Invisible Sensing of Vehicle Steering with Smartphones

Dongyao Chen, Kyong-Tak Cho, Sihui Han, Zhizhuo Jin, and Kang G. Shin
Real-Time Computing Laboratory
Department of Electrical Engineering and Computer Science
University of Michigan – Ann Arbor, U.S.A
{chendy, ktcho, sihuihan, jzzfrank, kgshin}@umich.edu

ABSTRACT

Detecting how a vehicle is steered and then alarming drivers in real time is of utmost importance to the vehicle and the driver's safety, since fatal accidents are often caused by dangerous steering. Existing solutions for detecting dangerous maneuvers are implemented either in only high-end vehicles or on smartphones as mobile applications. However, most of them rely on the use of cameras, the performance of which is seriously constrained by their high visibility requirement. Moreover, such an over/sole-reliance on the use of cameras can be a distraction to the driver.

To alleviate these problems, we develop a vehicle steering detection middleware called `V-Sense` which can run on commodity smartphones without additional sensors or infrastructure support. Instead of using cameras, the core of `V-Sense` senses a vehicle's steering by only utilizing non-vision sensors on the smartphone. We design and evaluate algorithms for detecting and differentiating various vehicle maneuvers, including lane-changes, turns, and driving on curvy roads. Since `V-Sense` does not rely on use of cameras, its detection of vehicle steering is not affected by the (in)visibility of road objects or other vehicles. We first detail the design, implementation and evaluation of `V-Sense` and then demonstrate its practicality with two prevalent use cases: camera-free steering detection and fine-grained lane guidance. Our extensive evaluation results show that `V-Sense` is accurate in determining and differentiating various steering maneuvers, and is thus useful for a wide range of safety-assistance applications without additional sensors or infrastructure.

1. INTRODUCTION

Automobiles bring a wide range of conveniences as well as fatalities. In 2012, the reported number of fatalities from road accidents was 30,800 in the U.S. alone [4]. Of these fatalities, 23.1% involved lane control — i.e., merging or changing lanes or driving on curvy roads — and 7.7% involved turning maneuvers, i.e., turning left/right or making U-turns. In total, 30.8% of the fatalities were related to vehicle steering maneuvers.[1]

As most of these fatalities had resulted from the driver's careless or erroneous steering, those accidents could have been minimized or prevented if effective safety mechanisms had been deployed in the vehicles. There has been an ongoing push for incorporating electronic safety features in the vehicles to assist drivers' steering.

Steering-assistance systems, such as lane-departure warning or lane-keeping assistance, are the typical examples. They all exploit the advanced built-in sensors (e.g., cameras, radars, and infrared sensors) to detect the lane for driving assistance [13]. However, since they require special sensors which are only available on recent high-end cars, such safety solutions cannot be applied to a wide range of type/year models of cars.

To overcome such limitations, instead of using built-in vehicle sensors, efforts are being made to exploit the sensors in smartphones to assist drivers in their steering maneuvers. At one end of the spectrum of such applications, cameras have been used widely. The front/rear cameras of a smartphone are exploited to capture the images of road objects (e.g., traffic lanes, curb, and the preceding vehicle) which are then analyzed with image processing [8, 20, 29]. Although

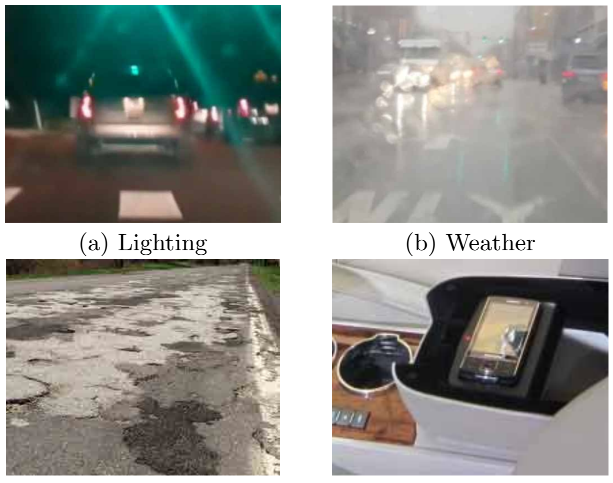

(a) Lighting (b) Weather (c) Pavement (d) Placement

Figure 1: Visibility distortion under different conditions

[1]We refer to *steering maneuvers* as either changing lanes, turning left/right, or driving on curvy roads.

MobiSys'15, May 18–22, 2015, Florence, Italy.
Copyright © 2015 ACM 978-1-4503-3494-5/15/05 ...$15.00.
http://dx.doi.org/10.1145/2742647.2742659.

such systems claim that smartphone cameras are sufficient in assisting the driver, they have limitations in terms of computational overhead and inaccuracy. The accuracy of camera-based approaches depends on *visibility* and can thus be infeasible, depending on the conditions listed below and shown in Fig. 1.

- **Lighting**: The functionality of camera-based approaches cannot be guaranteed in case of insufficient light, especially at night time.
- **Weather**: Rainy or snowy weather will make roads waterly or icy, and will thus distort the light reflection, rendering it difficult to identify road objects.
- **Pavement**: Bad pavement conditions will distort the shape of road objects and will thus cause false or miss detections.
- **Camera placement**: Placing the phone at a location where the camera cannot capture the road objects (e.g., in the driver's pocket), will diminish the feasibility of the camera-based approach.

The other end of the spectrum is to not use cameras. Smartphone sensors, such as gyroscope, accelerometer, magnetometer, etc., can be exploited to detect the vehicle steering maneuvers and thus perform the same steering-assistance functionalities that would have been achieved with use of cameras [25, 28]. These approaches have advantages of requiring much less computational resources and power, and also being immune to visibility distortions. However, it is known to be very difficult to differentiate the steering maneuvers, which is one of the main reasons for camera-based approaches being most prevalent.

In this paper, we propose `V-Sense`, a novel vehicle steering sensing middleware on smartphones, which overcomes the limitations and difficulties inherent in the existing camera-based and camera-free approaches. `V-Sense` is a camera-free middleware that can be utilized for various applications. It utilizes built-in Inertial Measurement Units (IMUs) on smartphones to detect various steering maneuvers of a vehicle. Specifically, `V-Sense` determines the changes in the angle of vehicle heading (i.e., steering) and the corresponding displacement during a steering maneuver. `V-Sense` classifies steering maneuvers into different types, such as turn, lane change, driving on curvy roads, etc., and exploits the classified results for various applications. We will elaborate on these applications in the following sections.

The contributions of this paper are three-fold:

- Design of `V-Sense`, an all-time vehicle steering sensing middleware which does not rely on use of cameras;
- Detection and differentiation of various steering maneuvers by only utilizing a smartphone's built-in sensors; and
- Proposal of two driving-assistance applications — i.e., careless steering detection and fine-grained lane guidance — which are based on `V-Sense` functionalities.

The remainder of this paper is organized as follows. Section 2 describes the motivation behind the design of `V-Sense`. Section 3 details the overall system and functionalities of `V-Sense`. We first evaluate the performance of `V-Sense` in Section 4 and then show two different applications of `V-Sense` in Sections 5 and 6. After reviewing the related work in Section 7, we finally conclude the paper in Section 8.

2. MOTIVATION

Without relying on use of cameras and by only utilizing non-vision sensors on commodity smartphones, `V-Sense` can detect various steering maneuvers, such as left/right turns, changing lanes, or driving on curvy roads. How could such functionalities of `V-Sense` without using cameras at all, assist the driver in terms of both convenience and safety? Can it actually help in reducing fatalities from road accidents? Given below are two proof-of-concept applications of `V-Sense` that enhance safety and convenience of the driver's steering.

Careless steering detection.

Careless steering — changing lanes or making turns *without* using the turn signal on the car — is one of the main reasons for steering-related fatalities. A study from the Society of Automotive Engineers (SAE) unveiled that people in the U.S. forget to use their turn signals 2 billion times each day in total, or roughly 750 billion times per year. Based on such figures, SAE argued that about 2 millions of accidents can be prevented by eliminating turn signal neglects or careless steering [27]. In this application, `V-Sense` provides the functionalities required for detecting such careless maneuver. By combining lane-change detection via `V-Sense` and turn signal sound detection — which is designed based on a matched filter — the application determines whether the steering maneuver was accompanied with a turn signal, i.e., detecting whether the steering was careless or not.

Fine-grained lane guidance.

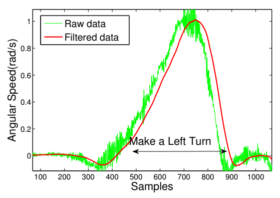
Existing navigation applications (e.g., Google map) provide information on *which* lane to stay on for preparing the next maneuver, i.e., indicating the correct lane. However, they lack functionalities of telling *whether* the vehicle is actually on that lane. If the driver fails to stay on the correct lane before its next maneuver, s/he would have to reroute or, in the worst case, take abrupt and thus dangerous lane changes to that lane. In order to provide assist the driver to determine whether the vehicle is on the correct lane, `V-Sense` provides the functionalities for fine-grained lane guidance. By integrating `V-Sense` and existing navigation systems, one can determine which lane the driver is currently on, and whether the lane is correct or not, without using cameras.

3. SYSTEM DESIGN

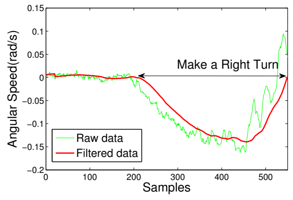
This section details the design and functionalities of `V-Sense`. First, we describe how IMUs on smartphones are utilized to determine whether the vehicle is making turn, changing lane, or driving on a curvy road. Then, we show how `V-Sense` classifies such different vehicle steering maneuvers based on the detection results.

3.1 Coordinate Alignment

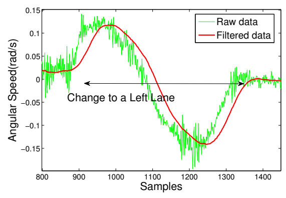
Since the phone's coordinate changes over time, in order to maintain the consistency of analysis, we align the smartphone coordinate ($\{X_p, Y_p, Z_p\}$) with the geo-frame coordinate ($\{X_e, Y_e, Z_e\}$), as shown in Fig. 2. This allows us to simplify the change of the readings from 3 degrees of



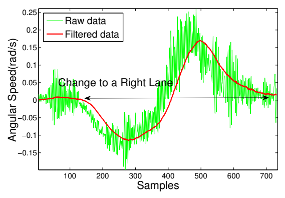
(a) Make a left turn



(b) Make a right turn



(c) Change to a left lane



(d) Change to a right lane

Figure 3: Gyroscope readings when the vehicle makes a left/right turn or left/right lane changing.

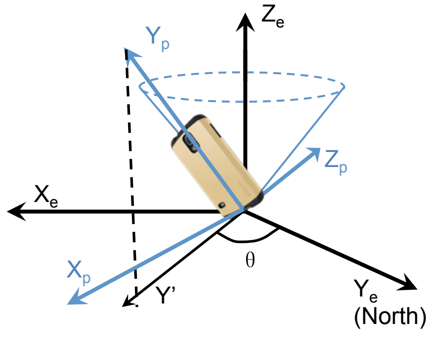


Figure 2: Align the phone coordinate system with the geo-frame coordinate system. This figure was borrowed from [30].

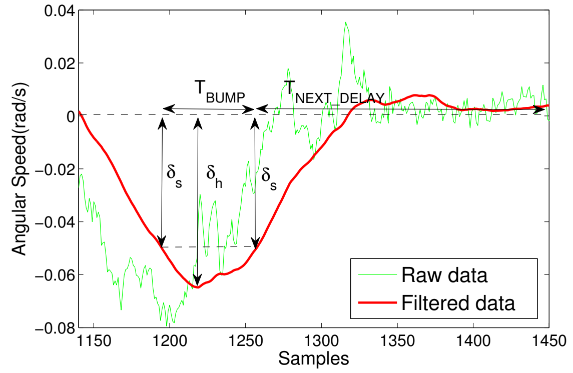
freedom (DoFs) to 1 DoF. The key idea is that with the measurements of the direction of the applied gravity to the smartphone (Y), the smartphone coordinate can be fixed within a cone. Then, combining the result with the angle (θ) derived from the magnetometer readings and the thus-determined rotation matrix, the smartphone coordinate can be aligned with the geo-frame coordinate. We refer interested readers to [30] for detailed formulation of the rotation matrix.

3.2 Bump Detection

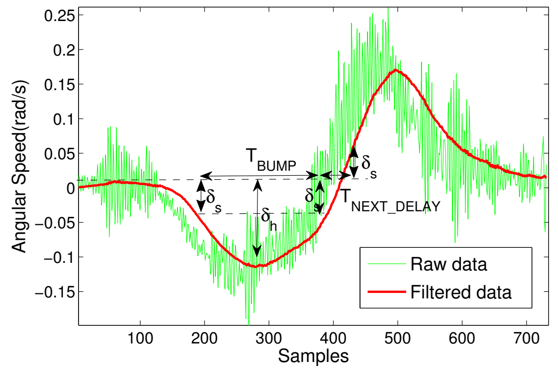
When a car changes its direction via steering (e.g., changing lanes, making turns, and driving on curvy roads), the Z-axis gyroscope reading (i.e., yaw rate reading) on the phone can be utilized to represent the vehicle angular speed of that change of direction. Fig. 3 illustrates the Z-axis gyroscope measurements from the phone during a right/left turn, and changing to a right/left lane, respectively. During a left turn, a counter-clockwise rotation around the Z-axis occurs and thus generates positive readings (i.e., a positive bump), whereas during a right turn, a clockwise rotation occurs and thus generates negative readings (i.e., a negative bump).[2] Similarly, during a left lane change, a positive bump is followed by a negative bump, whereas during a right lane change, the opposite occurs.

Based on this observation, we can infer that by detecting *bumps* in the Z-axis gyroscope readings, we can determine whether the vehicle has made a turn or has changed a lane. The other steering maneuver, i.e., driving on a curvy road, will show a similar *shape* but with a different *size* in terms of

[2]We refer to such a temporal rise/drop, or vice versa, of the yaw rate as *bumps*.



(a) One bump: a turn



(b) Two consecutive bumps: lane change

Figure 4: Statistical features of bumps in gyroscope reading during different steering maneuvers

width and height of the bumps. We will elaborate on how `V-Sense` differentiates such steering maneuvers in Section 3.3.

We adopt a moving average filter to remove noise from the raw gyroscope readings. The delay parameter of the filter is set to 60 samples which correspond to 0.05 second in the time domain. Such a decision was made based on our experimental observation: it is short but good enough to extract the waveform of the bumps.

As shown in Fig. 4, we define four system parameters: δ_s, δ_h, T_{BUMP}, and T_{NEXT_DELAY}. To reduce false positives and differentiate the bumps from jitters, a bump should satisfy the following three constraints for its validity: (1) all the readings during a bump should be larger than δ_s, (2) the largest value of a bump should be no less than δ_h, and (3) the duration of a bump should be no less than T_{BUMP}.

Algorithm 1 Algorithm for Detecting Bumps

```
 1: Inputs:
        State, $Y$ (Yaw rate), System parameters
 2: if State = No-Bump and $|Y| > \delta_s$ then
 3:     (Start of 1st bump)
 4:     State ← One-Bump
 5:     Record the start point of a possible bump
 6: else if State = One-Bump and $|Y| < \delta_s$ then
 7:     Record the end point of a possible bump
 8:     if bump is valid then
 9:         State ← Waiting-for-Bump
10:     else
11:         State ← No-Bump
12:     end if
13: else if State = Waiting-for-Bump then
14:     $T_{dwell}$ ← State dwell duration
15:     if $T_{dwell} < T_{NEXT\_DELAY}$ and $|Y| > \delta_s$ then
16:         (Start of 2nd bump)
17:         if 2nd bump is valid then
18:             Two valid bumps → "Lane change"
19:         else
20:             One valid bump → "Turn"
21:         end if
22:         State ← No-Bump
23:     else if $T_{dwell} > T_{NEXT\_DELAY}$ then
24:         One valid bump → Turn
25:         State ← No-Bump
26:     else
27:         Continue in Waiting-for-Bump state
28:     end if
29: else
30:     Continue in current state
31: end if
```

Based on these constraints of a valid bump, we designed an algorithm as shown in Algorithm 1, which keeps running when `V-Sense` operates. There are three states in the bump detection algorithm: *No-Bump*, *One-Bump*, and *Waiting-for-Bump*.

In *No-Bump* state, we continuously monitor the Z-axis gyroscope readings, i.e., yaw rate. When the absolute value of the measured yaw rate reaches δ_s, we interpret this as the start of a possible bump and the algorithm enters *One-Bump* state.

The *One-Bump* state terminates when the yaw rate drops back to a value below δ_s. If the sojourn/dwell time in *One-Bump* state was larger than T_{BUMP} and the largest measured yaw rate was larger than δ_h, hence satisfying the three constraints, we consider the first detected bump to be valid. In such a case, the algorithm enters *Waiting-for-Bump* state, Otherwise, it returns to *No-Bump*.

In *Waiting-for-Bump* state, it further monitors the yaw rate readings for a maximum dwell time T_{NEXT_DELAY}. In the meanwhile, if another bump starts, i.e., the yaw rate reaching δ_s with a sign opposite to the first bump's is detected, it goes through the same procedure as before in validating it. If determined as valid, this would mean that two consecutive bumps with opposite signs have been detected. Thus, the algorithm determines the maneuver to be a *lane change*. Otherwise, if the second bump turns out to be invalid, then it would mean that only a single valid bump was detected and thus the algorithm determines the maneuver to be a *turn*. After making all decisions, the algorithm goes back to the initial *No-Bump* state.

The bump-detection algorithm is executed iteratively for each collected sample, and goes through a different procedure depending on the current state.

3.3 Differentiating Steering Maneuvers

When the vehicle is steered, bumps in the yaw rate readings are constructed. Based on the bump detection algorithm, `V-Sense` detects such bumps and differentiates between maneuvers of a lane change and a turn.

One possible problem in using this would be when driving on a curvy road. As illustrated in Fig. 5, when driving on a curvy road, it might have the same *shape* of trajectory as in lane change or a turn, and hence construct the same number and shape of bumps. In such a case, `V-Sense` might misinterpret the drive on a curvy road as a lane change or turn, thus yielding false positives/negatives. Therefore, it is imperative for `V-Sense` to differentiate between lane changes, turns, and also driving on curvy roads.

We achieve this by classifying the maneuvers based on not only the number and shape of the bumps as in Algorithm 1, but also with their horizontal displacement.[3] Let W_{LANE} denote the horizontal displacement after a lane change. Since the average lane width is around 3.65 meters [14], W_{LANE} is expected to be around that value after a lane change. In contrast, while driving on a curvy road, the horizontal displacement, denoted as W_{CURVY}, is usually much larger than W_{LANE}. Based on this observation, if `V-Sense` has detected two bumps — which means a possible lane change — it then derives the horizontal displacement during that steering maneuver. If the derived value is larger than 3.65 meters, `V-Sense` determines the vehicle to be driving on a curvy road, rather than making a lane change.

Also, to differentiate between turns at the intersection (i.e., a sharp turn) and driving on a curvy road, we exploit the fact that the horizontal displacement during a turn is much smaller than that during driving on a curvy road. Figs. 5(c) and 5(d) illustrate this, where W_{TURN} and W_{CURVY} represent the horizontal displacements during a turn and driving on a curvy road, respectively. If `V-Sense` detects only one bump, it further examines the horizontal displacement to distinguish between turning and driving on a curvy road.

Note that to differentiate turns from lane changes, only the number and shape of the bumps are required, which can be met by the bump-detection algorithm.

3.4 Horizontal Displacement

In order to correctly distinguish between lane change, turn, and driving on a curvy road, we must determine the horizontal displacement, in addition to the detection of bumps. We derive the horizontal displacement from the readings of a smartphone's gyroscope and accelerometer.

Fig. 6 shows an example vehicle trajectory during a left lane change or maneuver on a curvy road as illustrated in Figs. 5(a) and 5(b). The dotted vertical line represents the time when the sensors are sampled with frequency of $1/T_s$. Here θ_n denotes the angle of the vehicle's heading, whereas v_n represents the average velocity during the sampling pe-

[3] *Horizontal displacement* is a value that represents the change of position in the X-axis after the steering maneuver.

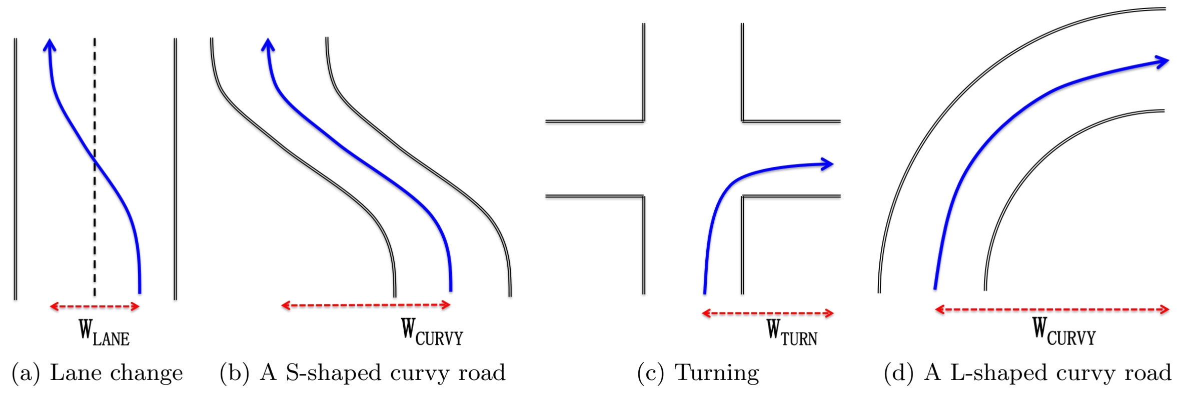


Figure 5: The same vehicle trajectory shape for four different scenarios: (a) lane change, (b) driving on an S-shaped curvy road, (c) turning, and (d) driving on an L-shaped curvy road.

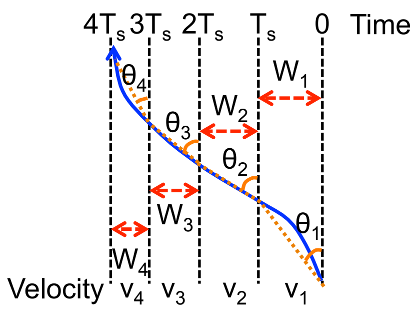


Figure 6: Deriving the horizontal displacement based on gyroscope readings and estimated velocity

riod. During each sampling period T_s, the vehicle's horizontal displacement can be expressed as:

$$W_n = v_n T_s sin(\theta_n). \tag{1}$$

Since the yaw-rate readings from the gyroscope represent the vehicle's angular velocity around the Z-axis, θ_n can be expressed as:

$$\begin{aligned}\theta_n &= \theta_{n-1} + Y_{avg} T_s \\ &\approx \theta_{n-1} + Y_n T_s,\end{aligned} \tag{2}$$

where Y_{avg} represents the average yaw rate during the sampling period, and Y_n the instantaneous yaw rate measured at the end of the sampling period. Note that the above approximation holds since the sampling period on smartphones can be significantly reduced. Thus, the total horizontal displacement from time 0 to NT_s can be derived as:

$$\begin{aligned}W_{final} &= \sum_{n=1}^{N} W_n \\ &= \sum_{n=1}^{N} v_n T_s sin(\theta_n) \\ &= \sum_{n=1}^{N} v_n T_s sin(\sum_{k=1}^{n} Y_k T_s)\end{aligned} \tag{3}$$

where T_s is a predefined parameter denoting the sampling period of the application. The third equality comes from the fact that the initial angle of the vehicle's heading, $\theta_0 = 0$, since this is the reference point. Y_k can be acquired from the gyroscope readings, while v_n can be derived from the accelerometer and GPS readings. We further elaborate on how to obtain an accurate value of v_n in Section 3.6.

We exploit the gyroscope and accelerometer readings to determine W_{final}. Then, by analyzing the thus-determined value, `V-Sense` distinguishes between the cases of lane change or turn and driving on curvy roads. Using in-depth evaluations, we will later show that the derivation of horizontal displacement is accurate for various cases (e.g., lane change, left/right turn, U-turn).

3.5 Change in Vehicle's Heading Angle

Based on bump detection and horizontal displacement, `V-Sense` classifies various steering maneuvers into three classes: lane change, turn, and driving on a curvy road. To further classify different turning maneuvers (e.g., left/right turn at the intersections, U-turn), `V-Sense` derives the change in the vehicle's heading angle, i.e., the difference in the heading angle between the start and the end of a steering maneuver.

As in Eq. (2), the angle of vehicle's heading at sampling time nT_s can be derived by accumulating the n yaw-rate measurements. As an example, consider Fig. 6; at sampling time $3T_s$, the angle of the vehicle's heading would be $\theta_3 = \sum_{n=1}^{3} Y_n T_s$. In other words, the change in the vehicle's heading from time 0 to NT_s can be expressed as:

$$\theta_{final} = \sum_{n=1}^{N} Y_n T_s. \tag{4}$$

For example, after making a left/right turn at the intersection, $\theta_{final} \approx \pm 90^\circ$, whereas after making a U-turn, $\theta_{final} \approx \pm 180^\circ$. Thus, by exploiting the derived values, `V-Sense` can further classify the turns into a left/right turn or a U-turn.

Fig. 7 summarizes the overall maneuver classification in `V-Sense` as a state diagram. `V-Sense` first determines whether the steering maneuver is a turn or a lane change by calculating the number of bumps. If it is a turn, `V-Sense` will calculate the angle change to determine whether the turn is a regular left/right turn or a sharp U-turn. Second, `V-Sense`

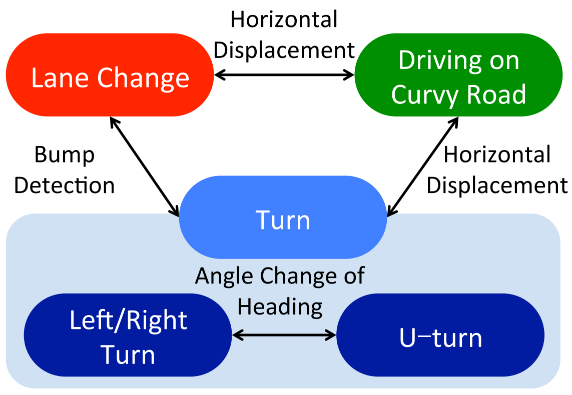


Figure 7: State diagram of maneuver classification in `V-Sense`.

calculates the horizontal displacement to determine whether it is an curvy road or not.

3.6 Velocity Estimation

In order to derive the horizontal displacement and set T_{BUMP} and T_{NEXT_DELAY}, we need accurate measurement of the vehicle's instantaneous velocity.

There are two ways of acquiring the velocity with a smartphone: reading the Speed Over Ground (SOG) output from the GPS module inside the smartphone, or exploiting the IMU. The GPS does provide measurements of the velocity, whereas the acceleration can be derived from IMU readings. However, the GPS output rate is very low, e.g., 1Hz on Samsung Galaxy S4, as shown in Fig. 8, and hence cannot properly capture velocity changes within a sampling period. On the other hand, the IMU has a much higher output rate but contains lots of noise as well as some biases as shown in Fig. 8. Thus, just simply using either the velocity measurement from GPS or taking an integral of the accelerator IMU output is not sufficient. Hence, in order to exploit the distinct advantages of GPS and IMU, we fuse the data by using a Kalman filter [19] to estimate the velocity.

We first construct a model for estimating the velocity:

$$v(k|k-1) = v(k-1|k-1) + (a(k) - b(k-1|k-1))T_s \quad (5)$$

where $v(k|k-1)$ is the estimated velocity at time k based on the optimized velocity at time $k-1$; $v(k-1|k-1)$ is the optimized velocity at time $k-1$; $a(k)$ is the acceleration output at time k; $b(k-1|k-1)$ is the optimized bias of the accelerometer at time $k-1$; T_s is the sampling period of the accelerometer.

Here we treat b as a constant bias [22]:

$$b(k|k-1) = b(k-1|k-1). \quad (6)$$

Thus, we have a matrix representation of the model as:

$$X(k|k-1) = AX(k-1|k-1) + BU(k) \quad (7)$$

where $X = \begin{bmatrix} v \\ b \end{bmatrix}$, $A = \begin{bmatrix} 1 & -T_s \\ 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} T_s \\ 0 \end{bmatrix}$, and U is the output from accelerometer. So, the covariance matrix is estimated by:

$$P(k|k-1) = APA^T + Q, \quad Q = \begin{bmatrix} q_v & 0 \\ 0 & q_a \end{bmatrix} \quad (8)$$

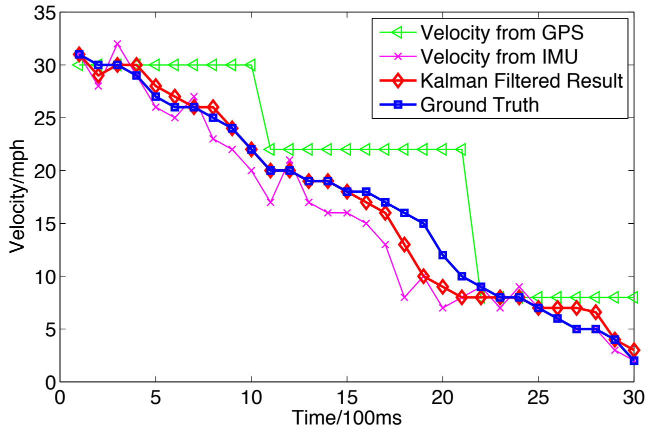


Figure 8: The accuracy of velocity estimation by fusing sensor readings

where P is the covariance matrix, and Q is the covariance of the process noise which can be regarded as the Gaussian white noise. Thus, the state can be estimated as:

$$X(k|k) = X(k|k-1) + g(k)(S(k) - HX(k|k-1)) \quad (9)$$

where $g(k)$ is the matrix of Kalman gain and $S(k)$ is the speed relative to the ground measured by the GPS, and $H = [1 \quad 0]$. We refer the interested readers to [19] for more details.

Fig. 8 shows velocity estimation by using such a model based on Kalman Filter. Here we get the ground truth velocity by directly reading it from the OBD-II port, and compare it with our estimation results. Fig. 8 shows that the velocity can be accurately estimated in real time, thus yielding accurate horizontal displacements.

3.7 Parameter Setting

The bump-detection algorithm uses four main parameters: δ_s, δ_h, T_{BUMP} and T_{NEXT_DELAY}. δ_s determines the start/end point of a bump and thus is the smallest reading value, whereas δ_h determines the largest reading value of the bump, and hence represents its height. Based on the constraints of a *valid* bump, its minimum and maximum should be larger than δ_s and δ_h, respectively.

With large values of δ_h and δ_s, small bumps — which may be caused by background noise or sensing errors — can be ignored and thus reduce the false-positive rate, whereas the false-negative rate might increase. On the other hand, with small values of δ_h and δ_s, the false-negative rate can be reduced but will become susceptible to background noise, thus increasing the false-positive rate. From extensive road tests, we found that parameters of $\delta_s = 0.05$ and $\delta_h = 0.07$ represent a good tradeoff, and are thus used as default values for simplicity. However, the optimal parameter setting may slightly vary with the driving habit. Developing an adaptive parameter selection mechanism is part of our future work.

As for the other two parameters, T_{BUMP} represents the time duration of a valid bump, whereas T_{NEXT_DELAY} represents the maximum waiting time for the following bump, in case of a lane change. Since the time duration of a turn or lane change is usually several seconds [24], we set $T_{BUMP} = 1.5$ seconds and $T_{NEXT\_DELAY} = 3$ seconds as their default values.

	Lane Change				U-turn			
	#1	#2	#3	Average	#1	#2	#3	Average
Displacement [m]	4.29	3.49	3.59	3.79	14.47	15.66	14.46	14.86
Angle Change [deg]	2.03	7.49	4.12	4.54	193.73	179.85	184.41	185.99

Table 1: Determined horizontal displacement and angle change of heading for lane changes/U-turns.

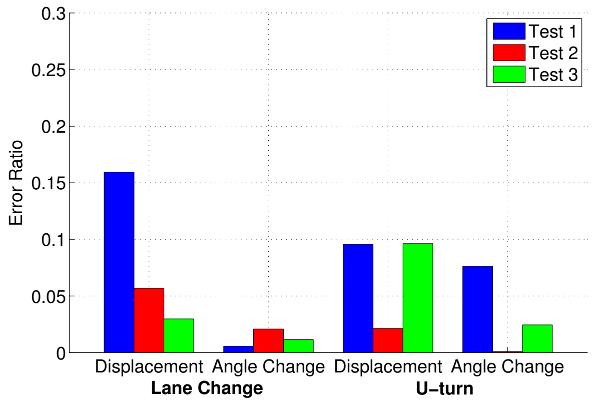


Figure 9: Error of the determined values compared with the ground truth.

4. EVALUATION

To evaluate the performance of V-Sense, we implemented V-Sense on a Samsung Galaxy S4 with a 1.6GHz quad-core processor running Android 4.4.1 KitKat OS. We have conducted a total of 40 hours of test, and tried to cover different environments both in a parking lot and real roads. First, we evaluated the accuracy of V-Sense in determining the change of heading angle and the horizontal displacement in a short road test. Then, we evaluate the performance of V-Sense's classification with a longer road test containing various road features. The cars we used for the test were a 2010 Mitsubishi Lancer and a 2006 Mazda 6. During these experiments, the smartphones were either mounted on the windshield, or kept in the driver's pocket.

4.1 Accuracy of Estimating Angle Change and Displacement

By making several lane changes, turns, and U-turns during a short road test, we evaluated the accuracy of V-Sense in estimating the change of heading angle and the horizontal displacement. During the road test, we made three lane changes, one to the left lane and the other two to the right lane, and three U-turns. We collected the horizontal displacements and changes of heading angle from V-Sense to check whether the estimated values are close to their ground truth. The results of the two separate tests are summarized in Table 1. For consistency, we present all numbers as their absolute values.

During a lane change, the ground truth horizontal displacement is expected to be equal to the actual lane width, which was around 3.7m for our experiment. However, for the change of heading angle, it is expected to be $0°$, since this is a measure of the difference between the initial and the final heading angles.

On the other hand, during a U-turn, the ground truth of horizontal displacement and the change of heading angle are the road width for U-turns, which was approximately 16m in our case, and $180°$, respectively.

Fig. 9 shows the error ratio — which is the ratio of the absolute deviation to the ground truth value — in the two experiments. For all cases, the estimated horizontal displacement and change of heading angle have a very low error ratio, i.e., V-Sense is very accurate.

The high accuracy of V-Sense in determining the two values means that it can correctly classify various steering maneuvers, which is validated in the following subsection by conducting long road tests.

4.2 Accuracy of Maneuver Classification

To evaluate how well V-Sense classifies different steering maneuvers, we performed two long road tests. To guarantee the generality of our experiment, we carefully chose two different test routes as shown in Figs. 10 (a) and (b) near our campus. The routes run through typical urban areas and freeways, the former including residential, downtown, and school areas. The road features are highlighted in both figures with detailed information as follows.

- Left/right turn (LT/RT): Each turn at an intersection exemplifies a left/right turn.
- Curvy road (CR): There are several curvy roads in the chosen routes. Among them, there are two long L-shaped curvy roads on the US#23 freeway, which are challenging for our bump detection scheme alone to determine.
- Multiple traffic lanes (LC): A lane change is possible in both urban areas and on the US#23 freeway. Specifically, there are 2 lanes in each direction in the urban road, and 4 lanes in each direction on the US#23 freeway.

The number of features in the examined routes is summarized in Table 2.

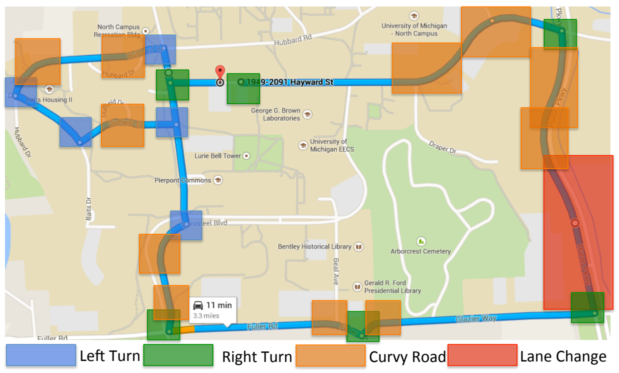
Route	Distance [miles]	RT	LT	LC	CR
#1	3.4	6	5	4	11
#2	8.3	5	5	15	9

Table 2: Summary of different road features in testing routes.

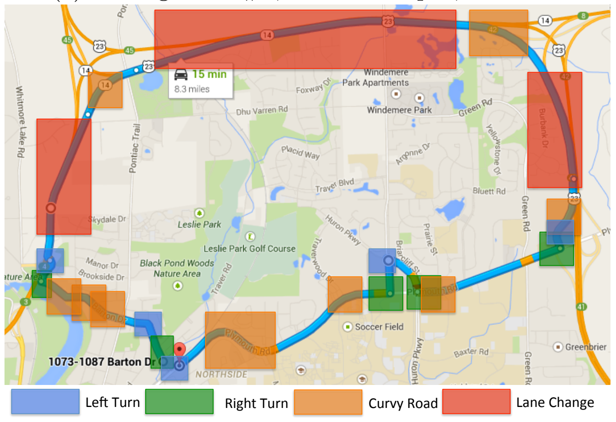
To validate the independence of V-Sense from driving habits, we had 5 volunteers participating in our test, three male drivers and two female drivers. Each of them drove twice on both route #1 and #2. In the first test, they mounted the phone on the windshield, whereas in the second test, the phone was kept inside the driver's pocket.

The on-road experimental results are plotted in Fig. 11, and can be highlighted as follows.

- V-Sense achieves 100% accuracy in detecting both right and left turns, regardless of the phone's placement and



(a) Testing route #1, around campus, 3.3 miles



(b) Testing route #2, freeway included, 8.3 miles

Figure 10: Real road testing routes used for evaluation in Ann Arbor, MI

road condition. This is because when turning, the heights of the bumps in the readings tend to be high enough to be accurately detected and classified.

- For lane changes, `V-Sense` achieves 93% accuracy when the phone is mounted on the windshield, and 85% accuracy when the phone is in the driver's pocket. The false-negative results are mostly due to the fact that `V-Sense` occasionally misinterprets a lane change as driving on a curvy road, because a few of the lane changes in our test took longer than expected, especially on the freeway where drivers tend to take extra caution, thus making slower lane changes. The accumulated error in the gyroscope reading can also degrade the performance in such a case [30]. However, its occurrence is expected to be rare considering the average elapsed time for lane changing, i.e., less than 6 seconds.

- `V-Sense` achieves nearly 97% accuracy in detecting curvy roads with the phone mounted on the windshield, and nearly 92% accuracy with the phone kept in the driver's pocket. These results also reflect the accuracy of the coordinate alignment mentioned in Section 3.1. Also, note that `V-Sense` was able to detect the two long L-shaped curvy roads on the US#23 freeway using bump detection and horizontal displacement derivation.

4.3 `V-Sense` vs. Camera-Based Approach

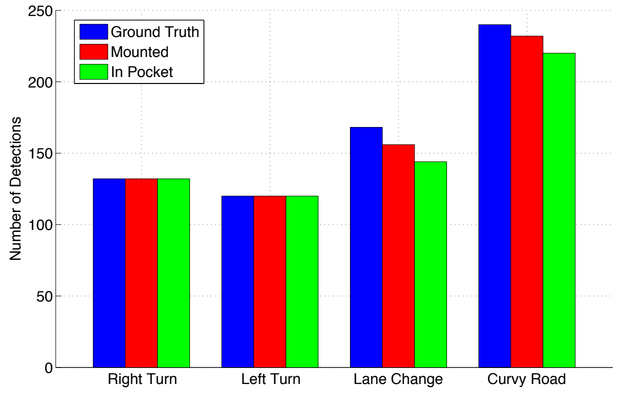


Figure 11: Performance of recognizing different steering patterns on both route #1 and #2

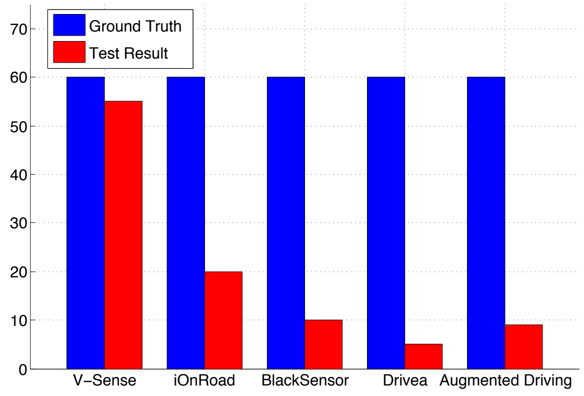
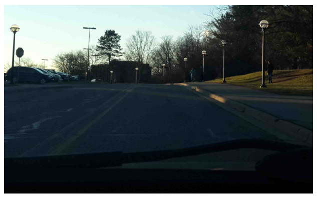


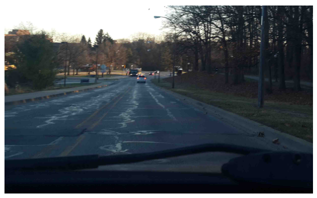
Figure 12: Comparison of `V-Sense`, iOnRoad, BlackSensor, Drivea, and Augmented Driving in lane-change detection.

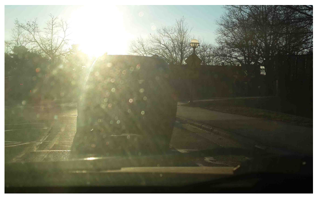
We also compare the performance of `V-Sense`, a camera-free approach, with existing camera-based approaches. Since most of the existing driving assistant applications can only detect lane changes, not turns or driving on curvy roads, we do this comparison with only the results of lane-change detection. The evaluation is based on tests of both route #1 and route #2. We choose iOnRoad [8], BlackSensor [2], Drivea [3], and Augmented Driving [1] for comparison with `V-Sense`. All of these four apps have the capability of detecting lane departures. Of these apps, iOnRoad is the most popular one with more than 1,000,000 downloads from Google Play, and it is still under active maintenance, whereas Drivea is the least popular, still with more than 10,000 downloads.

Since the use of cameras is almost 100% ineffective at night or under a bad weather, we compared the performance of `V-Sense` with difference apps based on the camera approach, during daytime and under a perfect weather condition. Here we used two Samsung Galaxy S4 smartphones: phone A runs `V-Sense` while phone B is running either iOnRoad, BlackSensor, Drivea, or Augemented Driving. In all experiments, phone A was placed next to the driver's seat, while phone B was mounted on the car's windshield to have a clear view of the road.

As shown in Fig. 12, `V-Sense` achieves about 3× better accuracy than iOnRoad, 5× better than BlackSensor and Augmented Driving, and 11× better than Drivea. According


(a) Heavy Shadow


(b) Broken Road


(c) Strong Sun Light

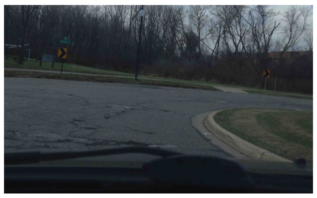

(d) Sharp Turns

Figure 13: Non-functional environments for the camera-based driving assistant application in the experiment.

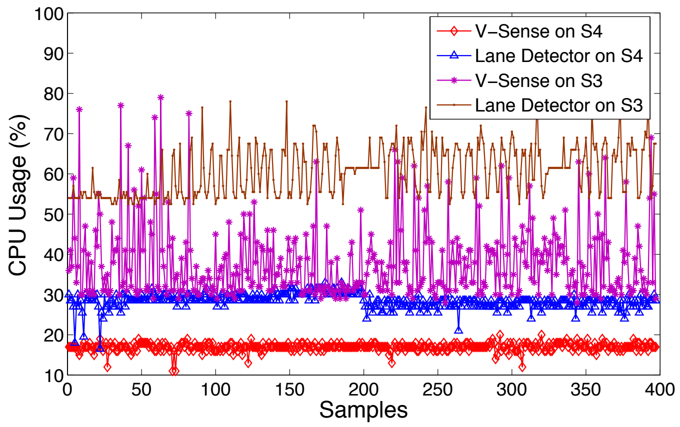


Figure 14: Comparison of CPU usage between V-Sense and lane detector

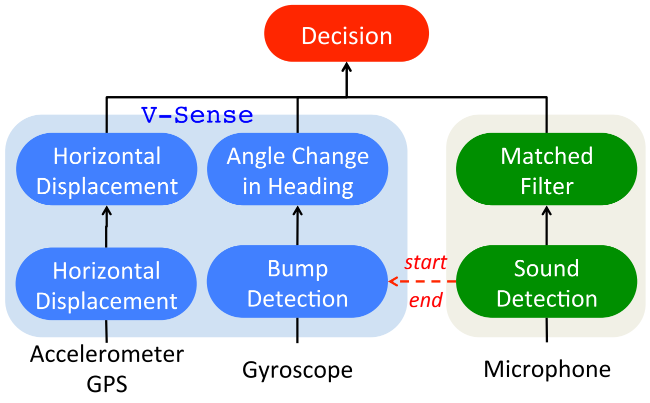


Figure 15: Information flow of careless steering detection by combining V-Sense and sound detection module.

to our experiments conducted even under a perfect weather condition, the performances of all the compared apps were still seriously constrained by the environment, while V-Sense worked well, irrespective of the environment. In particular, the performance of camera-based lane detection degraded severely in at least four cases: heavy shadow on the road, broken road, strong sunlight, and sharp turns as shown in Fig. 13.

4.4 Computational Cost of V-Sense

The high computational requirement is always the problem for existing driver assistant applications. For example, CarSafe [29] indicates its CPU utilization to be 64% when run on Samsung Galaxy S3. We evaluated the computational cost of V-Sense on smartphones, including Samsung Galaxy S3 (with 1.4GHz quad-core Cortex-A9 CPU), and Samsung Galaxy S4 (with 1.6GHz quad-core Cortex-A15 CPU). The CPU utilization was monitored by using `adb top` provided by Android SDK.

For a fair comparison with the camera-based approach, we extracted the lane change detection functionality and implemented a simple application only with the lane-change detection function. The techniques used in this application exploit the phone's rear camera to acquire the road's image, and implement a popular lane-detection algorithm [17] to extract the lane.

Our experimental results are plotted in Fig. 14. We found that, on Galaxy S4, the average CPU utilization of V-Sense was 16.9%, whereas for the lane detector, it was 28.4%. On Galaxy S3, the average CPU utilization of V-Sense was 38.1%, whereas for the lane detector, it was 60.6%. These results show that V-Sense uses 50% less CPU than the camera-based approach. In other words, V-Sense not only achieves higher accuracy but also lower computational overhead than camera-based approaches.

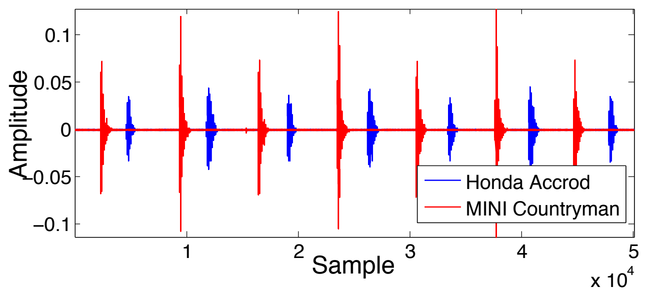
5. APPLICATION I: DETECTION OF CARELESS STEERING

V-Sense can be used to detect careless steering: changing lanes or making turns without turning on the turn signal. Detecting a driver's careless steering is important, since it would enhance the safety of not only the driver but also people/vehicles around him. Moreover, it can also be used by insurance companies in monitoring the driver's driving habit and thus determining the insurance premium accordingly, which would then motivate the drivers to avoid careless steering.

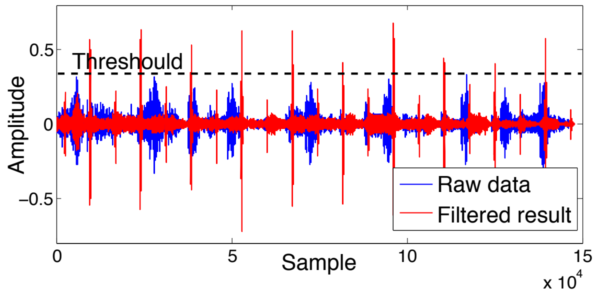
In this section, we present and evaluate a simple proof-of-concept of careless steering detection. We use V-Sense to detect and differentiate various steering maneuvers, such as making turns, changing lanes, and driving on curvy roads. Furthermore, we present a scheme to detect whether the driver has actually used the turn signal during each steering maneuver.

5.1 Overview

Fig. 15 shows the information flow of careless steering detection using data collected from the gyroscope, GPS, accelerometer, and microphone on a smartphone. This application is comprised of V-Sense and sound detection module. V-Sense detects possible lane changes or turns using the gyroscope readings. Upon detecting the start point of a lane



(a) Signal sample



(b) Matched filter result

Figure 16: Turn signal samples and the filtered result by using a matched filter, (a) turn signal sample from a 2006 HONDA Accord and MINI Countryman; (b) matched result from the filter with the existence of background noises inside the car.

change or turn, i.e. bump, the sound detection module activates the microphone and starts to detect the turn signal sound. If a lane change or turn is detected without the detection of signal sound, the application declares the driver is involved in careless driving, and triggers alarm to notify the driver. Otherwise, the application declares it as attentive driving.

5.2 Detection of Turn Signal

In order to detect whether the driver has used the turn signal, `V-Sense` uses the following three steps: (i) collect training samples of the turn signal sound; (ii) eliminate background noise with a matched filter; (iii) make a decision on whether the turn signal was used during the turn or lane change.

5.2.1 Collection of Training Sample Data Set

We first collected the turn signal sounds from two different cars, 2006 Honda Accord and MINI Countryman, which are used as sample data sets as shown in Fig. 16(a). The measured steady rates of the turn signal in the 2006 Honda Accord and MINI Countryman were 161 and 163 ticks per second (shown in Fig. 16), respectively.

As the turn signal sounds acquired from the 2006 Honda Accord has lower amplitude, and would thus be more difficult to detect, we studied the sound detection module using this data set. To test the performance of our turn signal detection module in real driving scenario, we turned on the engine and played music which acts the background noise inside the car.

5.2.2 Elimination of Noise with a Matched Filter

To detect the sound emitted from the turn signals, the detection module has to overcome two challenges: (i) it must be resilient to the variation of SNR due to unpredictable detection conditions; (ii) the delay in detecting a single turn signal must be low in order to be ready for detecting the subsequent signal. We utilized a matched filter to meet these challenges.

The matched filter is used to detect the presence of a known signal in the unknown signals [23]. The key idea behind the matched filter is to design an impulse response that maximizes the output SNR. Due to unpredictable driving conditions, the noise inside the car cannot be easily modeled. Thus, we model the turn signal and extract the signal sound by using one convolution with the matched filter kernel. Since the turn (sound) signal can be modeled as series of discrete signals, we use the discrete version of matched filter, in which the output, $y[n]$, can be expressed as:

$$y[n] = \sum_{k=-\infty}^{\infty} h[n-k]h[k], \tag{10}$$

where the impulse response, $h[n]$, of the matched filter is

$$h[n] = g[n] \otimes v[n_0 - n], \tag{11}$$

where $g[n]$ denotes the power spectral density of background noise. We can thus acquire the matched signal $Matched$ by applying

$$Result[n] = Signal[n] \otimes h[n], \tag{12}$$

where $signal[n]$ is the sound recorded by the smartphone's microphone inside the car and $Result[n]$ is the output of the matched filter.

5.2.3 Making a Decision

If the amplitude of the matched filter output is larger than T, a pre-defined threshold set to 0.35 by default, `V-Sense` declares the detection of a turn signal sound. If the detected turn signal is accompanied by a lane change or turn detected by `V-Sense`, then the application declares the steering maneuver as being attentive. On the other hand, if no such turn signal sound was detected, the application declares the steering to be careless, and then alarms the driver.

5.3 Performance of Sound Detection

Fig. 16(b) shows the performance of our sound detection module which extract signal sound from background noise. We conducted experiments in a regular driving setting, where music played inside the car and the passengers were talking occasionally. The matched filter was able to extract and identify the sound of the turn signals from the background noise, even when the amplitude of the noise was very high (radio played music at the max volume).

By integrating this accurate sound detection module with `V-Sense`, the application detects careless steering and thus enhances driving safety significantly.

6. APPLICATION II: FINE-GRAINED LANE GUIDANCE

In this section, we demonstrate a proof-of-concept fine-grained lane guidance application using `V-Sense`. Fine-grained

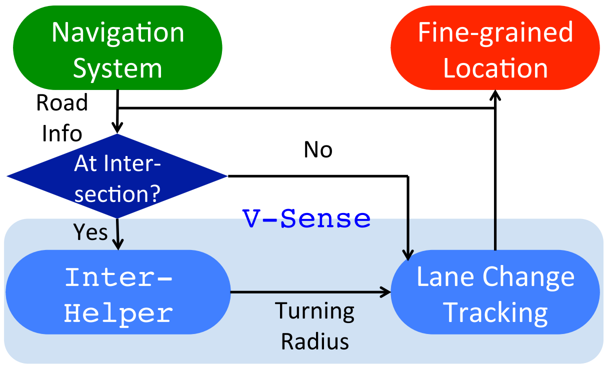


Figure 17: Information flow of fine-grained lane guidance by incorporating navigation system and `V-Sense`.

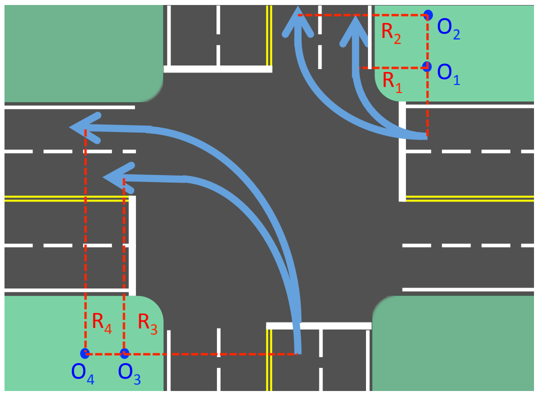


Figure 18: Turns and the corresponding turning radius at a 4-lane single carriageway intersection.

lane guidance allows existing navigation systems provide higher guidance accuracy. Specifically, fine-grained lane guidance detects whether the driver is in the correct lane and alarms the driver if not.

Existing navigation systems on smartphones are constrained by the accuracy of the built-in GPS sensor, which is at best 5∼10m [26]. When the line-of-sight transmission between the satellite and the phone is blocked by obstacles, such as tunnels, bridges, and tall buildings, the accuracy quickly drops to 20∼100m [18]. Such limitations make it impossible for legacy navigation systems to recognize the exact traffic lane that the vehicle is on. The latest update of Google Maps does include a lane guidance function [5], but in a rather limited way: it can only provide information on which lane the vehicle should stay, not whether it is actually on that lane.

We incorporate `V-Sense` in an existing navigation system to provide a true fine-grained lane guidance. Fine-grained lane guidance is important, since it can reduce abrupt lane changes, and also very helpful for drivers who have lack driving experience.

6.1 Achieving Fine-Grained Lane Guidance

Based on information from an on-line map, the correct lane for the next maneuver can be easily determined, which is a function already provided by existing navigation systems. Hence, the main challenge of realizing the fine-grained lane guidance application is the determination of the current lane that the vehicle is running on. To meet this challenge, we need to determine the vehicle's current lane via lane change detection. The current lane can be determined based on whether and how the vehicle has changed its lane. Thus, by detecting and analyzing the lane changes made by the vehicle, we can determine the vehicle's current lane.

Lane changes may take place in two situations: (i) middle of a road or; (ii) at intersections.[4] For the first case, `V-Sense` can reliably detect lane changes on the road using techniques in Section 3. To implement accurate lane tracking for the second case, we develop an add-on module of `V-Sense` called `InterHelper`. In Fig. 17, we show how the navigation system and `V-Sense` cooperate to determine the fine-grained location. The navigation system is capable for determining whether the vehicle is at the intersection. Once the vehicle reaches an intersection, `InterHelper` is triggered and starts to estimate the turning radius, R_i. Note that R_i is equivalent to the horizontal displacement during the turn, which can be derived by using the techniques described in Section 3.4. This information enable `V-Sense` to finally determine the fine-grained location.

[4]Although it is illegal to change lane at intersections in some U.S. states (e.g., California [16]), we still consider such a case for generality.

As shown in Fig. 18, there are four possibilities of lane change at a typical 4-lane single carriageway intersection. That is, each car has two choices of making either right or left turn. Here, we assume the turning trajectory is an arc, which is a common assumption in intersection design [7]. O_1, O_2, O_3 and O_4 are centers of turning circles. `InterHelper` classifies each case by differentiating the turning radius, i.e., R_1, R_2, R_3 and R_4.

For a typical intersection, the right turn radius, R_1 is 10.8m [15], the left turn radius, R_3 is 20.7m, and the width of a typical intersection is 19.2m [12]. Moreover, the lane width is around 3.65m [24]. Based on these facts and extensive road experiments, we set the threshold of differentiating R_1 and R_2 as 13.1m, and the threshold of differentiating R_3 and R_4 as 21.64m. Using such thresholds and the horizontal displacement obtained from `V-Sense`, the application determines whether the vehicle has changed its lane during a turn at the intersection.

6.2 Performance of `InterHelper`

In order to evaluate the performance of the fine-grained lane guidance application, we conducted 80 left and right turns at different intersections in Ann Arbor, Michigan, U.S., and the results are shown in Fig. 19.

The application is shown to be able to detect 95% of right turns with R_1, 90% with R_2, 90% of left turns with R_3 and 85% with R_4. We can therefore conclude that by integrating `InterHelper` into `V-Sense`, the application is capable of detecting lane changes in all cases, thus determining the vehicle's current lane.

7. RELATED WORK

Detecting vehicle dynamics is critical for driving assistant systems and has also been an active research area. The related efforts can be categorized into two main approaches: camera-based or camera-free. For the camera-based approach, the vehicle or a stand-alone device detects the vehicle's maneuver by using its position (determined by the

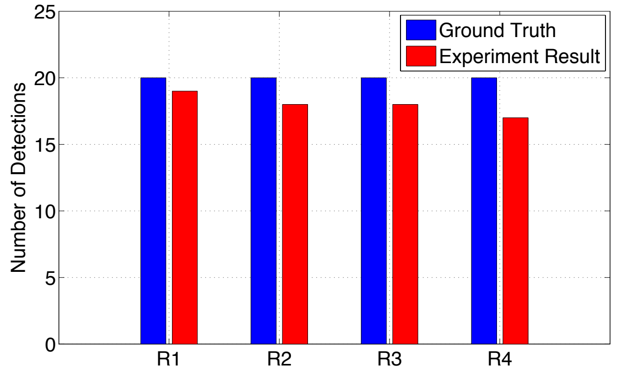


Figure 19: Performance of `InterHelper`.

captured images) with respect to the lane boundaries [6, 10, 9]. Some commercial applications, such as iOnRoad [8] and Blacksensor [2], are capable of detecting lane departures by processing the images taken by cameras on smartphones, and thereby recognizes turns. However, all these methods require a mounted or built-in camera and also a clear view of the road. So, their performance can be seriously undermined if the visibility of the road is poor.

Camera-free systems achieve comparable (to camera-based systems) results regardless of the visibility of the road. The authors of [28] utilized the gyroscope, accelerometer readings of the smartphone and other direct readings from the OBD-II port to detect left/right turns and also the vehicle's real-time velocity. MIROAD [25] also utilized the gyroscope and accelerometer of the smartphone to acquire necessary data to detect vehicle motions via dynamic time wrapping (DTW). The authors of [21] utilized the orientation sensor and accelerometer to detect the driving pattern, thus determining if the driver is intoxicated. In contrast to [28], `V-Sense` is infrastructure-free, i.e., no additional hardware is required. It combines the GPS and IMU readings to acquire accurate and real-time velocity estimation. More importantly, in contrast with the existing work, `V-Sense` differentiates not only between left and right turns but also between lane change, U-turn, and driving on curvy roads. These in turn enable `V-Sense` to accurately handle various scenarios, such as fine-grained lane navigation in Section 6.

Furthermore, recent research and industrial efforts have been focusing on building driving assistant applications on smartphones. CarSafe [29] detects drowsiness by observing the driver's eye movement with the phone's front camera. Commercial applications [8, 2, 3, 1] utilized the phone's front camera to detect the front car and traffic lane, alerting the driver if any dangerous scenario is detected. Most of these applications require the user to mount the phone and use the camera to collect necessary information. This mount-before-go feature could limit the users' willingness, and eventually undermine the application's usability.

In contrast, `V-Sense` is a mount-free design for detecting steering maneuvers regardless of the position of the smartphone. Besides, without image processing, `V-Sense` incurs relatively lower computational cost. This mount-free design could pave s way for the development of many various driving assistant applications. For example, a recent study shows steering patterns could be utilized as a drowsiness indicator [11].

8. CONCLUSION

As an important and emerging subject in both research and industry communities, several driving assistant systems have been proposed. However, the capability of many existing work is limited by its reliance on the visibility of the road objects. Such an approach is only effective when the phone is carefully mounted and also has good visibility, i.e., its performance is undermined by its environment.

In this paper, we proposed `V-Sense`, a camera-free middleware for driving assistant systems. `V-Sense` can accurately and inexpensively detect and differentiate vehicle steering by only utilizing built-in sensors on smartphones. By leveraging an effective bump detection algorithm and studying the nature of steering, `V-Sense` is capable of differentiating various steering patterns, such as lane change, turn, and driving on curvy roads. Based on the camera-free feature of `V-Sense`, we presented two proof-of-concept applications: careless steering detection and fine-grained lane guidance. `V-Sense` provides new functionalities without relying on cameras to provide a broader range of driving assistance.

9. ACKNOWLEDGEMENTS

We would like to thank Morley Mao and Yuanchao Shu of University of Michigan for discussions of this project. We are also grateful to our shepherd Janne Lindqvist and the anonymous reviewers for their helpful suggestions and comments.

10. REFERENCES

[1] Augmented driving. https://itunes.apple.com/us/app/augmented-driving/id366841514?mt=8.
[2] Blacksensor on google play. https://play.google.com/store/apps/details?id=com.chahoo.bsdrive&hl=en.
[3] Drivea app. http://www.drivea.info/.
[4] Fatality analysis reporting system (fars) encyclopedia. http://www-fars.nhtsa.dot.gov/Main/index.aspx.
[5] Google map enable lane guidance. https://support.google.com/gmm/answer/3273406?hl=en.
[6] Honda, advanced driver-assistive system. http://world.honda.com/news/2014/4141024Honda-SENSING-Driver-Assistive-System/.
[7] Intersection design. http://www.deldot.gov/information/pubs_forms/manuals/road_design/pdf/revisions062811/07_Intersections.pdf?100411.
[8] ionroad app. http://www.ionroad.com/.
[9] Mobileye. http://www.mobileye.com/.
[10] Road edge and barrier detection with steer assist. https://www.youtube.com/watch?v=xrLVaWnJmMI.
[11] Steering patterns as drowsy indicator. http://www.sae.org/events/gim/presentations/2012/sgambati.pdf.
[12] Turning radius and intersection size. http://articles.latimes.com/2005/apr/20/autos/hy-wheel20.
[13] Volvo xc90. http://www.volvocars.com/us/cars/new-models/all-new-xc90.
[14] Traffic lane. http://en.wikipedia.org/wiki/Lane.
[15] Turning radius. http://en.wikipedia.org/wiki/Turning_radius.

[16] Lane change at intersection in california. `http://articles.latimes.com/2005/apr/20/autos/hy-wheel20`.

[17] M. Aly. Real time detection of lane markers in urban streets. In *Intelligent Vehicles Symposium, 2008 IEEE*, pages 7–12, June 2008.

[18] C. Bo, X.-Y. Li, T. Jung, X. Mao, Y. Tao, and L. Yao. Smartloc: Push the limit of the inertial sensor based metropolitan localization using smartphone. In *Proc. of ACM Mobicom*, pages 195–198, 2013.

[19] F. Caron, E. Duflos, D. Pomorski, and P. Vanheeghe. Gps/imu data fusion using multisensor kalman filtering: introduction of contextual aspects. *Information Fusion*, 7(2):221 – 230, 2006.

[20] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. In *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, August 2006.

[21] J. Dai, J. Teng, X. Bai, Z. Shen, and D. Xuan. Mobile phone based drunk driving detection. In *International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*, pages 1–8, March 2010.

[22] B. Friedland. Treatment of bias in recursive filtering. *IEEE Transactions on Automatic Control*, 14(4):359–367, Aug 1969.

[23] J. A. Gubner. *Probability and random processes for electrical and computer engineers*. Cambridge University Press, 2006.

[24] S. Hetrick. Examination of driver lane change behavior and the potential effectiveness of warning onset rules for lane change or "side" crash avoidance systems, 1997.

[25] D. Johnson and M. Trivedi. Driving style recognition using a smartphone as a sensor platform. In *International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1609–1615, Oct 2011.

[26] T. Menard, J. Miller, M. Nowak, and D. Norris. Comparing the gps capabilities of the samsung galaxy s, motorola droid x, and the apple iphone for vehicle tracking using freesim mobile. In *IEEE Intelligent Transportation Systems (ITSC)*, pages 985–990, 2011.

[27] R. Ponziani. Turn signal usage rate results: A comprehensive field study of 12,000 observed turning vehicles. *SAE International Paper, 2012-01-0261*, 2012.

[28] Y. Wang, J. Yang, H. Liu, Y. Chen, M. Gruteser, and R. P. Martin. Sensing vehicle dynamics for determining driver phone use. In *Proc. of ACM MobiSys*. ACM, 2013.

[29] C.-W. You, N. D. Lane, F. Chen, R. Wang, Z. Chen, T. J. Bao, M. Montes-de Oca, Y. Cheng, M. Lin, L. Torresani, and A. T. Campbell. Carsafe app: Alerting drowsy and distracted drivers using dual cameras on smartphones. In *Proc. of ACM MobiSys*. ACM, 2013.

[30] P. Zhou, M. Li, and G. Shen. Use it free: Instantly knowing your phone attitude. In *Proc. of ACM Mobicom*. ACM, 2014.