Practical Driving Analytics with Smartphone Sensors

Lei Kang and Suman Banerjee
University of Wisconsin-Madison

***Abstract*—Sensing various driving behaviors, such as accelerations, brakes, turns, and change lanes — is of great interest to many applications, e.g., understanding drive quality, detecting road conditions, and more. Many such applications rely on using smartphone placed in a vehicle to collect such data for ease of deployment and use. However, several driving analytics techniques in the recent past, including our own, make simplifying assumptions that the smartphone is stably fixed with certain orientation and the car is driving on flat roads. Our deployment experience reveals that existing approaches may cause orientation misalignment and acceleration over/under estimation due to road slopes and human interactions, which lead to significant sensing errors for driving analytics applications.**

In this paper, we present several innovative techniques to improve the overall accuracy and usability of smartphone sensors. First, we use machine learning techniques to detect smartphone's relative orientation changes caused by human interactions. Second, we design a slope-aware alignment algorithm to improve alignment accuracy. Third, we track the linear acceleration of the vehicle to address acceleration over/under estimation problems. Fourth, we evaluate the tradeoffs between GPS and inertial sensors, and fuse inertial sensors with GPS to improve the overall accuracy and usability. We develop a smartphone application called XSense that adopts the novel techniques to improve the overall accuracy on driving analytics. Our evaluation of XSense is conducted through measurements of more than 2,000 trips (more than 13,000 miles) from 16 drivers in the past three years, and shows that XSense improves the 75-percentile accuracy by $5\times$ comparing with well-tuned inertial sensors in traditional approach.

I. Introduction

Smartphone-based sensing has opened up a whole gamut of applications and services across many domains. In the world of the transportation system, it is being increasingly used for crowd-sourcing various forms of information, ranging from road and traffic conditions to data on traffic light patterns, and even interesting annotations by participants. One such application is the ability to independently monitor driving behavior — how well is one driving the vehicle, e.g., aggressive driving actions, such as rough acceleration, hard brakes, sharp turns, and more. The popularity of smartphones enable innovative ways of monitoring such behaviors and actions. For example, Cambridge Mobile Telematics [1] develops a smartphone app to capture driving behaviors and monitor driver distractions. Similarly, the well-known ride-sharing company Uber announced it will start tracking Uber drivers' driving behaviors with their smartphones and give them feedbacks that are more detailed than the five-star rating customers leave for each driver [2]. Another set of applications that can leverage sensing of motion parameters is to understand road conditions, to say, detect potholes [3] or to detect icy stretches during a bad snow event.

A common technique to detect various vehicle parameters, especially acceleration, braking, and turns — events which happen in short timescales — is to use built-in inertial measurement unit (IMU) sensors available in the mobile devices [4], [5], [6]. The general approach taken by such systems is to measure the three-dimensional acceleration using the accelerometer and to measure the three-dimensional relative rotation speed using a gyroscope, including various de-noising techniques to make precise estimates. An important step in such design is one of coordinate alignment where the system needs to calculate the rotation matrix that translates the coordinates of the smartphone to that of the vehicle.

However, we find that exsiting approaches may lead to significant estimation errors (especially acceleration) if the smartphone is not tightly mounted, there are relative orientation changes, and/or the vehicle is moving over road slopes. If the mobile device is not tightly fixed in a mount, the device may move occasionally within the mount and its orientation relative to the vehicle may frequently change causing measurement errors. Worse, if the device is occasionally held in a person's hand, the update process needs to be applied continuously significantly exacerbating the challenge. Finally, even if the device is perfectly fixed to a mount, there are potential errors since the device itself might not be able to determine whether the vehicle itself is on flat earth or is tilted due to the slope of a road.

To understand and improve the performance of inertial sensors, we develop several techniques to detect orientation changes, model mounting stability, and improve coordinate alignment accuracy. We model the orientation of the smartphone as a cluster of accelerometer data and use moving variance to detect possible orientation changes. We use *intra-cluster variance* (ICV) to model mounting stability of the smartphone and use x-percentile of the variance as an indicator of motion estimation accuracy. Our intuition is that loose placement introduces more noises into sensor data and leads to less accurate acceleration estimation. We also find that existing coordinate alignment techniques produce less accuracy when there are (even small) slopes due to the gravity components sensed by accelerometer. We use an example trip to show that slope may cause coordinate misalignment and acceleration over/under estimation. Therefore, we use IMU sensors to tracked the slope gradients and subtract the gravity components sensed by accelerometer.

We conclude that the performance of inertial sensors are

constrained due to road conditions or human interactions and only well-tuned algorithm can produce higher accuracy than GPS in low speed scenarios, while slope has limited impact on steering motion estimation performance by gyroscope. In addition, we found that, the 90-percentile accuracy of GPS is under $0.5m/s^2$, which indicate a better accuracy than accelerometer on average. In particular, we find that GPS is more accurate in high speed scenarios and only well-tuned inertial sensors can achieve comparable accuracy.

The second goal of this paper is to use the above observations to develop a combined system to conduct driving analytics, called XSense, a mobile application that achieve higher estimation accuracy due to slope awareness, and leverages GPS to complement inertial sensors, when the latter lacks necessary accuracy and is not available. More specifically, XSense is able to select the current best estimation based on confidence value from GPS and inertial sensors. By considering both inputs from GPS and inertial sensors, XSense has the $75th$ percentile error of $0.2m/s^2$, which is $5\times$ better than well-tuned inertial sensors in traditional approach.

We summarize our contributions as follows.

- We illustrate that the accuracy of smartphone built-in sensors are very sensitive to road conditions and human interactions when conducting driving analytics. Traditional slope-unaware approach may caused misalignment and acceleration over/under estimation, which may cause significant sensing errors. We develop several techniques to identify the usability and accuracy of inertial sensors and improve their performance.
- We consider using commodity mobile device GPS receivers to sense vehicle motion parameters, especially those related to acceleration and brakes, and evaluate the performance by more than 10,000 miles of driving data. We find that GPS can be a good candidate to estimate such vehicle motions, especially in medium and high speed scenarios (higher than $10m/s$ or $22mph$).
- We develop XSense, an Android application component that can selectively use GPS and inertial sensors for driving analytics. We release it to 9 volunteers for six months. Our evaluation shows that XSense can improve the overall performance comparing with using either GPS or well-tuned inertial sensors.

II. The Problems

Smartphone inertial sensors are used for driving analytics. The IMU accelerometer measures 3D acceleration changes and can be used to detect accelerations and brakes. The IMU gyroscope measures 3D angular change and can be used to detect steering motions such as turns and lane changes. Existing driving analytics approaches [4], [5], [6] assume ideal scenarios, i.e., the car is moving on flat road and the smartphone is stably mounted with fixed relative orientation to the car.

In trying to relax such assumptions, we found that the accuracy of inertial sensors, especially the accelerometer, are sensitive to road conditions, orientation changes and mounting stability (as a result of human interactions). We start with an example trip to illustrate the *acceleration over/under estimation* problem due to slopes and the effects of gravitation. We show that existing coordinate alignment method may cause *coordinate misalignment*. Such problems cannot be easily fixed due to the low accuracy of inertial sensors. We also discuss unmanaged smartphone usage cases and the impacts on driving analytics, i.e., relative orientation changes and the smartphone is not stably mounted.

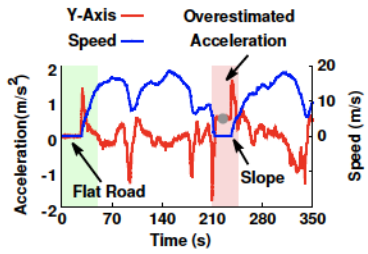

Fig. 1: The accelerometer y-axis (along the car's heading direction) and OBD speed readings.

A. Sensitive to Road Conditions

1) Acceleration Over/Under Estimation: We collect an example driving trip with various driving activities such as brakes and accelerations using a LG Nexus 5. Before the trip, we park the car in a flat parking lot and fix the smartphone with a frame mounted between the driver seat and the passenger seat. The coordinates of the phone are manually aligned (as precisely as we can) with the car. We use a customized Android application to record the sensor data and OBD speed data. The sensor and OBD data traces of the trip are illustrated in Fig. 1. As can be seen from the figure, the trip started on a flat road and the speed was $0m/s$ with acceleration $0m/s^2$. At time $210s$, there is a stop as the OBD speed reading was $0m/s$ while the accelerometer reading is around $0.5m/s^2$ indicating the car is accelerating. It is an overestimated acceleration due to road slope. This is because the y-axis of the accelerometer can sense the gravity. The gravitational force may have an incremental or decremental effect on the estimation of vehicle motion parameters. For example, a misalignment of five degrees may cause $0.85m/s^2$ acceleration estimation error. For reference, the *Snapshot Program* [7] records a hard brake if the deceleration is around $3m/s^2$. Therefore, we conclude that *road slopes and associated gravitational effect cause acceleration over/under estimation.*

2) Accumulated Gyroscope Errors: The gyroscope can be used to track three-dimensional angular changes and is an ideal input for slope estimation. However, it is known to suffer accumulated errors [8]. We observe similar error accumulation in our motivation experiment and the illustration figure can be found in the extended version [9]. There are several reasons why we see accumulated errors of gyroscope. The first one is the constant drifts, where the gyroscope reading is not zero in still due to limited hardware precision. As the angular changes add up, the constant drifts accumulate correspondingly, which leads to a rough linear function between accumulated error and time. The second one is the vibration of the vehicle, which accelerates the drifts of the gyroscope readings. Misalignment (caused by either manual alignment in our experiment or

the coordinate alignment algorithm) between the car and the smartphone can also introduce accumulated errors. This is because the x-axis of the gyroscope can also sense car's steering angular change like turns and lane changes when the smartphone is misaligned with the car.

3) Coordinate Misalignment: Coordinate alignment is the process to align the coordinates of the smartphone to those of the car [5], [4], [6]. We find that slopes cause misalignment if the coordinate alignment is conducted on slope. Coordinate misalignment refers to the case when the aligned coordinates of the smartphone is not perfectly aligned with the coordinates of the car. Since the accelerometer can sense gravity, a misalignment may also cause acceleration over/under estimation problem. For example, the car is moving upslope while the coordinate alignment algorithm assumes the car is moving on flat road, which leads to the intersection angle θ between aligned coordinates of the smartphone and the coordinates of the car. Such misalignment may cause acceleration over/under estimation when the driver is driving on flat road.

B. Sensitive to Human Interactions

Human interactions may change the relative orientation and mounting stability of the smartphone, which degrade the usability and accuracy of inertial sensors.

The relative orientation change causes the sensor readings are too noisy to be used. Suppose the smartphone is rotated horizontally 180 degree, then the output of accleleromter is completely reversed, i.e., the smartphone detects brake when the car is accelerating. To eliminate such errors, the rotation matrix should be re-estimated for accurate output. Coordinate alignment is an expensive process and frequent coordinate alignment may lead to gray periods where the inertial sensors cannot used for driving analytics.

Mounting stability is also an important factor when conducting driving analytics by using inertial sensors. When the user puts the smartphone in the pocket, the vibration of smartphone may add extra noises. We advocate the use of a metric to quantify mounting stability and understand the sensor accuracy under various stability levels.

III. Sensing with Inertial Sensors

IMU sensors are used in many driving analytics applications [5], [4], [6]. However, existing work assume the car is moving on flat roads and the smartphone is stably mounted in the vehicle. Different from these work, we propose several novel techniques to improve the accuracy and usability of inertial sensors by detecting orientation change, modeling stability, conducting slope-aware coordinate alignment and linear acceleration estimation in a more practical manner. We design a slope-aware solutions to conduct coordinate alignment and track linear acceleration by removing the gravity component dynamically from the aligned accelerometer readings. Second, we use clustering techniques to detect relative orientation changes. Third, we use moving variance to model the mounting stability of the smartphone and evaluate sensing accruacy based on mounting statiblity. The major steps are presented in this section, interested readers may refer to [9] for more illustration examples.

A. Slope-Aware Alignment

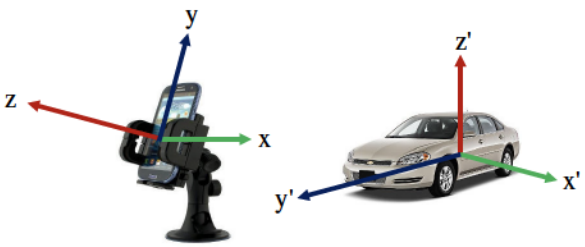

Fig. 2: Coordinate system difference between a smartphone $[x, y, z]$ and a car $[x', y', z']$.

As illustrated in Fig. 2, we use $[x, y, z]$ to represent the three dimensions of a smartphone and use $[x', y', z']$ to represent the three dimensions of a car. Coordinate alignment is the process that trains the rotation matrix $R = [\hat{i}, \hat{j}, \hat{k}]$, where $\hat{i}$, $\hat{j}$ and $\hat{k}$ are three unit coordinate vectors, so that $[x', y', z'] = [x, y, z] \times [\hat{i}, \hat{j}, \hat{k}]$.

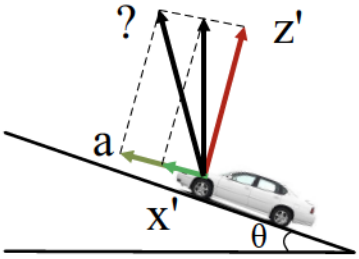

Fig. 3: Since the actual acceleration of the car and the component of gravitational force (along the slope) are unknown, the direction of force is uncertain when conducting vertical alignment.

Step 1: Stop Points Extraction.

Identifying stop points are useful to conduct an initial alignment. We use a sliding window to track the deviation of the accelerometer readings for this purpose. The deviation is expected to be small when the car is stopped, i.e., in front of stop sign of red traffic light. But different cars have different vibrations, which affect the readings of the accelerometer. To identify a threshold for the deviation, we extract the stop points according to the speed information collected from the OBD port. We record the start time s_t and end time e_t that any speed reading in between is zero. To eliminate possible asynchronization and drifted value after passing low-pass filter, we remove the points of the first 500ms and the last 500ms, i.e., the data points within $[s_t + 500, e_t - 500]$. This process helps us to set the threshold to detect stops. We only use OBD as a training input, our method can be used on any smartphone and vehicle settings without OBD inputs.

Step 2: Horizontal Alignment.

Road slope affects the accuracy of coordinate alignment in each step. During horizontal alignment, a slope-aware alignment method can be much more effective than a slope-unaware approach. Slope-unaware solutions assume all the data points pass the origin point and try to fit all the data points

for a single fit curve, while slope-aware solution treats each road segment as different inputs (deviating from the origin point due to slopes) and combine to improve the accuracy. To train the rotation matrix, we need to select the segments that the car is moving straight, and estimate the angle between the heading direction and the smartphone's horizontal coordinates [4]. To derive the rotation matrix from discrete sensor data points, we need to fit the curve and find the direction unit vector. Different from traditional approaches, we train the horizontal unit vector for each segment and combine each training results gradually with different weights. Each segment is selected based on the number of data points that indicate the car is moving. The intuition is that more data points could be more statistical significant.

Step 3: Slope-Aware Alignment.

Estimating road slope at alignment time is challenging. In horizontal alignment, we can fit a curve as the moving direction of the car (or the direction of force) is fixed and not much interfered by gravity. In vertical alignment, however, the direction of force is changing. As illustrated in Fig. 3, the direction of force changes as the change of the acceleration of the car. There are two parameters are unknown, the slope gradient and the vehicle acceleration. The vehicle acceleration varies at each data point, which varies the direction of force and makes it is very challenging to estimate the parameters. We use one heuristic searching algorithm to search for best alignment angle. The algorithm relies on the property that the $z' - axis$ of the car should be constant over the same slope. Firstly, we assume the horizontal alignment is complete and we convert a 3D alignment problem into a 2D alignment problem. The two dimensions are $y-axis$ and $z-axis$, as illustrated in Fig. 3. By iterating over all the possible angles, we calculate deviation of the $z' - axis$ and locate the one with the least deviation. The pseudo code of the search algorithm can be found in the extended version [9].

Step 4: Linear Acceleration Estimation.

After coordinate alignment is complete, we can track the acceleration of a car by using the accelerometer's y-axis. We illustrate the acceleration values from aligned accelerometer in an example trip illustrated in Fig. 4. As the figure shows, comparing with the accelerations by OBD, the accelerations by the accelerometer of the first $90s$ are underestimated and the following $60s$ are overestimated. This is because the car is moving mainly downslope and then mainly upslope. The deviated estimations may cause false positives/negatives on capturing driving behaviors such as brakes and accelerations.

We apply the same rotation matrix to the gyroscope data and illustrate the gyroscope x-axis data in Fig. 4. After removing the constant drifts, it shows clear trends that the car is moving downslope and then upslope. Given the similar trends illustrated in two plots, we can estimate the slope gradient and deduct the gravitational force components. We use similar calibration techniques proposed in [8]. We identify the road segments and stop points where accelerometer can provide more accurate slope gradients to calibrate gyroscope.

B. Detecting Relative Orientation Change

Using inertial sensors to estimate vehicle motions is also vulnerable to smartphone relative orientation changes. Relative orientation refers to the relative orientation between the smartphone and the car. When the smartphone is fixed in the car and the car is moving upslope or making turns, the relative orientation of the smartphone does not change as there is no relative movement between the smartphone and the car. The relative orientation is usually changed by the smartphone user, e.g., moving the smartphone from pocket to the car mount for navigation.

Moving Variance (MV). To timely detect orientation change, we look at the accelerometer data change in a moving window. The moving window contains m data samples, and we use the variance of the moving window to detect orientation changes. The intuition behind is that the changes of sensed gravity components is much more significant than those of caused by vehicle's movements. The variance can be calculated as $Var(x) = E[(X - \mu)^2]$, where X is the euclidean distance to the "moving cluster" center and μ is the average distance. There are cases that the movement is too small to be detected by MV. For example, a small horizontal rotation of the smartphone will not change the variance. We use the stability model to estimate such small changes. We will show in the later section that such looseness make the acceleration estimation inaccurate.

C. Estimating Mounting Stability

Another factor affects the accuracy of vehicle motion sensing is the mounting stability of the smartphone. The stability of the smartphone depends on how the smartphone is fixed/placed/held in the car. A good case scenario is the user placing the smartphone on a stable car mount holder. A bad case scenario is the user or passenger playing motion games such as car racing which requires rotating the smartphone. The derived sensor readings of the car are over noisy due to shaking and/or rotating of the smartphone.

We model one fixed relative orientation as a cluster of sensor readings. We use *intra-cluster variance (ICV)* to estimate the stability of the smartphone. A stable mounting of the smartphone is expected to produce a smaller variance than unstable holding by hands. Since ICV is correlated with the cluster size, we use the ICV of the subclusters to represent the ICV of the whole cluster. We define one subcluster as any continous n points of the whole cluster. We use ICV to evaluate the mounting stability of the tablets from dataset #1. We firstly remove all the trips that there is orientation change detected.

IV. The Promise of GPS in Driving Analytics

GPS is the most ubiquitous localization system and generally provides absolute coordinates. The usability of GPS in driving analytics has not been well discussed by the communities due to its high energy cost and coarse-grained measurements. Our measurements indicate GPS is a good candidate especially when IMU sensor is not available to use or less accurate. We find that GPS has high accuracy in

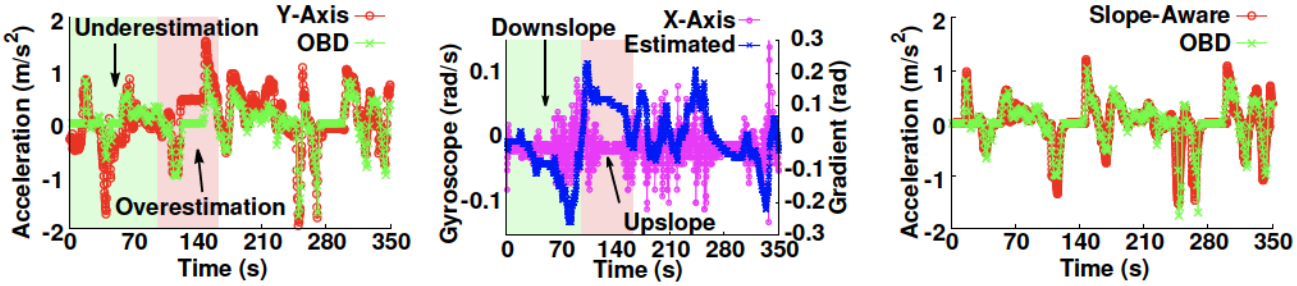

Fig. 4: Slope-Aware coordinate alignment and linear acceleration estimation. The figures are about acceleration over/under estimation caused by slopes (left), using gyroscope to estimate road slopes (middle), comparing estimated linear acceleration with groundtruth acceleration (right).

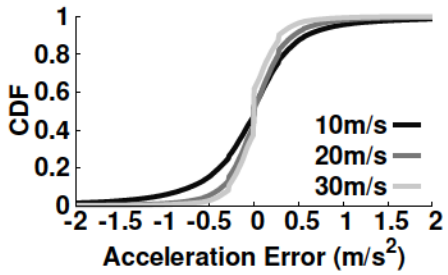

Fig. 5: Acceleration estimation errors of GPS in urban and highway environments. The estimation is more accurate in higher speed ($> 10m/s$) than lower speed ($< 10m/s$).

estimating speed and accelerations, especially in medium and high speed scenarios (e.g., $> 10m/s$). Given the simplicity of acceleration estimation and high accuracy of GPS, we believe inertial sensors can be augmented by GPS in vehicle speed and acceleration estimation applications. Please refer to [9] for the background about how GPS works and a more complete evaluation.

A. Acceleration Estimation

The acceleration is comparatively evaluated by GPS points and OBD speed data. The CDF of estimation error is shown in Fig. 5. Each curve stands for the acceleration estimation error no more than that speed, i.e., $10m/s$ refers to any speeds not higher than $10m/s$ and $20m/s$ refers to any speeds between $10m/s$ and $20m/s$. The estimation errors of GPS in low speed are caused by GPS location estimation errors, which could be caused by either the weak GPS satellite signal or strong thermo-noise floor. Additionally, the errors could also be caused by the building and vehicle blockage. The acceleration estimation when the vehicular speed is higher than $20m/s$ is highly accurate. This result indicates that GPS is more accurate in high speed scenarios.

B. GPS Availability

GPS signal is not always available, especially in indoor environments. If the smartphone is inside the car, there are chances that the smartphone GPS receiver is not able to detect satellite signals. Also, if the car is passing through underground tunnel or high buildings, GPS signal might be too

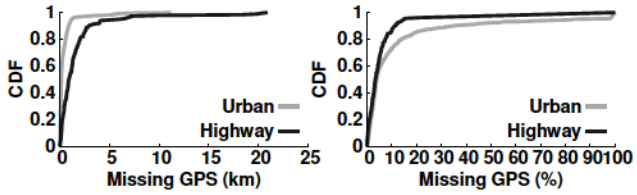

Fig. 6: GPS availability in urban and highway environments.

weak to be received. We compare GPS and OBD data point by point, and record the accumulated distance of trip fragment without GPS points. The missing GPS of each trip is illustrated in Fig. 6. There are a couple of trips that are entirely missing, while we are not sure if it is caused by software bugs or blocked signals. The GPS is available most of the time, while there are cases that the GPS is missing at the start and/or end of the trip.

V. Design and Implementation

We present the design and implementation of XSense in this section. The illustration of the workflow and the QR code for a link to download the xsense.apk package are provided in the extended version [9].

A. Design of XSense

1) Workflow: XSense uses several modules to evaluate sensor performance and estimate vehicle motion paramters. The inputs of XSense are accelerometer, gyroscope and GPS. In each module, it uses different sensors as input. The corresponding sensor icon is placed along with the module if it requires that sensor as input. Our method uses the accelerometer extensively as it provides a baseline by measuring the absolute acceleration while the gyroscope measures only relative angular speed. The first module is to detect relative orientation changes. If an orientation change is detected, XSense will restart the rotation matrix training process until the relative orientation of the phone is fixed. The second module is called stability monitoring module, which is used to monitor the mounting stability of the smartphone. The stability of the smartphone is quantified by a threshold, which depends on the accuracy of GPS and that of the accelerometer. If the accuracy of the accelerometer is higher than that of GPS, we think the smartphone is mounted stably. In other case, we use GPS to estimate acceleration. The third one is slope-aware

coordinate alignment module, which aligns the coordinates from the smartphone to those of the car and estimates the gradients of the training road segment. Both horizontal and vertical alignment are conducted in this module. The last one is the linear acceleration estimation module, which is used to estimate the slope gradients along the way. XSense may selective use GPS or accelerometer to estimate accelerations based on the accuracy of each method.

GPS is used to detect accelerations and brakes when the IMU sensors are not available or not ready to be used. IMU sensor is not available when the user frequently changes the relative orientation of the smartphone or does not mount the smartphone stably. For example, if the user is holding the smartphone to play game or send text message, the IMU sensor readings are too noisy to use. Even the user mounts the smartphone in a fixed place, it takes tens of seconds to minutes to collect enough data for training the rotation matrix. During the training process, the IMU sensors are not ready to be used.

B. Implementation

We implement XSense as a software module in Java and import it to an Android application we wrote. To make our test easier, we develop an offline trace replay engine. The replay engine sorts the sensor data based on timestamps, and feeds them into an event listener function in chronological order. We use an abstraction called *Trace* to represent the sensor data, GPS data and OBD parameters. It is similar to SensorEvent used by Android API [10] and provides additional flexibility to store OBD and GPS data. The event listener function process each *Trace* according to corresponding sensor type. The trace replay engine makes it easier to import XSense to the Android application. The Android application is aiming to monitor and record daily driving trips. The app is implemented by less than 4,000 lines of Java code, but supports a variety of functionalities such as trip recording, real time display, trip management, trip display on Google map, user management and access control, online/offline uploading, data synchronization with remote server etc. XSense serves as a module that provide a more accurate estimation on vehicle motions, e.g., hard brakes. For this submission, we highlight XSense module and remove some functionalities such as trip upload, user management etc.

VI. Evaluation

We present the dataset and accuracy evaluation of XSense in this section. More complete evaluation such as the steering angular velocity estimation and training time of rotation matrix can be found in [9].

A. Dataset

We use three datasets in our study. There are totally more than 13,000 miles of driving data collected in both controlled and uncontrolled environments over the past three years. We will make our dateset available to public after this work has been published.

Dataset #1. We deploy Xoom tablets installed with our app on 10 different cars. The tablets collect the vehicular speed data from the On-board diagnostics (OBD) port and various sensor data (including GPS and inertial sensors). Each tablet is placed in the back pocket of the passenger seat. The invovled cars are from different models and the stability of the tablet placement is different, which provides the opportunity to study the effects of mounting stability on motion estimation accuracy.

Dataset #2. We also collect some data in more controlled environments, where we know what is happening and the groundtruth data are recorded. First, we collect the data when the smartphone is placed in various scenarios, i.e., holding by hand, fixed by car mount holder, placed in cup holder etc. These data are used to understand various orientations and the relation between mounting stability and motion estimation accuracy. Second, we collect some data when randomly changing the orientation of the smartphone, i.e., move the smartphone from pocket and fix it on car mount holder. These data are used to evaluate the accuracy of our orientation change detection module. Third, we collect some data from two devices, where one device is manually aligned with the car (as best as we can), and the other is fixed in car mount holder or held in passenger's hand. These data are used in two cases. One trip is used to understand acceleration overestimation problem caused by gravitational force. Another 10 trips are used to estimate the vehicle steering motions, where the gyroscope readings of manually aligned device is used as groundtruth data.

Dataset #3. We release the beta version of the Android app to 9 volunteers. The trip recording module is written as an Android Service, so it is running in the background while the user may use the smartphone for navigation, game or any other activities. These data are used to understand how different users are placing their smartphones while driving or sitting in the car.

B. Road Slope Statistics

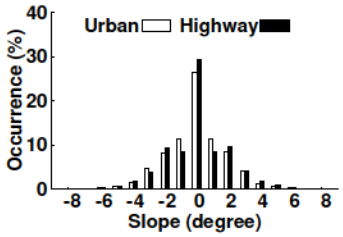

Fig. 7: Histogram of slope gradients. The acceleration estimation error is proportional to $gsin\theta$, where θ is the road slope angle.

As we discussed above, the slopes introduce linear acceleration estimation errors. An important question is how many road segments are sloping and how many of them are actually level. Some cities, e.g., San Diego and Los Angeles, are full of hills, and most roads are sloping roads. Surprisingly, in a plain area in US (Madison area) where we collect data, there are also full of slopes. To obtain the road gradient, we use the Google elevation dataset [11]. For each GPS data

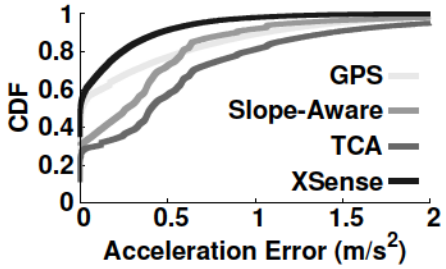

Fig. 8: Comparing acceleration estimation accuracy among various methods.

point, we queried the elevation from the dataset. and calculated the gradient by the elevation difference and distance. We eliminated the close consecutive GPS data points that less than 5 meters in distance and removed the data points where the speed is less than $10mph$. As we can see from the histogram in Fig. 7, more than half of the roads are not flat. Those sloping roads, even as small as two degrees, may introduce accumulated errors on coordinate alignment and further slope estimation, i.e., $0.34m/s^2$. The aggregated error may cause significant linear acceleration estimation error and introduce false positives/negatives on brake/acceleration monitoring.

C. Improved Overall Accuracy

We use dataset #1 to evaluate the accuracy of XSense and other methods. The other three methods are using Accelerometer with Traditional Coordinate Alignment (TCA) [5], [4], [6], Slope-Aware coordinate alignment with linear acceleration estimation, and GPS. In this evaluation, we only use the tight group data where the tablet is stably fixed in the car. Therefore, the results generated by sensors are the best cases can be achieved for sensor-based motion estimations. We compare the acceleration difference between each method and the groudtruth (calculated by OBD speed). The results are shown in Fig. 8. The estimation made by well-tuned sensor coordinate alignment and linear acceleration estimation (Slope-Aware curve) shows similar 80% accuracy with GPS. The gap between Slope-Aware and Accelerometer are caused by road slopes, where slope-unaware solution will over/under-estimate acceleration due to gravitational force. The accuracy gain of XSense is from the acceleration estimation compensation by sensors when the GPS speed is low.

D. Orientation Change Detection

We evaluate the orientation change module by using dataset #2 in this section.

Detection Rate. The performance of inertial sensors in vehicle motion sensing applications highly depends on the fixed relative orientation between the smartphone and the car. Therefore, detecting orientation change is very important in such applications. We evaluate the orientation change detection methods in two settings, tight setting and loose setting. In tight setting, the smartphone is mounted in various orientations on the car mount holder, or mounted by car cup holder. In loose setting, the smartphone is put in pocket, placed in passenger seat, or holding by passenger's hand. For each orientation, we record the groundtruth by a customized app. We use more than 10 trips and record 98 orientation changes in tight setting and 82 orientation changes in loose setting. The detection rate is presented in Table I. The Moving Variance method can detect most of the orientation changes expect when there is small horizontal orientation change. But such orientation change will increase the Intra-Cluster Variance so that the stability detection module can identify the polluted sensor output.

TABLE I: Orientation Change Detection Accuracy

Method	Tight	Loose
MV	96.9%	87.8%
MV + ICV	100%	96.3%

E. Comparison Between GPS and IMU Sensors

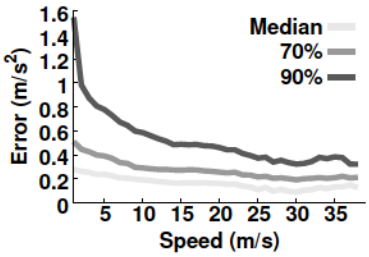

Fig. 9: GPS acceleration estimation errors under various speeds.

TABLE II: Median ICV and Acceleration Estimation Accuracy

ICV Median	Median Error	$70th-\%$	$90th-\%$
0.05	$0.15m/s^2$	$0.18m/s^2$	$0.31m/s^2$
0.21	$0.21m/s^2$	$0.38m/s^2$	$0.95m/s^2$
0.87	$0.45m/s^2$	$0.78m/s^2$	$1.56m/s^2$

To select between IMU sensor and GPS as input for acceleration estimation, we compare the accuracy based on current speed and mounting stability. To estimate the accuracy of GPS, we use one pipeline to process GPS stream data and track the speeds. Each GPS point is associated with a confidence value. The confidence value is the βth percentile estimation accuracy under given speed. The percentile accuracy under various speeds is illustrated in Fig. 9. To estimate the accuracy of IMU sensors, we use ICV to track the mounting stability of the smartphone. We use the median ICV as an indicator of the mounting stability. The corresponding percentile errors of different ICV are illustrated in Table II.

VII. Related Work

In this section, we summarize the most relevant work in the field of IMU sensing and GPS tracking. More discussion can be found in the extended version [9].

A. Driving Analytics with IMU Sensors

The smartphone built-in sensors enable lots of vehicular applications that sense vehicle movements and capture driving behaviors [5], [4], [6], [2], [12]. [5] uses an accelerometer to estimate vehicular speed. [4] captures turns to determine driver phone use by comparing centrifugal force with a reference point. [6] develops a middleware that can detect and differentiate various vehicle maneuvers, including lane changes, turns, and driving on curvy roads, by using non-vision sensors. The steering events are captured by the z-axis of the gyroscope. [13], [14] use inertial sensors to detect the driving quality of the driver. They identify driving events like lanechanging and acceleration/deceleration and rate the driver according to the frequency and suddenness of these events. Existing work assume the car is moving on flat road and the smartphone is stably mounted within the car. In relaxing such assumptions, we use slope-aware coordinate alignment and linear acceleration components to enhance the performance of these techniques. XSense also combines the inputs from both GPS and inertial sensors to achieve higher accuracy.

B. GPS-based Localization and Tracking

The most recent work about GPS relative localizations are [15], [16], [17]. [15] conducts relative localization and tracking among a network of GPS receivers by sharing the raw satellite observations and using carrier phases combined with double differentials to improve the accuracy of localization. [16] is a continuous work of [15] that eliminates the requirement of manual baseline initialization. [17] uses multiple GPS receivers on the drone as a failsafe mechanism for IMU failures. These techniques utilize multiple GPS receivers and cannot be used in single smartphone vehicular applications. [18] uses extra steerable, high-gain directional antenna as the front-end of GPS receiver to achieve direct GPS-based indoor localization. Different from these work, we use only single commodity smartphone GPS receiver to estimate vehicle motions. Improvement on the accuracy of GPS can improve the overall accuracy of XSense, since we use selectively use GPS and IMU sensor. We focus on single commercial smartphones where such techiniques are not yet available to use and will explore the possibility to integrate differential techniques in the future.

VIII. Conclusions

Smartphones are commonly used for driving analytics applications. Traditional approaches assume experimental cases where the smartphone is stably mounted with fixed relative orientation and the vehicle is travelling on flat roads. By using an example experiment, we show that even perfectly aligned accelerometer suffers acceleration overestimation or underestimation, which is caused by gravitational force, misalignment and slope estimation error. Moreover, the accuracy of IMU sensors are sensitive to human interactions as well. For example, frequent relative orientation change and less stable mounting may cause significant estimation errors. We present several innovative techniques to remedy such defects. We show through our experiments and analysis that compared to current state of the art techniques, our method improves the 75-percentile accuracy by $5\times$ comparing with well-tuned inertial sensors in traditional approach.

IX. Acknowledgments

We would like to acknowledge the anonymous reviewers. This research project is supported in part by the US National Sci- ence Foundation through awards CHE-1230751, CNS-1343363, CNS-1345293, CNS-1405667, CNS-1525586, CNS-1555426, CNS-1629833 and CNS-1647152.

References

[1] Cambridge Mobile Telematics. http://www.cmtelematics.com/.
[2] Uber Newsroom. https://newsroom.uber.com/curb-your-enthusiasm/.
[3] Jakob Eriksson, Lewis Girod, Bret Hull, Ryan Newton, Samuel Madden, and Hari Balakrishnan. The pothole patrol: using a mobile sensor network for road surface monitoring. In *Proceedings of the 6th international conference on Mobile systems, applications, and services*, pages 29–39. ACM, 2008.
[4] Yan Wang, Jie Yang, Hongbo Liu, Yingying Chen, Marco Gruteser, and Richard P Martin. Sensing vehicle dynamics for determining driver phone use. *ACM Mobisys*, 2013.
[5] H. Han, J. Yu, H. Zhu, Y. Chen, J. Yang, Y. Zhu, G. Xue, and M. Li. Senspeed: Sensing driving conditions to estimate vehicle speed in urban environments. 2014.
[6] Dongyao Chen, Kyong-Tak Cho, Sihui Han, Zhizhuo Jin, and Kang G Shin. Invisible sensing of vehicle steering with smartphones. In *Mobisys*. ACM, 2015.
[7] Progressive. http://www.progressive.com/auto/snapshot/.
[8] Pengfei Zhou, Mo Li, and Guobin Shen. Use it free: Instantly knowing your phone attitude. In *Mobicom*. ACM, 2014.
[9] Extended Version. https://www.dropbox.com/s/ocmjwdx6vup5i1k/full-version.pdf?dl=0.
[10] Andriod Developer Sensor Event. http://developer.android.com/reference/android/hardware/sensorevent.html.
[11] Google elevation api. *https://developers.google.com/maps*.
[12] Cambridge Mobile Telematics. http://www.cmtelematics.com/.
[13] Pushpendra Singh, Nikita Juneja, and Shruti Kapoor. Using mobile phone sensors to detect driving behavior. In *Proceedings of the 3rd ACM Symposium on Computing for Development*, page 53. ACM, 2013.
[14] M. Fazeen, B. Gozick, R. Dantu, M. Bhukhiya, et al. Safe driving using mobile phones. *TITS*, 2012.
[15] Will Hedgecock, Miklos Maroti, Janos Sallai, Peter Volgyesi, and Akos Ledeczi. High-accuracy differential tracking of low-cost gps receivers. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*, pages 221–234. ACM, 2013.
[16] Will Hedgecock, Miklos Maroti, Akos Ledeczi, Peter Volgyesi, and Rueben Banalagay. Accurate real-time relative localization using single-frequency gps. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, pages 206–220. ACM, 2014.
[17] Mahanth Gowda, Justin Manweiler, Ashutosh Dhekne, Romit Roy Choudhury, and Justin D Weisz. Tracking drone orientation with multiple gps receivers. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 280–293. ACM, 2016.
[18] Shahriar Nirjon, Jie Liu, Gerald DeJean, Bodhi Priyantha, Yuzhe Jin, and Ted Hart. Coin-gps: indoor localization from direct gps receiving. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*. ACM, 2014.