal is to enable the mobile devices to learn their attitude and trajectory relative to the vehicle and support tracking-based gesture recognition like the solution based on computer vision. By sensing the vehicle's movement and calculating its impact on the motion sensors, the mobile device can extract its own motion and hence track the hand movement and recognize gestures. However, a key challenge lies in the fact that computing the accurate alignment between the coordinate systems of the mobile device and the vehicle poses prohibitively high overhead. To address this challenge, we propose a light-weight method that allows the smartwatch to efficiently search for the alignment between the coordinate systems of the vehicle and itself. The alignment of coordinate systems then allows a smartwatch to extract a user's hand motion. Our system is light-weight and can be run in real-time on mobile devices without communicating with the cloud. We have implemented RAMT on a smartphone and a smartwatch with an application that can recognize gestures by deep learning method. A video demo is available at <https://youtu.be/96vskQycRn8>. Our experiments based on real driving trips of 10 subjects show that the overall error of the coordinate system alignment is around 5° for the smartphone and 10° for the smartwatch and over 84% of the pre-defined and customized hand gestures can be accurately recognized.

2 RELATED WORK

Recently, a number of proprietary in-vehicle technologies have been deployed to monitor driver's performance, record the hand motion, and improve driving safety. For instance, drivers' level of attention can be monitored by measuring the vehicle's acceleration and speed [28] [10] [20] [22], driver's brainwave [7], heart rate [33] [6], and the holding force on the steering wheel [32]. The hand gestures can be classified based on the video captured by cameras [3] [21] [31] [42]. The systems proposed by [46] [45] use capacitive sensors to track the hand motion. By introducing image-based and depth-based sensors like Kinect, the gestures can be classified with a high accuracy based on the trajectory [47] [14]. However, these systems are based on proprietary technologies and incur significant costs of adoption.

Several studies have been focused on gesture recognition using mobile devices in static environments. For instance, the gesture of arm can be identified by a smartwatch by applying a particle filter on the motion signals [44]. Deep learning algorithms are developed to utilize motion sensors to control games [17], recognize hand gestures [5] [43], and analyze performance of physical activities [5] [25]. However, in these works, the devices still fails to estimate the horizontal heading. They rely on an initial coordinate system alignment between the device and the user or the earth. They also assume that the motion of mobile devices is solely induced by users.

With the high penetration of mobile devices, several systems utilize smartwatches and smartphones to detect whether the driver's hands are on the steering wheel and classify the secondary tasks [27] [4]. However, these systems only detect the shape of the gesture but cannot track hand motion in the 3D space. Several systems can sense the movement of the vehicle, including speed, braking, acceleration, and turning of the vehicle [34] [48] and infer the driving habits [36]. However, to our best knowledge, there does not exist an approach to align the coordinate systems of mobile devices and the vehicle solely based on motion sensors, which is a key enabling technology to precisely track hand motion and recognize hand gestures.

For the gesture recognition based on motion data, some existing methods define the segments of gestures, then recognize a single whole gesture as a combination of the segments using the Hidden Markov Model [13] [12]. Furthermore, because the gesture is related to the samples in the time-series, Dynamic Time Wrapping helps reducing the errors of recognizing the gestures with various sizes and duration [9] [35]. RAMT can improve the precision of the gesture recognition by providing fine-grained tracking of the attitude of the wearable. In this paper, we demonstrate an application with a deep learning model on mobile phone, which shows a hand gesture can be recognized with both shape and position based on the output of RAMT.

3 REQUIREMENTS AND CHALLENGES

The primary goal of RAMT is to let the mobile device recognize and keep track of the vehicle's coordinate system. As shown in Fig. 2, the devices must learn the vehicle's forward, right, and up directions, and represent these directions under the devices' own coordinate systems as unit vectors $\mathbf{r}$, $\mathbf{f}$, and $\mathbf{u}$. Once these unit vectors are determined, we can transform any vector $\mathbf{a}$ under the original coordinate system of a mobile device into a vector $\mathbf{a}'$ under the vehicle's coordinate system, by

$$\mathbf{a}' = \langle \text{PROJECT}(\mathbf{a}, \mathbf{r}), \text{PROJECT}(\mathbf{a}, \mathbf{f}), \text{PROJECT}(\mathbf{a}, \mathbf{u}) \rangle. \quad (1)$$

Specifically, the requirements of our design are, 1) the mobile device in the moving vehicle should learn its attitude relative to the vehicle; 2) the mobile device should be able to distinguish the vehicle's motion and the user's action from the collected motion signals; 3) the user's hand gesture in the vehicle should be effectively and accurately recognized based on tracking.

However, several challenges must be addressed. First, without prior knowledge, the vehicle's motion and the user's action are very difficult to be distinguished by the wearables. The motion sensors can sense either motion signal but they only output a combination of them. Second, it is difficult to learn the vehicle's coordinate system

Fig. 2. The goal of our design. The coordinate system of a mobile device is represented as $\langle X, Y, Z \rangle$. Our goal is to find and maintain a set of unit vectors $\langle \mathbf{r}, \mathbf{f}, \mathbf{u} \rangle$ under the coordinate systems of the mobile device, and those vectors always point to the forward, right, and up directions of the vehicle.

solely based on the motion signal, because it does not provide a concrete reference like location and orientation. Third, the hand motion tracking and the coordinate system alignment must be performed with a high level of accuracy and in real time during a driving trip. This requires processing high-dimension motion signals on resource-constrained mobile devices.

4 SYSTEM DESIGN

4.1 System Overview

RAMT targets to align the coordinate system of each mobile device to that of the vehicle. To learn the attitude and track the motion of the mobile devices, e.g. a smartphone and a smartwatch, we process and analyze the fused motion signals from both devices. The smartphone, which is not being used while driving, is able to pick up the vehicle's accelerating, braking and turning. On the other side, the motion data collected by the smartwatch contains not only the vehicle's motion but also the driver's hand motion. Our key idea is that the smartwatch receives the information about the vehicle's motion from the smartphone via the Bluetooth, and uses this information to learn the attitude of itself, then extracts the hand motion from the mixed motion signal.

Fig. 3 illustrates RAMT's pipeline. Each device calculates the up direction of the vehicle individually; the smartphone senses the vehicle's movement and calculates the forward direction, then sends this information to the smartwatch via the Bluetooth; the smartwatch calculates the vehicle's forward direction based on the received information and recognizes its attitude in the vehicle. According to the concept of the orthogonal coordinates, the right direction can then be calculated by the vector's cross product by the up and forward directions. Specifically, the motion sensors, i.e. accelerometers and gyroscopes on the smartphone and the smartwatch, continuously

Fig. 3. The pipeline of RAMT.

sample the motion signal. The direction of the gravity is calculated on each device by processing the collected data. This information is used to derive the up direction of the vehicle. The smartphone calculates the forward direction of the vehicle by processing the captured vehicle's motion. After the alignment of the coordinate systems between the phone and the vehicle, the vehicle's acceleration and turning at each moment are sent to the smartwatch. By resampling the received vehicle's motion, the smartwatch compares the motion data collected by itself with the vehicle's motion. Then, we use the gradient descent method to minimize the differential of the speed between the smartwatch and the vehicle, which searches for the best alignment between the two coordinate systems. Finally, the forward and the right directions of the vehicle are determined and maintained continuously by the smartwatch. The motion tracking of the smartwatch is performed simultaneously based on its real-time attitude.

4.2 Gravity Sensing

A common solution to determine the attitude of a device is to sense the direction of the gravity under the device's coordinate system. Since the measurement $\mathbf{a}$ from the accelerometer is always interfered by the gravity, the real acceleration caused by the motion of the device, which is denoted as $\mathbf{a}_L$, must be calculated by subtracting the impact of gravity from $\mathbf{a}$. The existing studies show that the reading of accelerometer can be used to represent the gravity by applying a low-pass filter, if the device is at rest [38]. However, for the mobile devices in a dynamic environment, e.g. in a moving vehicle, the traditional approach does not work. In this paper, we introduce a new approach that can achieve highly accurate gravity sensing even when the device is constantly in motion.

Hereafter, we denote the impact of the gravity on a device as $\mathbf{g}$. According to the characteristics of gravity, we have:

$$|\mathbf{g}| \approx 9.8, \quad \mathbf{a} = \mathbf{g} + \mathbf{a}_L. \quad (2)$$

The value of $\mathbf{g}$ can be derived by applying a low-pass filter on $\mathbf{a}$ while the device is moving at a constant speed or at rest, i.e. when $\mathbf{a}$ is close to $9.8m/s^2$ in a period of time and the reading from the gyroscope is close to 0 [16]. This state is referred to as "stable". The other state "moving" represents the condition in which the low-pass filter cannot yield an accurate value of $\mathbf{g}$, and this happens in the following two cases,

- 1) The motion data from the gyroscope shows that the device is rotating. For the smartphone, it may be picked up but not move along with the vehicle. This is a common case for smartwatch since it usually rotates with the wrist of the user. This may often occur to a smartphone in a vehicle when it is picked up by the user.
- 2) The device is not rotating but $\mathbf{a}$ is not close to $9.8m/s^2$. In this case, both the smartphone and the smartwatch may pick up the vehicle's horizontal acceleration, i.e. when the vehicle is accelerating, braking, or turning.

In these cases, we must calculate $\mathbf{g}$ by leveraging the rotation, which is sampled by the gyroscope and denoted as $\boldsymbol{\omega}$ at each moment. Assume $\mathbf{g}$ at the previous sampling moment is known, which is denoted as $\mathbf{g}_{old}$, and the rotation sensed by the gyroscope is also known as $\boldsymbol{\omega}$ at the current moment, then we can calculate the current impact of gravity $\mathbf{g}_{new}$ with the following steps:

1) Transform the rotation data $\boldsymbol{\omega}$ from the gyroscope to the axis-angle representation $\mathbf{R}$, following

$$\mathbf{R} = \left\{ \frac{\boldsymbol{\omega}}{|\boldsymbol{\omega}|}, |\boldsymbol{\omega}| \right\}. \quad (3)$$

2) Transform the axis-angle representation $\mathbf{R}$ to the rotation matrix $\mathcal{M}$ [19].

3) Calculate $\mathbf{g}_{new}$ by the matrix multiplication, as

$$\mathbf{g}_{new} = \mathcal{M}^{-1} \mathbf{g}_{old}. \quad (4)$$

In short, gravity sensing follows two distinct procedures in two cases. If the device is at rest or moving at a constant speed, the measurement by the accelerometer is exactly the impact of gravity. Otherwise, the direction of gravity can be calculated based on previous status and the current rotation. This process can be done in real time on each device. However, due to the bias and errors of the gyroscope, the accuracy of the sensed gravity gradually decreases if the device keeps moving, and it will be recovered once the device enters in the “stable” state.

The output of the gravity sensing is used to determine the upward direction of the vehicle, which is the same as the upward direction of any device under the earth’s coordinate system. Since $\mathbf{g}$ has the same direction as $\mathbf{u}$, we have

$$\mathbf{u} = \frac{\mathbf{g}}{|\mathbf{g}|}, \quad (5)$$

where $\mathbf{u}$ is the unit vector pointing to the up direction of the vehicle.

As shown in Fig. 4, after the gravity sensing, we can calculate the real acceleration $\mathbf{a}_L$, then decompose it with a horizontal acceleration $\mathbf{a}_H$ and a vertical acceleration $\mathbf{a}_V$. For the smartphone, $\mathbf{a}_H$ mainly contains the vehicle’s acceleration, braking, and turning, because the vehicle is supposed to move horizontally; $\mathbf{a}_V$ is mostly induced by from the noise, vibration, and harassment caused by the engine and the road condition. For the smartwatch, the real acceleration $\mathbf{a}_L$ not only contains the vehicle’s motion but also the motion of the user’s hand.

4.3 Vehicle Motion Detection

After deriving the gravity and calculating $\mathbf{a}_H$, the smartphone can detect the forward direction of the vehicle based on the vehicle’s horizontal movement. Our design is based on the following observations. First, since the smartphone is placed in the vehicle and not supposed to be used while driving, its motion sensors only pick up the movement of the vehicle. Second, except turning, the vehicle’s movement does not involve the rotation that changes the direction of gravity under the vehicle’s coordinate system, and the vehicle only moves along the axis pointing to the forward direction. Considering these features, we conclude that the smartphone can detect the forward direction of the vehicle by analyzing the collected motion data. Specifically, once the smartphone phone senses a significant $\mathbf{a}_H$, the detection of the vehicle’s movement can be done in three steps:

1) In a short period of time T , smartphone samples the rotations around the gravity. These samples correspond to the vehicle’s turning. We use β_i to denote the value of the rotation, which is a counter-clockwise angle between the beginning of the period and the moment i . This value can be calculated by,

$$\beta_i = \begin{cases} \Delta_t |\boldsymbol{\omega}_i|, & \text{if } \boldsymbol{\omega}_i \cdot \mathbf{u}_i \geq 0; \\ -\Delta_t |\boldsymbol{\omega}_i|, & \text{if } \boldsymbol{\omega}_i \cdot \mathbf{u}_i < 0; \end{cases}, \quad (6)$$

Fig. 4. The components of the accelerometer's measurement. (a) The measurement $\mathbf{a}$ of the accelerometer is always interfered by the gravity G , which induces a virtual acceleration $\mathbf{g}$. (b) $\mathbf{a}$ is the sum of $\mathbf{g}$ and real acceleration $\mathbf{a}_L$. (c) $\mathbf{a}_L$ can be decomposed as a vertical acceleration $\mathbf{a}_V$ and a horizontal acceleration $\mathbf{a}_H$.

Fig. 5. An example of the measured acceleration when the vehicle starts to move. Because the vertical vibration is more significant when the vehicle is moving, the horizontal acceleration can be observed from accelerating but not braking, i.e. the horizontal acceleration is in the forward direction of the vehicle.

Fig. 6. The detection of the motion of the vehicle. (a) Select the largest $\mathbf{a}_H$ as $\mathbf{a}_{Hmax}$, then project each $\mathbf{a}_H$ in the 1 second to $\mathbf{a}_{Hmax}$. (b) Define $\mathbf{a}_{Hmax}$ as the axis of the vehicle's forward or backward direction, the vehicle's motion in the 1 second is described by those projections.

where ω_i and $\mathbf{u}_i$ are the readings from the gyroscope and the unit vector pointing to the vehicle's up direction at the moment i in a frame with length Δ_t , respectively.

2) To detect the vehicle's acceleration, the smartphone finds the largest $\mathbf{a}_H$ in T . This vector of acceleration is denoted as $\mathbf{a}_{Hmax}$. The direction of $\mathbf{a}_{Hmax}$ is assumed to be either the forward or the backward direction of the vehicle, and we can calculate the vehicle's forward direction under the smartphone's coordinate system by

$$\mathbf{f} = \begin{cases} \frac{\mathbf{a}_{Hmax}}{|\mathbf{a}_{Hmax}|}, & \text{if accelerating;} \\ -\frac{\mathbf{a}_{Hmax}}{|\mathbf{a}_{Hmax}|}, & \text{if braking;} \end{cases}, \quad (7)$$

where $\mathbf{f}$ is the unit vector pointing to the vehicle's forward direction.

3) The $\mathbf{a}_H$ at each moment in the 1 second is projected to $\mathbf{a}_{Hmax}$ as the vehicle's acceleration. We define the vehicle's acceleration at the moment i as b_i , and we have

$$b_i = \begin{cases} |\text{PROJECT}(\mathbf{a}_{H,i}, \mathbf{f})|, & \text{if } \mathbf{a}_{H,i} \cdot \mathbf{f} \geq 0; \\ -|\text{PROJECT}(\mathbf{a}_{H,i}, \mathbf{f})|, & \text{if } \mathbf{a}_{H,i} \cdot \mathbf{f} < 0; \end{cases} \quad (8)$$

By calculating b_i , the smartphone learns the axis that points to the forward or the backward direction of the vehicle. In order to identify the direction of $\mathbf{a}_{Hmax}$, we need to know whether the vehicle is actually accelerating or braking at this moment. A practical solution is to examine the amplitude of the vertical vibration. Based on our experiments, the vibration is much more significant when the vehicle is moving as shown in Fig. 5. According to this observation, whenever the vehicle starts to move or comes to a stop, the smartphone can determine the forward and the backward direction. In some rare cases, the measurements of the accelerometer on the smartphone cannot be directly used to derive the forward direction of the vehicle. The first case is when the vehicle is moving backward. The acceleration's direction is the opposite of the forward direction. This case is rare when the car is in normal driving condition and does not significantly affect the performance. The second case is driving on a long curved road like the ramp on the exit of the entrance of the freeway. The constant turning leads to a centripetal acceleration that can be sensed by the smartphone. This error can be fixed soon when the vehicle is moving straightly. The third case is related to the horizontal vibration caused by the road condition. However, compared with the accelerating or braking, the vibration has a special pattern with a large variance within a short period of time, which can be easily identified and filtered out.

Along with the detection of the vehicle's motion, the smartphone logs a series of the rotations $\{\beta\}$ as well as the vehicle's accelerations $\{b\}$. These data are sent to the smartwatch. In the next section, we introduce how the smartwatch learns the coordinate system of the vehicle based on the vehicle's motion.

Fig. 7. The maintenance of the vehicle's coordinate system on the smartwatch. (a) Although the smartwatch's orientation is subject to constant changes due to users' hand motion, the vehicle's coordinate system should always be adapted with the movement of the vehicle. (b) An example of adaptive coordinate system maintenance when the smartwatch enters in "stable" state. Here $\mathbf{u}$ is calibrated by the gravity sensing, then the new $\mathbf{f}$ is calculated by projecting the previous one to the horizontal plane, and $\mathbf{r}$ is calculated by the vector's cross product.

4.4 Adaptive Coordinate System Maintenance

Unlike the smartphone, since the smartwatch's motion may contain the user's hand motion, we must ensure that the recognized vehicle's coordinate system keeps absolutely still but not varies with the motion. For example, as shown in Fig. 7 (a), the smartwatch is rotated due to the motion of the hand. As a result, the vehicle's forward and right axes should also be rotated and point to the absolute directions under the smartwatch's coordinate system. To accomplish this, we need to process the rotations from the gyroscope and continuously update the

vehicle's forward direction f on the smartwatch. Just like how we keep track of the gravity, we calculate the f_{new} based on the f_{old} and ω . Specifically, we transform ω into axis-angle representation R , then to the rotation matrix M [19]:

$$f_{new} = M^{-1} f_{old}. \quad (9)$$

However, due to the error and bias of the gyroscope, we need to calibrate f when the sensed gravity g is adjusted after the smartwatch enters the “stable” state. In other words, the recognized vehicle's coordinate system is calibrated adaptively whenever the accumulated errors of the gyroscope are eliminated through the gravity sensing. At this moment, f_{old} is projected to the horizontal plane to calculate the direction of f_{new} , as shown in Fig. 7 (b). Moreover, the right direction of the vehicle can be determined by calculating the cross product of u and f .

By adaptively adjusting the recognized vehicle's coordinate system of the smartwatch, f points to an absolute direction no matter how the smartwatch moves. The smartwatch keeps logging f and combines them with the received vehicle's motion to align itself to the actual vehicle's coordinate system, which will be discussed in detail in the next section.

Fig. 8. Detect the vehicle's forward direction by the smartwatch. (a)(b) Assume that f points to the forward direction of the vehicle, calculate the acceleration caused by the driver's action by subtracting the vehicle's motion from a_H in this series of samples. (c) Rotate the assumed vehicle's forward direction by θ , assume that f' points to the forward direction of the vehicle. (d) Calculate the acceleration in this case, and it can be observed that the calculated acceleration caused by the driver's action is lower than the previous case. This means f' is more likely to be pointing to the vehicle's forward direction than f .

4.5 Coordinate System Alignment on Smartwatch

After receiving the vehicle's motion detected by the smartphone, the smartwatch uses the data to align the coordinate systems between the vehicle and itself. A key observation is since the smartwatch stays in the vehicle, its speed should be similar to the vehicle's speed when the smartwatch is in the “stable” state. If the smartwatch has prior knowledge of the vehicle's forward direction, i.e. a set $\{f\}$ containing the recognized forward direction of the vehicle under its coordinate system for the previous moments, it is possible to calculate the smartwatch's

acceleration along with the vehicle's forward axis at each moment i as $\text{PROJECT}(\mathbf{a}_H, \mathbf{f}_i)$. Moreover, if each $\mathbf{f}$ is recognized correctly at each moment during T , the accumulation of the horizontal acceleration, i.e. the smartwatch's speed along with the vehicle's forward axis, must be the same value as the vehicle's speed at the end of the hand movement. Based on this observation, we have,

$$\epsilon = \left| \sum_{i=0}^n (\text{PROJECT}(\mathbf{a}_{H,i}, \mathbf{f}'_i) - b_i \mathbf{f}'_i) \right|, \quad \epsilon \rightarrow 0, \quad (10)$$

where $\mathbf{a}_{H,i}$ is the horizontal acceleration sensed by smartphone at the moment i , $\mathbf{f}'_i$ is the unit vector correctly pointing to the forward direction of the vehicle at the moment i , and b_i is the vehicle's acceleration at the moment i . Here, for samples at moment 0 and moment n , the smartwatch is in the "stable" state. The differential ϵ is generally close to 0. Our goal is to search for a series of $\mathbf{f}'_i$ that can minimize the differential based on a set of priorly known $\{\mathbf{f}\}$ on the smartwatch and the received data from the smartwatch containing vehicle's turning angles $\{\beta\}$ as well as the vehicle's acceleration $\{b\}$.

Since the motion sensors on the smartwatch and the smartphone may have different sampling rates, the first step of this process is to resample the motion data that represents the vehicle's motion by linear nearest-neighbor interpolation. Fig. 9 shows an example of the process of resampling.

Fig. 9. An example of resampling the received vehicle's accelerations under the smartwatch's sampling rate.

Since the correct forward direction of the vehicle $\mathbf{f}'$ exists on the same horizontal plane with the priorly known one $\mathbf{f}$, our goal can be simplified to be searching for θ , which is the angle between $\mathbf{f}$ and $\mathbf{f}'$, and we have,

$$\mathbf{f}'_i = \text{ROTATE}(\mathbf{f}_i, \mathbf{u}_i, \sum_{j=0}^i \beta_j + \theta). \quad (11)$$

where function $\text{ROTATE}(\mathbf{v}, \mathbf{s}, \alpha)$ returns a vector that is calculated by rotating vector $\mathbf{v}$ with angle α counter-clockwise around axis $\mathbf{s}$. Fig. 8 shows an example of the selection of θ and how the differential ϵ changes by using $\mathbf{f}'$ instead of $\mathbf{f}$. According to Equation (10) and (11), we need to solve the value of θ that minimizes ϵ . However, this formula is a high-degree polynomial equation with trigonometric functions, and a general algebraic solution does not exist according to the Abel-Ruffini theorem [2]. In the following, we propose a gradient descent method to solve θ . Specifically, on the first attempt, we start with the $\theta = 0$, which is the previous alignment between the coordinate systems. Then, we search for θ with 0.01 learning rate in a range between -90° to 90° . The advantage is that not only it requires only a small memory footprint, but also the expectation of the error can be limited within 1° after around 200 trials. This algorithm can be executed within 50ms by an off-the-shelf smartwatch.

At the startup, $\mathbf{f}$ on the smartwatch is set to $\langle 0, 1, 0 \rangle$. With this method, every time the smartwatch receives the information about the vehicle's movement, we rotate the current $\mathbf{f}$ on the smartwatch with angle $\hat{\theta}$ around $\mathbf{u}$ in order to align it to the vehicle's actual forward direction.

4.6 Motion Tracking and Gesture Recognition

After a set of unit vectors $\langle \mathbf{r}, \mathbf{f}, \mathbf{u} \rangle$ are found and maintained by the previous procedures, the smartwatch's attitude is known under the vehicle's coordinate system. Since the smartwatch's motion is related to the user's action, the information of its attitude in the vehicle can be used to recognize the user's hand gesture in a 3-D space. According to the design of the motion sensors, the X-axis of the smartwatch is always parallel to the user's forearm, the user's gesture can be defined as how the direction of its X-axis is represented under the coordinate system of the vehicle, i.e. where the hand points to. We denote this vector as $\mathbf{x}'$, and we have

$$\begin{aligned} \mathbf{x}' &= \langle \text{PROJECT}(\mathbf{x}, \mathbf{r}), \text{PROJECT}(\mathbf{x}, \mathbf{u}) \rangle, \\ \mathbf{x} &= \langle 1, 0, 0 \rangle. \end{aligned} \quad (12)$$

Fig. 10. Basic idea of tracking-based gesture recognition. (a) The range of $\mathbf{x}'$ is limited to the area in front of the user in the vehicle. (b) The value of $\mathbf{x}'$ is used to distinguish gestures with similar shapes at different positions.

Fig. 11. Representation of a gesture by samples. (a) A gesture can be initially represented as a series of sample of $\mathbf{x}'$. (b) After a gesture is performed, two re-sampling methods are used. The first method samples data uniformly based on the time, the second method samples data uniformly based on the trajectory.

Note that we only need to know the projection of $\mathbf{x}$ to $\langle \mathbf{r}, \mathbf{u} \rangle$ plane, because two coordinate values are enough to determine a 3-D unit vector. We developed an application to recognize the gesture based on the result by a Convolutional Neural Network (CNN) deep learning model. In our experiment, when the watch face is tilt to upward, the sampling of a gesture starts [37]. The gesture can be represented as a series of $\mathbf{x}'$ as shown in Fig. 10(a). To keep the consistency of size of the data that represents a gesture, we take another resampling step as shown in Fig. 11(b). We pre-defined five basic and easy-to-understand shapes of the gestures, which are the circle

like “O”, the letter “Z”, the vertical down stroke like “1”, the horizontal right stroke like “-”, and the cross like “X”. Each gesture can be performed at one of the nine positions as shown in Fig. 10(b).

The pipeline of the design of the gesture recognition is shown in Fig. 13. We recorded 20 sample-series of each of the 45 combinations and trained the classifier by TensorFlow [1] on a laptop. The classifier uses a CNN model as shown in Fig. 12 with the input as a series of both time-based and trajectory-based sampled $\mathbf{x}'$. The model is strong in capturing the individual connectivity patterns for each gesture according to the design of CNN. The training can be done with those pre-defined gestures as well as additional customized ones. After this offline training, the generated model is downloaded to the smartphone for real-time gesture recognition with TensorFlow Lite.

Fig. 12. The structure of the CNN deep learning model. It takes k samples of the gesture as the input.

Fig. 13. The pipeline of training and testing process of gesture recognition based on RAMT.

5 EVALUATION

In order to evaluate the performance of RMA, We have implemented it on a smartphone (Moto G 2nd Gen.) and a smartwatch (Moto G 360 2nd Gen.). To evaluate the performance of the alignment, the recognized coordination

Table 1. Information of the devices

Device	CPU	RAM	TensorFlow
Moto G 2nd Gen.	Qualcomm Snapdragon 400, 1.2GHz, qual-cores	1GB	TensorFlow Lite
Moto G 360 2nd Gen.	Qualcomm Snapdragon 400, 1.2GHz quad-core	512MB	N/A
Lenovo ThinkPad	Intel Core i5-7300, 2.6GHz / 2.71GHz, quad-core	8GB	TensorFlow 1.8.1

Table 2. Information of the deep learning model

Number of Samples	Model Size	Time of Training	Time of Prediction	Accuracy of Validation
10	45KB	2min	310ms	87.3%
20	85KB	3min	430ms	95.8%
30	125KB	6min	535ms	98.3%
40	165KB	9min	625ms	98.4%

Table 3. Processing time of each step

Smartphone		Bluetooth	Smartwatch		
Sampling, Gravity Sensing	Vehicle Motion Extraction	Transmission	Sampling, Gravity Sensing, etc.	Resampling	Gradient Descent
1.32ms	4.18ms	257ms	3.77ms	6.79ms	38.6ms

system of the vehicle is displayed on both devices in real time. The displayed result can be switched between the RAMT and GPS with compass. For gesture recognition, the smartphone runs the CNN model. The recognized gesture is displayed on the smartphone. In addition to our pre-defined gestures, each subject is allowed to record two more customized gestures during the experiment, and a new CNN model will be trained immediately on a laptop.

We have recruited 10 subjects to test RAMT under the approval by the Institutional Review Board (IRB) of the author's institution. The experiment is done in a moving vehicle. In the vehicle, the ground truth of the coordinate system of the vehicle is indicated as a preset cotton string, and the gesture performed by the subject is captured by a camera above the seat as shown in Fig. 14. Each subject is asked to wear the smartwatch (in our experiment, all of the 10 subjects choose to wear the smartwatch on the left hand) and carry the smartphone. For safety concerns, in each experimental session, the subject performing the experiments is a passenger during a 30-min driving trip. During this trip, the subject is responsible to regularly check the accuracy of the alignment, and he/she is also required to perform 3 times of each customized gestures and 30 randomly chosen gestures, which comes from the 45 pre-defined shape-position combinations, then records the errors of the gesture recognition.

5.1 Computational Performance

We use devices listed in Table. 1 for this experiment. As shown in Fig. 17, RAMT is able to run on multiple threads on each mobile device. The processing time of each sample is sufficiently short comparing with the sampling rate of the motion sensors. The details of the average processing time of each step is listed in Table. ??.

The size and the processing time of the CNN model depend on the number of samples we extracted from the gestures as shown in Fig. 12. In our study, we use a 0.05 learning rate and 50000 rounds of the gradient descents to train the model. The outcomes of each setup are summarized in Table. 2. In the experiment, we choose 30 samples for each gesture as a balance between the processing time and accuracy.

Fig. 14. RAMT displays the recognized coordinate system of the vehicle on the smartwatch. The error can be manually read by measuring the angle between the displayed front direction and the ground truth, which is labeled by a preset cotton string inside the vehicle. We also provide a protractor for the subject to assist the measurement. A camera records the gesture of the subjects.

Fig. 15. The route of our experiment. The city streets contain frequent stops and turns as well as a low-speed limit. Driving on the freeway does not require stops and sharp turns, and its speed limit is high. Rural roads are mostly straight with a small traffic.

Fig. 16. The error of the alignment, represented as the angle between the detected forward direction of the vehicle and the truth.

Fig. 17. The timeline chart of an example of the processing of RAMT.

5.2 Coordinate System Alignment and Maintenance

As mentioned in Section 3, RAMT searches for a set of unit vectors $\mathbf{r}$, $\mathbf{f}$, and $\mathbf{u}$ under the native coordinate system of each device, which represents the coordinate system of the vehicle (the forward, right, and up directions). Since $\mathbf{u}$ can be calculated via gravity sensing and $\mathbf{r}$ can be calculated by a cross product with $\mathbf{f}$ and $\mathbf{u}$, we only focus on the horizontal accuracy of $\mathbf{f}$. To this end, the subject is asked to compare a horizontal cotton string in the vehicle with the detected forward direction, which is displayed on the mobile device's screen. The angle between those two directions is the error of the coordinate system alignment in the horizontal panel.

Fig. 18. The average error of the alignment on each type of road and the standard deviation.

Fig. 19. The error of the alignment on a turn. The samples are collected on the last turn on the rural roads.

The smartphone captures the vehicle's horizontal motion and recognizes the forward direction of the vehicle. According to the design of RAMT, the alignment will be performed after a significant change in the vehicle's speed. These often occur before or after the vehicle's turning. As shown in Fig. 16 and Fig. 18, the average error is around 5° throughout the trip. The error is lower on the city streets, where the vehicle stops and turns frequently. However, an error of around 15° can be observed when the vehicle enters or exits the freeway. The long-lasting turning leads to a significant centripetal force, which is wrongly detected as an acceleration. The error can be fixed soon when the vehicle leaves the ramp. The smartwatch receives the extracted vehicle's motion from the smartphone and calculates the forward direction of the vehicle. The average error is around 10° all through the trip. In addition to the errors occurring on the ramp, the error gradually increases while the vehicle is driving straightly for a long distance, e.g. on the freeway or the rural road, where the vehicle's speed does not vary significantly. Our study shows that the error comes from the zero drifting of the gyroscopes on both of the smartphone and the smartwatch.

As shown in Fig. 18, the solution based on GPS and compass is relatively better for the smartphone when the vehicle is moving straightly. However, when the vehicle turns or shifts a lane, the error appears in a burst. Fig. 19 shows a detailed result of how the accuracy varies when the vehicle enters or exits the turn. For both the smartphone and the smartwatch, the errors are reduced when the vehicle was braking or accelerating, and the errors while turning are less than 20° . The GPS and compass solution fails to maintain accuracy while turning because the GPS must process a series of samples to estimate the new direction, which leads to long latency. The compass and GPS solution has a relatively low accuracy for the smartwatch. The attitude estimation by the magnetometer is not robust for an object under frequent motion. Moreover, due to the irregular hand motion, the smartwatch constantly travels across the magnetic field from the vehicle or other devices [40]. As a result, the measurements of the magnetometer are not reliable for detecting a moving device's attitude relative to the earth.

5.3 Hand Gesture Recognition

Based on the result of hand gesture recognition, we build an error matrix as Fig. 20 to show the accuracy. Here we have 45 rows and columns (5 shapes multiplies 9 positions) for the pre-defined gestures and 20 rows and columns (2 customized gestures for each of 10 subjects) for the customized gestures. Each cell represents the number of errors in recognition, which corresponds to the truth on the row and predicted result on the column. In total, 360 gestures by the subjects are recognized by our application. The number of total errors is 57, which means approximately 84% accuracy. Among the predefined gestures, "z" has the least errors of both the recognition of the position and the shape. Gesture "o" and "x" are often misclassified with each other, because "x" often contains a similar shape as the circle as shown in Fig. 11. The recognition of the vertical position of gesture "I" has lower accuracy, because the start and the end of this gesture have a large variety among the subjects. This issue also happens to the horizontal position of the gesture "-". Overall, most of the errors appear at the upper and upper right position. This is related to the basic idea of RAMT and the special posture of the user in the vehicle. When the user sits in the vehicle and reaches to the upper-right position with the left hand, the lower arm actually has more freedom. Thus, although the subject supposes that he/she moves the hand to the upper-right position, the forearm posture, i.e. the attitude of the smartwatch, has a large variety.

For customized gestures, most of the subject only observe two or three errors, including false alarm and missed detection. However, the accuracy is relatively low for some customized gestures, which are shown in Fig. 21. Specifically, gesture "o" is often misclassified as the second one of the subject 1 and the second customized one of subject 10 (double circles), because a circle exists in those customized gestures. The second gesture of subject 4 does not only contains a circle but also contains a "z" shape. This causes some difficulty for our system to accurately recognize it. We observe that shape "v" is a preferred choice among most of the subjects. The result also shows that postures with "v" shape are not likely to be confused with other gestures.

6 DISCUSSION AND FUTURE WORK

Our results suggest that the accuracy of the alignment should be higher in a well-controlled situation. Once the smartphone is put in a vehicle and it is not touched or moved, the alignment of the coordinate system between the smartphone and the vehicle only needs be executed once. However, in our experiment, it is required to check the error of the alignment by manual operation. Thus, the alignment must constantly run on the smartphone to maintain accuracy. A possible improvement is only executing the alignment when the smartphone is moved by the user. This requires RMAT to distinguish the vehicle's motion (e.g. accelerating, turning, braking, vibration, etc.) and the activities by the user. The solution is left for future work.

Although deep learning has a good performance in gesture recognition, its computing overhead is still heavy for a mobile device. A recommended technique to recognize gestures is decomposing the gesture into multiple segments, then recognizing the whole gesture by analyzing the combination of the segments [23]. This technique

Fig. 20. The error matrix for detecting all gestures. Each entry is a number of errors of the gesture recognition. Each of the five pre-defined gesture is evaluated with nine positions, respectively.

Fig. 21. The trajectories of all the customized gestures by the subjects.

is proved to be effective with computer vision, but it is not designed for motion-based gesture recognition, which contains significantly more unpredictable errors in data sampling. It is hence important to develop a better machine learning methodology for gesture recognition or motion tracking solely based on motion signals.

Once the coordinate systems on the smartphone and the smartwatch are correctly aligned, RAMT can maintain an accurate alignment following the procedure in Section 4.4. The only factor that affects the accuracy is the accumulated zero drifting on the gyroscopes. Several studies have been conducted to predict the value of drifting based on various factors, such as ambient temperature [11] and the device's attitude [41]. However, these methods are not designed to work in real time and hence are ill-suited for our scenario. One possible solution in the future is to periodically use the GPS and compass alignment as a reference to mitigate the zero drifting, especially on the highway, where the acceleration and braking are rare.

7 CONCLUSION

In this paper, we present the design and implementation of RAMT, which aligns the coordinate systems of the mobile devices in a moving vehicle solely based on the motion signals. We address several challenges including processing high-dimension motion signal containing both motions of the vehicle and the user's hand on resource-limited mobile devices. RAMT comprises several new lightweight algorithms to derive the direction of the gravity on each mobile device and sense vehicle's movement. This method is evaluated by real driving trips with 10 subjects. The results show that the overall error of the alignment is around 5° on the smartphone and 10° on the smartwatch. The accuracy is relatively lower when driving on the city streets, which includes frequently accelerating and braking. We also present the effectiveness of hand tracking with an application of gesture recognition. The evaluation shows that over 84% of the hand gestures can be accurately recognized.

A LIST OF SYMBOLS

Symbol	Definition
$\mathbf{a}$	A sample of 3D acceleration, measured by the accelerometer.
$\boldsymbol{\omega}$	A sample of rotated angles on 3 axes, measured by the gyroscope.
$\mathbf{r}$	A unit vector representing the right direction of the vehicle.
$\mathbf{f}$	A unit vector representing the forward direction of the vehicle.
$\mathbf{u}$	A unit vector representing the up direction of the vehicle.
G	Acceleration caused by the gravity.
$\mathbf{g}$	Acceleration that is the opposite of G .
$\mathbf{a}_L$	Real acceleration of the mobile device without the impact of $\mathbf{g}$.
$\mathbf{a}_H$	Horizontal acceleration of the mobile device.
$\mathbf{a}_V$	Vertical acceleration of the mobile device.
R	Axis-angle representation of $\boldsymbol{\omega}$.
$\mathcal{M}$	Rotation matrix generated by R .
Δ_t	The length of a time frame, determined by the sampling rate of the gyroscope.
β	Angle of rotation around $\mathbf{u}$ on the smartphone, which is the angle of the vehicle's turning.
b	Value of the horizontal acceleration calculated on the smartphone, which is related to the vehicle's accelerating and braking.
θ	The angle between the prior known $\mathbf{f}$ and the true $\mathbf{f}$ on the smartwatch.

Function	Definition
$\text{PROJECT}(\mathbf{t}_1, \mathbf{t}_2)$	Returns a vector that is calculated by projecting $\mathbf{t}_1$ to $\mathbf{t}_2$.
$\text{ROTATE}(\mathbf{v}, \mathbf{s}, \alpha)$	Returns a vector that is calculated by rotating vector $\mathbf{v}$ with angle α counter-clockwisely around axis $\mathbf{s}$.

ACKNOWLEDGMENTS

This work is supported in part by the U.S. National Science Foundation under grant IIS1622659.

REFERENCES

- [1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, and others. Tensorflow: a system for large-scale machine learning.
- [2] Niels Henrik Abel. 1824. *Mémoire Sur Les Équations Algébriques Où on Démontre L'impssibilité de la Résolution de L'equation Générale Du Cinquième Degré...* Christiania, Groendahl, 1824. University of Oslo.
- [3] V. Belagiannis, S. Amin, M. Andriluka, B. Schiele, N. Navab, and S. Ilic. 2016. 3D Pictorial Structures Revisited: Multiple Human Pose Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 10 (Oct 2016), 1929–1942. DOI: <http://dx.doi.org/10.1109/TPAMI.2015.2509986>
- [4] Chongguang Bi, Jun Huang, Guoliang Xing, Landu Jiang, Xue Liu, and Minghua Chen. 2017. SafeWatch: A Wearable Hand Motion Tracking System for Improving Driving Safety. In *Internet-of-Things Design and Implementation (IoTDI), 2017 IEEE/ACM Second International Conference on*. IEEE, 223–232.
- [5] Henrik Blunck, Sourav Bhattacharya, Allan Stisen, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Mads Møller Jensen, and Tobias Sonne. 2016. ACTIVITY RECOGNITION ON SMART DEVICES: Dealing with Diversity in the Wild. *GetMobile: Mobile Comp. and Comm.* 20, 1 (July 2016), 34–38. DOI: <http://dx.doi.org/10.1145/2972413.2972425>
- [6] Amelie Bonde, Shijia Pan, Zhenhua Jia, Yanyong Zhang, Hae Young Noh, and Pei Zhang. 2018. VVRRM: Vehicular Vibration-Based Heart RR-Interval Monitoring System. In *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications (HotMobile '18)*. ACM, New York, NY, USA, 37–42. DOI: <http://dx.doi.org/10.1145/3177102.3177110>
- [7] Lee Boon-Leng, Lee Dae-Seok, and Lee Boon-Giin. 2015. Mobile-based wearable-type of driver fatigue detection by GSR and EMG. In *TENCON 2015-2015 IEEE Region 10 Conference*. IEEE, 1–4.
- [8] PC Burns, A Parkes, S Burton, RK Smith, and D Burch. 2002. *How Dangerous is Driving with a Mobile Phone?: Benchmarking the Impairment to Alcohol*. Transport Research Laboratory Berkshire, United Kingdom.
- [9] Josep Maria Carmona and Joan Climent. 2012. A Performance Evaluation of HMM and DTW for Gesture Recognition. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, Luis Alvarez, Marta Mejail, Luis Gomez, and Julio Jacobo (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 236–243.
- [10] Kang Chen and Haiying Shen. 2016. RoadAware: Learning Personalized Road Information on Daily Routes with Smartphones. In *Computer Communication and Networks (ICCCN), 2016 25th International Conference on*. IEEE, 1–9.
- [11] Shen Chong, Song Rui, Li Jie, Zhang Xiaoming, Tang Jun, Shi Yunbo, Liu Jun, and Cao Huiliang. 2016. Temperature drift modeling of MEMS gyroscope based on genetic-Elman neural network. *Mechanical Systems And Signal Processing* 72 (2016), 897–905.
- [12] J. W. Deng and H. T. Tsui. 2000. An HMM-based approach for gesture segmentation and recognition. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, Vol. 3. 679–682 vol.3. DOI: <http://dx.doi.org/10.1109/ICPR.2000.903636>
- [13] N. Deo, A. Rangesh, and M. Trivedi. 2016. In-vehicle Hand Gesture Recognition using Hidden Markov models. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. 2179–2184. DOI: <http://dx.doi.org/10.1109/ITSC.2016.7795908>
- [14] Cao Dong, Ming C Leu, and Zhaozheng Yin. 2015. American sign language alphabet recognition using microsoft kinect. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 44–52.
- [15] Kenneth Gade. 2016. The seven ways to find heading. *The Journal of Navigation* 69, 5 (2016), 955–970.
- [16] D. Gebre-Egziabher, G. H. Elkaim, J. D. Powell, and B. W. Parkinson. 2000. A gyro-free quaternion-based attitude determination system suitable for implementation using low cost sensors. In *IEEE 2000. Position Location and Navigation Symposium (Cat. No.00CH37062)*. 185–192. DOI: <http://dx.doi.org/10.1109/PLANS.2000.838301>
- [17] Paul Gilbertson, Paul Coulton, Fadi Chehimi, and Tamas Vajk. 2008. Using Tilt As an Interface to Control No-button 3-D Mobile Games. *Comput. Entertain.* 6, 3, Article 38 (Nov. 2008), 13 pages. DOI: <http://dx.doi.org/10.1145/1394021.1394031>
- [18] Paul D Groves. 2013. *Principles of GNSS, inertial, and multisensor integrated navigation systems*. Artech house.
- [19] HR Harrison. 1999. Quaternions and Rotation Sequences: a Primer with Applications to Orbits, Aerospace and Virtual Reality, Kuipers JB, Princeton University Press, 41 William Street, Princeton, NJ 08540, USA. 1999. 372pp. Illustrated.£ 35.00. ISBN 0-691-05872-5. *The Aeronautical Journal* 103, 1021 (1999), 175–175.

- [20] Bo-Jhang Ho, Paul Martin, Prashanth Swaminathan, and Mani Srivastava. 2015. From Pressure to Path: Barometer-based Vehicle Tracking. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 65–74.
- [21] David Hogg. 1983. Model-based vision: a program to see a walking person. *Image and Vision Computing* 1, 1 (1983), 5 – 20. DOI: [http://dx.doi.org/10.1016/0262-8856\(83\)90003-3](http://dx.doi.org/10.1016/0262-8856(83)90003-3)
- [22] Yunfei Hou, Abhishek Gupta, Tong Guan, Shaohan Hu, Lu Su, and Chunming Qiao. 2017. VehSense: Slippery Road Detection Using Smartphones. (05 2017).
- [23] Fu-An Huang, Chung-Yen Su, and Tsai-Te Chu. 2013. Kinect-based mid-air handwritten digit recognition using multiple segments and scaled coding. In *Intelligent Signal Processing and Communications Systems (ISPACS), 2013 International Symposium on*. IEEE, 694–697.
- [24] Hua Huang and Shan Lin. 2016. Toothbrushing monitoring using wrist watch. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*. ACM, 202–215.
- [25] D. A. James, N. Davey, and T. Rice. 2004. An accelerometer based sensor platform for insitu elite athlete performance analysis. In *Proceedings of IEEE Sensors, 2004*. 1373–1376 vol.3. DOI: <http://dx.doi.org/10.1109/ICSENS.2004.1426439>
- [26] James W Jenness, Raymond J Lattanzio, Maura O'Toole, and Nancy Taylor. 2002. Voice-activated dialing or eating a cheeseburger: which is more distracting during simulated driving?. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 46. SAGE Publications, 592–596.
- [27] Landu Jiang, Xinye Lin, Xue Liu, Chongguang Bi, and Guoliang Xing. 2018. SafeDrive: Detecting Distracted Driving Behaviors Using Wrist-Worn Devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (2018), 144.
- [28] Derick A Johnson and Mohan M Trivedi. 2011. Driving style recognition using a smartphone as a sensor platform. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*. IEEE, 1609–1615.
- [29] Sheila G Klauer, Feng Guo, Bruce G Simons-Morton, Marie Claude Ouimet, Suzanne E Lee, and Thomas A Dingus. 2014. Distracted driving and risk of road crashes among novice and experienced drivers. *New England journal of medicine* 370, 1 (2014), 54–59.
- [30] Markus Klausner and Wolfgang Grimm. 2006. Method for detecting the position of hands on a steering wheel. (March 28 2006). US Patent 7,019,623.
- [31] M. P. Kumar, A. Zisserman, and P. H. S. Torr. 2009. Efficient discriminative learning of parts-based models. In *2009 IEEE 12th International Conference on Computer Vision*. 552–559. DOI: <http://dx.doi.org/10.1109/ICCV.2009.5459192>
- [32] Larry Leavitt. 2000. Sleep-detecting driving gloves. (Jan. 18 2000). US Patent 6,016,103.
- [33] Lee Boon Leng, Lee Boon Giin, and Wan-Young Chung. 2015. Wearable driver drowsiness detection system based on biomedical and motion sensors. In *SENSORS, 2015 IEEE*. IEEE, 1–4.
- [34] Yantao Li, Fengtao Xue, Lei Feng, and Zehui Qu. 2017. A driving behavior detection system based on a smartphone's built-in sensor. *International Journal of Communication Systems* 30, 8 (2017).
- [35] Jeroen F Lichtenauer, Emile A Hendriks, and Marcel JT Reinders. 2008. Sign language recognition by combining statistical DTW and independent classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 11 (2008), 2040–2046.
- [36] Luyang Liu, Cagdas Karatas, Hongyu Li, Sheng Tan, Marco Gruteser, Jie Yang, Yingying Chen, and Richard P Martin. 2015. Toward detection of unsafe driving with wearables. In *Proceedings of the 2015 workshop on Wearable Systems and Applications*. ACM, 27–32.
- [37] Google LLC. 2019. Use wrist gestures on Wear. (2019). <https://developer.android.com/training/wearables/ui/wrist-gestures>
- [38] Jorge Lobo and Jorge Dias. 2003. Vision and inertial sensor cooperation using gravity as a vertical reference. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25, 12 (2003), 1597–1608.
- [39] Jun Ma and Yuchun Du. 2017. Study on the evaluation method of in-vehicle gesture control. In *Control Science and Systems Engineering (ICCSSE), 2017 3rd IEEE International Conference on*. IEEE, 145–148.
- [40] Stanley V Marshall. 1978. Vehicle detection using a magnetic field sensor. *IEEE transactions on vehicular technology* 27, 2 (1978), 65–68.
- [41] O Nonnarit, Armando Barreto, and others. 2016. Gyroscope drift correction algorithm for inertial measurement unit used in hand motion tracking. In *SENSORS, 2016 IEEE*. IEEE, 1–3.
- [42] C. A. Pickering, K. J. Burnham, and M. J. Richardson. 2007. A Research Study of Hand Gesture Recognition Technologies and Applications for Human Vehicle Interaction. In *2007 3rd Institution of Engineering and Technology Conference on Automotive Electronics*. 1–15.
- [43] Krishna Rath, Dinesh Patil, Sayli Bhavsar, Ketaki Jadhav, and Saurabh V Thakur. 2017. Gesture Human-Machine Interface (GHMI) in Home Automation. (2017).
- [44] Sheng Shen, He Wang, and Romit Roy Choudhury. 2016. I Am a Smartwatch and I Can Track My User's Arm. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '16)*. ACM, New York, NY, USA, 85–96. DOI: <http://dx.doi.org/10.1145/2906388.2906407>
- [45] Jason Sterkenburg, Steven Landry, and Myounghoon Jeon. 2017. Eyes-free In-vehicle Gesture Controls: Auditory-only Displays Reduced Visual Distraction and Workload. In *Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications Adjunct*. ACM, 195–200.
- [46] Hoang Truong, Phuc Nguyen, Nam Bui, Anh Nguyen, and Tam Vu. 2017. Demo: Low-power Capacitive Sensing Wristband for Hand Gesture Recognition. In *Proceedings of the 9th ACM Workshop on Wireless of the Students, by the Students, and for the Students (S3 '17)*.

- ACM, New York, NY, USA, 21–21. DOI: <http://dx.doi.org/10.1145/3131348.3131358>
- [47] Chong Wang, Zhong Liu, and Shing-Chow Chan. 2015. Superpixel-based hand gesture recognition with kinect depth camera. *IEEE transactions on multimedia* 17, 1 (2015), 29–39.
- [48] Jiadi Yu, Hongzi Zhu, Haofu Han, Yingying Jennifer Chen, Jie Yang, Yanmin Zhu, Zhongyang Chen, Guangtao Xue, and Minglu Li. 2016. Senspeed: Sensing driving conditions to estimate vehicle speed in urban environments. *IEEE Transactions on Mobile Computing* 15, 1 (2016), 202–216.

Received August 2018; revised February 2019; accepted April 2019